

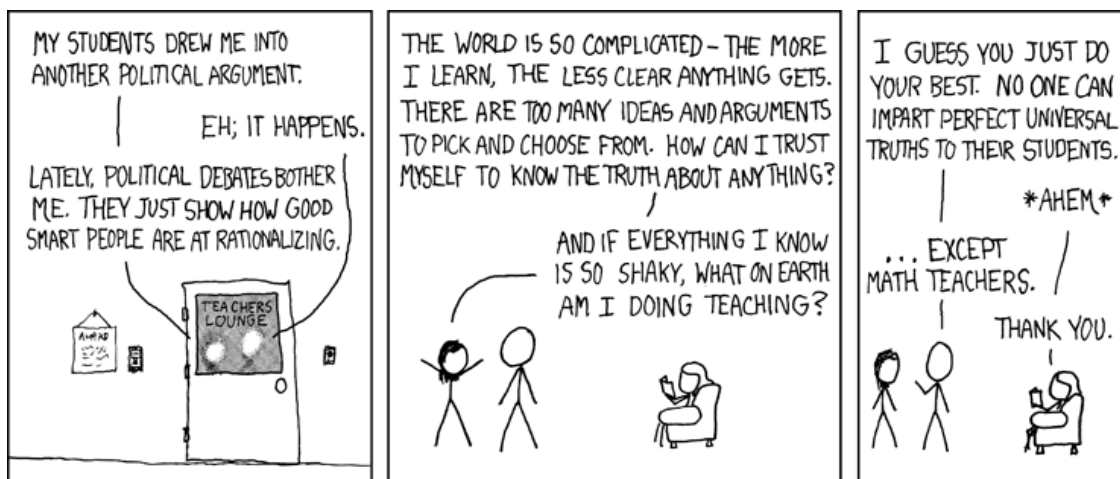
MAT 112

Integers and Modern Applications for the
Uninitiated

Sebastian Pauli *et al*

August 18, 2026

Figure 0.0.1: *Certainty* by R. Munroe (<https://xkcd.com/263>).



$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$. Politicize that, ...

Contents

Preface	7
I Integers and Algorithms	13
1 Foundations	17
1.1 Integers	17
1.2 Statements	19
1.3 Variables	22
1.4 Exponentiation	27
2 Algorithms	33
2.1 Definition of an Algorithm	33
2.2 The Instruction return	35
2.3 The Conditional if_then	36
2.4 The Assignment let_ $_$:=	38
2.5 The Loop repeat_until	39
2.6 Exponentiation Algorithm	44
3 Division	47
3.1 Quotients and Remainders	47
3.2 Division Algorithm	48
3.3 Long Division	52
3.4 The Operation mod	55
3.5 Clock Arithmetic	59
3.6 Application: ISBN	61
4 Divisors	65
4.1 Divisibility	65
4.2 Greatest Common Divisors	66
4.3 The Euclidean Algorithm	68
4.4 Bézout's Identity	70

II	Sets and Functions	73
5	Sets	77
5.1	Definition of a Set	77
5.2	Roster Form	78
5.3	Membership and Equality	78
5.4	Special Sets	80
5.5	Set-Builder Notation	81
6	More on Sets	83
6.1	Subsets	83
6.2	Cartesian Products	84
6.3	Applications of Cartesian Products	85
7	Functions	89
7.1	Definition of a Function	89
7.2	Graphs of Functions	92
7.3	Equality of Functions	94
7.4	Composite Functions	96
7.5	Identity Functions	98
7.6	Inverse Functions	99
8	Codes	105
8.1	Character Encoding	105
8.2	Symmetric Key Cryptography	107
8.3	Caesar Cipher	107
8.4	Other Substitution Ciphers	111
8.5	Frequency Analysis	114

III	Numbers and Counting	117
9	Cardinality	121
9.1	Definition of Cardinality	121
9.2	Infinite Sets	123
9.3	Cardinality of Cartesian Products	124
9.4	Number of Subsets	125
10	Primes	127
10.1	Definition of a Prime	127
10.2	Prime Factorization	128
10.3	Infinitude of Primes	132
10.4	The Twin Prime Conjecture	133
11	Other Bases	137
11.1	Decimal Numbers	137
11.2	Binary Numbers	138
11.3	Conversion from Base 10 to Base 2	140
11.4	Base b Numbers	142
11.5	Conversion from Base 10 to Base b	145
12	Applications of other Bases	149
12.1	Images	149
12.2	Colors	151
12.3	Text	155

IV Groups and Cryptography	159
13 Binary Operations	163
13.1 Definition	163
13.2 Associativity	164
13.3 Identity	166
13.4 Inverses	168
13.5 Commutativity	170
14 Groups	173
14.1 Definition of a Group	173
14.2 Examples of Groups	174
14.3 Modular Addition and Multiplication	176
14.4 The additive groups $(\mathbb{Z}_n, \oplus)$	177
14.5 The multiplicative groups $(\mathbb{Z}_p^\otimes, \otimes)$	178
15 Powers and Logarithms	183
15.1 Exponentiation	183
15.2 Repeated Squaring	185
15.3 Fast Exponentiation	188
15.4 Discrete Logarithm	192
16 Public Key Cryptography	199
16.1 Introduction	199
16.2 Diffie-Hellman key exchange	202
16.3 ElGamal Encryption System	206
Table of Symbols	211
Figures	214
Index	217

Preface

Die ganzen Zahlen hat der liebe Gott gemacht, alles andere ist Menschenwerk
(God made the integers, all else is the work of humans.), Leopold Kronecker,
1886¹

Following Kronecker’s premise, we assume knowledge of the integers along with the operations addition (plus), subtraction (minus), and multiplication (times).

Humans are curious and not content with this. They start asking further questions about the integers. With these questions the work of humans begins and thus mathematics begins. One natural question to ask is: “How many integers are there?” We find that we cannot name a greatest integer, since anytime we have a candidate for greatest integer, we can add one to it and thus obtain a larger integer. This leads to the concept of infinity. The positive integers also exhibit the property that we can always find a greater positive integer. A next question is: “Are there infinitely many positive integers?” The work of humans continues in the form of a creative process. A notion for collections of numbers (and other objects) having the same (infinite) “size” is introduced. With that notion there are as many positive integers as there are integers. We will give the details on this at the end of Part II.

These Notes

In these notes we give you examples of the work of humans that is called mathematics. Some will seem like strange constructions. We give practical applications of all of these.

The topics do not depend on any other mathematical knowledge from courses the students have taken in the past. The presentation is rigorous but basic enough for its intended audience to follow. The content of the course includes applications that are relevant for the digital age as well as pure mathematics that are linked to other liberal arts disciplines. The course culminates with the topic of public key cryptography.

Student Learning Outcomes

We give the *student learning outcomes* (SLOs) of a course based on these notes.

Upon completion of this course students will be able to:

- (1) Compute with integers, sets, and functions, and in groups.
- (2) Apply integers, sets, functions, and groups in the encoding and encryption of information
- (3) Communicate statements about integers, sets, functions, and groups

¹kronecker.

After the title of each section there is such a yellow box that contains a list of tasks that a student should be able to perform after working through the section and the corresponding exercises. These student learning outcomes tell you what you should be able to do and thus are very useful in preparing for exams.

Some of the student learning outcomes require other topics from the chapter that are not explicitly mentioned in the outcomes themselves. You will need a good understanding of all the topics covered in the chapter in order to master the topics in the student learning outcomes.

Overview

We give an overview of the topics covered in this course.

Part I Integers and Algorithms We recall what the integers are and use them as examples when we introduce foundations of mathematical language (Section 1). We introduce instructions that we use in the formulation of algorithms throughout the course and apply them in the formulation of algorithms for integers (Section 2). As a particularly important algorithm for integers, the division algorithm and applications of its output are given (Section 3). These are the tools we need to formulate the Euclidean algorithm for computing greatest common divisors (Section 4).

Part II Sets and Functions Sets are one of the fundamental structures in mathematics. After introducing basic notation and definitions for working with sets in Section 5, we introduce subsets and Cartesian products (Section 6). Functions are used in mathematics to assign each element in one set to an element in another set. Often they are used to change the representation of objects. We start with definitions and properties of functions in Section 7. We apply functions in the encoding and encryption of texts (Section 8).

Part III In Section 9 we define the cardinality of sets, talk about the cardinality of infinite sets. We introduce prime numbers and some of their applications (Section 10). We show that there are infinitely many prime numbers and present the Twin Prime Conjecture. The Twin Prime Conjecture is a mathematical statement that is believed to be true, but has not been proven yet. In Section 11 we discuss different representations of integers and apply those in the encoding of colors, images, and text in Section 12.

Part IV Groups and Cryptography In the last chapter we introduce a new mathematical structure called a group. Although groups are very simple structures, they have many practical applications, some of which we present at the end of the chapter. We define binary operations, which are a specific kind of function, in Section 13. In Section 14 we introduce groups that consist of a set and a binary operation that fulfills certain properties. We give further applications of the operation mod from Part I and show that some finite sets of integers with operations based on addition, multiplication and mod form groups. We generalize exponentiation (introduced in Section 1), present an exponentiation algorithm that is more efficient than the algorithm for this purpose from Section 2, and demonstrate that, in groups, it is more difficult to compute loga-

rithms (the inverse function of exponentiation) than powers (Section 15). In Section 16 we present a method for exchanging encryption keys and a public key crypto system whose security is based on the difficulty of computing discrete logarithms in groups. These two systems are widely used in everyday life.

Definitions and Theorems

If we think of mathematics as a building, then *definitions* provide the foundation, *theorems* are the bricks, and logic is the mortar that connects them and holds them together. Definitions introduce terminology to define mathematical objects and properties. Theorems are statements about defined objects. A theorem uses defined terms and is derived from a sequence of logical arguments using definitions and other, previously proven theorems. To prove a theorem is to construct a sequence of logical arguments that make it a true statement (there can be more than one such sequence). The sequence of logical arguments used to derive the theorem is called a proof of the theorem.

In this course we do not expect you to come up with new theorems or to be able to prove known theorems. Nevertheless we will prove most theorems in these notes, if only to show you that everything follows from the definitions in a sequence of logical steps. Proofs of theorems are either given after the theorems (they start with *Proof.* and end with $\square$) or the argument for the correctness is given before the statement.

Although it is possible to give definitions of the integers and their arithmetic and to prove their properties, we will assume familiarity with them.

We would also remark that all the definitions presented here are man-made and, to some extent, arbitrary. We use these particular definitions because they work and help us solve problems that we can formulate in the language of mathematics. It remains a constraint, of course, that the definitions have to work together so that we obtain a structurally-sound mathematics building. The logical consistency and the precise nature of the definitions we choose to use and the theorems that we can prove starting with them give us the certainty that is unique to the discipline of mathematics as referred to in Figure 0.0.1.

Examples, Problems, and Exercises

Definitions sometimes are quite abstract. We illustrate the objects or properties or operations defined by giving concrete examples. Similarly we demonstrate what the statement of a theorem does in examples. The examples are formulated as *examples* or as *problems* with solutions. In addition we also give *exercises* without solutions. While, some of the examples are instructional, in general, what is done in the examples and problems is close to what you are expected to do in the assignments and on exams.

Navigation

In these notes text in red is a hyperlink. These hyperlinks allow you to navigate within the notes. On the bottom left of every page you can find hyperlinks to [Contents](#) with direct links to Parts [1](#), [2](#), [3](#), and [4](#) in the table of contents. On the bottom right there are links to the Table of [Symbols](#), List of [Figures](#), and the [Index](#). All the entries in the table of contents, the index, and the table of symbols are hyperlinks. Also you will find cross references to theorems and examples throughout the text. Hyperlinks to sources outside the notes are given with the full URL (universal resource locator) in purple.

The definitions and theorems in this text often build on other definitions and theorems. Also in other places you will find that we refer to statements that were previously presented in the text. We reference algorithms, definitions, examples, problems, and theorems by their number. All these are hyperlinks that let you easily jump to the location in the text that was referenced.

xkcd Comics

Throughout these notes you find *xkcd comic strip* by Randall Munroe related to the material. For the complete collection of strips see

<https://xkcd.com>

The web site *explain xkcd* has detailed explanations of all xkcd comic strips:

<http://www.explainxkcd.com>

Acknowledgments

The first version of these notes were written in collaboration with Beth Lewis for Fall 2016. Without her this project would have never gotten off the ground. We thank Beth for the work she put into the notes and the many fruitful discussions. We would like to thank Dan Yasaki for many constructive conversations about the material in this course and for letting us use parts of the notes from his discrete mathematics course. Jonathan Milstead and Tracey Howell also let us use their notes for earlier versions of this course. We thank Sandi Rudzinski and James Rudzinski for the revisions they made for Fall 2017 and Sandi Rudzinski, Nathan Fontes for proofreading the notes for Winter 2018, and Aaron Rapp for improvements made in Spring 2019. We thank Tom Lewis for his work on diagrams in Part [I](#) and [IV](#). Further thanks go to Frances Clerk, Victoria Hayes, Jonathan Milstead, and Debbie White.

The writing of the notes was supported by the Department of Mathematics and Statistics at UNCG and an *Open Education Mini-Grant* from the UNCG Library.

Figure 0.0.2: *Forgot Algebra* by R. Munroe (<https://xkcd.com/1050>).



The only things you **HAVE** to know are how to make enough of a living to stay alive and how to get your taxes done. All the fun parts of life are optional.

Part I

Integers and Algorithms

In these notes we assume some familiarity with the integers and the operations plus, minus, and times. We first cover properties of the integers that will be familiar to most students and use them to introduce the language of mathematics. We discuss basic notions about mathematical statements that will help you to read definitions and theorems (Chapter 1). We then introduce a way of formulating algorithms and encounter some basic algorithms for computations with integers (Chapter 2). This is followed by algorithms for division, applications of the output of the division algorithm in Chapter 3, and an algorithm for computing greatest common divisors (Chapter 4). As an application of the latter we present Bézout’s identity.

Chapter 1

Foundations

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Recognize whether simple statements about the integers are true or false.
- (2) Give examples that illustrate true statements about the integers.
- (3) Give counterexamples for false statements.
- (4) Compute powers of integers.
- (5) Apply the properties of exponentiation.

This section has two purposes. It is a reminder of some of the more basic mathematics that you have learned growing up at the same time we give an introduction to the language of mathematics. In particular we discuss statements about integers and definitions of properties of integers. We introduce variables as placeholders for integers and explain how they are employed in mathematics. We end the section with an introduction to exponentiation and properties of exponentiation.

Although in this section we work with integers, the same mathematical language is used when considering other mathematical objects. Sets, functions, and groups are other mathematical objects that we will consider in coming chapters.

1.1 Integers

In mathematics symbols are used to obtain a clearer and shorter presentation. The first of these symbols is the *ellipses* (...). When we use this symbol in mathematics, it means “continuing in this manner.” When a pattern is evident, we can use the ellipses (...) to indicate that the pattern continues. We use this to define the integers.

The *integers* are the numbers

$$\dots, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, \dots$$

The *natural numbers*, or *positive integers* are:

$$1, 2, 3, 4, 5, 6, 7, \dots$$

The *negative integers* are:

$$\dots, -7, -6, -5, -4, -3, -2, -1$$

The integer 0 is not considered to be positive or negative.

Figure 1.1.1: (a) the integers on the number line (b) the natural numbers (or positive integers) on the number line (c) the negative integers on the number line

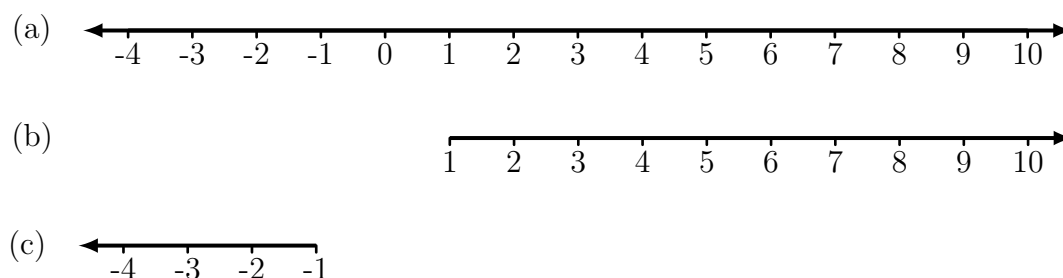


Figure 1.1.1 (a) shows the integers on a number line, which extend both to the left and to the right. Figure 1.1.1 (b) shows the natural numbers (positive integers), which extend only to the right. Figure 1.1.1 (c) shows the negative integers, which extend only to the left.

1.1.1 Comparing Integers

The symbols $=$, $\neq$, $<$, $\leq$, $>$, and $\geq$ are used to compare integers.

They are read as follows:

symbol	read as
$=$	“is equal to”
$\neq$	“is not equal to”
$>$	“is greater than”
$\geq$	“is greater than or equal to”
$<$	“is less than”
$\leq$	“is less than or equal to”

The first symbol is the equality symbol, $=$. Two integers are equal if they are the same integer. To indicate that two integers are not equal we use the symbol, $\neq$.

The other symbols compare the positions of two integers on the number line. An integer is greater than another integer if the first integer is to the right of the second integer on the number line. An integer is less than another integer if the first integer is to the left of the second integer on the number line.

Example 1.1.1. We give examples of comparisons and how to read them.

- (i) $2 = 2$ is read “2 is equal to 2.”
- (ii) $2 \neq 3$ is read “2 is not equal to 3.”
- (iii) $3 > 2$ is read “3 is greater than 2.”
- (iv) $3 \geq 2$ is read “3 is greater than or equal to 2.”
- (v) $2 < 3$ is read “2 is less than 3.”
- (vi) $2 \leq 3$ is read “2 is less than or equal to 3.”

1.2 Statements

Mathematical statements are declarative sentences that are either *true* or *false*. The statements are formulated in such a way that any reader, who knows what all the words mean, can understand them.

Example 1.2.1. (i) “Victoria likes cookies.” is a declarative sentence, and it is either true or false, so it is a statement.

(ii) “Broccoli is green.” is a declarative sentence and it is true, so it is a statement.

(iii) “Broccoli is pink.” is a declarative sentence and it is false, so it is a statement.

(iv) “Cookies!” is not a declarative sentence, so it is not a statement.

In this section we concentrate on statements about the integers.

Example 1.2.2. Consider the following:

(i) “2 is equal to 3.” is a statement. It is false.

(ii) “2 plus 3 is equal to 5.” is a statement. It is true.

(iii) “2 plus 3” is not a statement, as it is not a declarative sentence; it is not even a sentence, as it does not contain a verb.

When we write a statement using the symbols $=$, $\neq$, $<$, $\leq$, $>$, or $\geq$, the comparison symbol takes the place of the verb. A mathematical statement always has a verb or a symbol that takes the place of the verb, just as a sentence does.

Example 1.2.3. We formulate Example 1.2.2 using symbols.

(i) “ $2 = 3$ ” is a statement. It is false.

(ii) “ $2 + 3 = 5$ ” is a statement. It is true.

(iii) “ $2 + 3$ ” is not a statement.

Example 1.2.4. We identify whether statements about integers are true or false.

(i) $2 = 2$ is true.

(ii) $2 = 3$ is false.

(iii) $2 > 3$ is false.

(iv) $-2 < -3$ is false.

(v) $2 \geq 3$ is false.

(vi) $2 \neq 3$ is true.

(vii) $2 \neq 2$ is false.

(viii) $2 \leq 2$ is true.

If a statement is true, we usually do not write “is true.”

Problem 1.2.5. *Decide whether the following are statements or not. If they are statements decide whether they are true or false.*

(i) “Sunflower”

(ii) “Stop signs are red.”

- (iii) “2 is equal to 3.”
- (iv) $(1 + 2) - 4687$
- (v) $2 + 3 = 7$
- (vi) $3 > -100$

Solution. (i) “Sunflower” is not a sentence, so it is not a statement.
(ii) “Stop signs are red.” is a declarative sentence, so it is a statement. It is true.
(iii) “2 is equal to 3” is a declarative sentence, so it is a statement. As $2 \neq 3$ the statement is false.
(iv) $(1 + 2) - 4687$ is not a statement as it has no verb.
(v) $2 + 3 = 7$ is a statement, the verb is ‘=’ (is equal to). As $2 + 3 = 5$ it is a false statement.
(vi) $3 > -100$ is a statement, the verb is “>” (is greater than). It is a true statement.

1.2.1 Operations

Addition, negation, subtraction, and multiplication are the basic operations of integers. We write “+” for plus, “−” for minus, and “.” for times.

Example 1.2.6. We give some examples of statements that involve integer operations. As we do not say “is false,” we mean that all of these equality statements are true.

- (i) $2 + 3 = 5$ is read “2 plus 3 is equal to 5”
- (ii) $2 + 0 = 2$ is read “2 plus 0 is equal to 2”
- (iii) $2 + (-2) = 0$ is read “2 plus negative 2 is equal to 0”
- (iv) $2 - 2 = 0$ is read “2 minus 2 is equal to 0”
- (v) $2 \cdot 5 = 10$ is read “2 times 5 is equal to 10”
- (vi) $2 \cdot (-5) = -10$ is read “2 times negative 5 is equal to negative 10”
- (vii) $(-2) \cdot (-5) = 10$ is read “negative 2 times negative 5 is equal to 10”

Multiplication of a natural number with an integer can be viewed as repeated addition.

Example 1.2.7. We give examples of multiplication viewed as repeated addition.

- (i) $3 \cdot 5 = 5 + 5 + 5 = 15$
- (ii) $3 \cdot (-5) = (-5) + (-5) + (-5) = -15$
- (iii) Again, we can use ellipses (...) to represent a continuing pattern:

$$100 \cdot 5 = \underbrace{5 + 5 + \dots + 5}_{100 \text{ times}} = 500.$$

Defining the multiplication of two negative integers is more difficult, and we appeal to your previously acquired knowledge about integers for that. Recall that the product of two negative integers is positive.

Example 1.2.8. We give examples of multiplication of integers and negative integers:

- (i) $3 \cdot (-5) = -15$
- (ii) $(-3) \cdot 5 = -15$
- (iii) $(-3) \cdot (-5) = 15$

1.2.2 Expressions

A mathematical *expression* consists of objects and operations. The objects can be numbers or variables (see the next section) and the operations can be, for example $+$, $\cdot$, or $-$. Unlike a statement, an expression has no comparison symbol, that means it has no “verb.” So expressions by themselves are not true or false, but expressions can be used in statements, as in Example 1.2.6.

Example 1.2.9. We give some examples of expressions and statements and identify them.

- (i) “ $2 + 3$ ” is an expression.
- (ii) “ $2 + 3 = 5$ ” is a statement.
- (iii) “ $2 + 1 + 5$ ” is an expression.
- (iv) “ $2 + 1 + 5 < 10$ ” is a statement.

1.2.3 Compound Statements

In mathematics we often deal with multiple statements that overlap. In these cases instead of writing each statement separately, we often write them as one string of statements. This allows us to connect the statements directly.

Example 1.2.10. Instead of writing “ $2+3 = 5$ ” and “ $5 = 1+4$,” we write “ $2+3 = 5 = 1+4$.”

We can also do this with inequalities.

Example 1.2.11. Writing “ $2+5 = 7 < 10$ ” means both “ $2+5 = 7$ ” and “ $7 < 10$.” In words, “2 plus 5 is 7 and 7 is less than 10.”

Compound statements are often used to prove identities, that is, when proving that two expressions are equal. The proof of Theorem 1.4.5 in the next chapter is written that way.

1.2.4 Order of Operations

We use parentheses to indicate the order in which expressions should be executed. We evaluate the expressions in the innermost parentheses first and then work our way outwards.

Example 1.2.12. We give examples for order of operations. The numbers and the operations are the same; only the grouping of the expressions given by the parentheses differs.

- (i) $(2 + 3) \cdot 4 = 5 \cdot 4 = 20$
- (ii) $2 + (3 \cdot 4) = 2 + 12 = 14$

Example 1.2.13. We give examples for order of operations. The numbers and the operations are the same; only the grouping of the expressions given by the parentheses differs.

- (i) $5 \cdot (2 + (3 \cdot 4)) = 5 \cdot (2 + 12) = 5 \cdot 14 = 70$
- (ii) $(5 \cdot 2) + (3 \cdot 4) = 10 + 12 = 22$

Figure 1.2.1: *Mnemonics* (excerpt) by R. Munroe (<https://xkcd.com/992>).



It follows from the associative property of addition that the order of operations does not matter for repeated addition. Likewise the associative property of multiplication tells us that the order of operations does not matter for repeated multiplication. We recall these properties in the next subsection (Examples 1.3.11 and 1.3.13).

Example 1.2.14. We illustrate that the order of operations does not matter for repeated addition by computing the same sums in the order indicated by the parentheses.

- (i) $((1 + 2) + 3) + 4 = (3 + 3) + 4 = 6 + 4 = 10$
- (ii) $1 + ((2 + 3) + 4) = 1 + (5 + 4) = 1 + 9 = 10$
- (iii) $(1 + 2) + (3 + 4) = 3 + 7 = 10$

Usually we write $1 + 2 + 3 + 4 = 10$.

In most cases we will use parentheses to indicate the order of operations. There are other conventions for implicit order of operations (see Figure 1.2.1). One of these conventions is that multiplication is performed before addition and subtraction. We will use this convention when we feel that the additional parentheses will make it hard to read the expressions under consideration.

1.3 Variables

Variables are placeholders for mathematical objects. In this chapter variables will be placeholders for integers. We use the characters $a, b, c, \dots, z$ and $A, B, C, \dots, Z$ as variables. Note that when we use a letter as a variable, it is written in italics. We use variables in several ways, which we describe below. Sometimes we simply want to give a value a name. If we do not assign a concrete value, for example a number, to a variable, we specify what values the variable can have, for example, a natural number. Sometimes, we use a variable that does not have a concrete value in a mathematical statement, such as an equation or an inequality. Finding a solution to such an equation or inequality means finding values for the variables that make the equation or inequality true.

1.3.1 Assigning a Value to a Variable

The most concrete use of variables is to assign a value to a variable. To assign a value to the variable a we say or write “let a be ___” or “let $a :=$ ___.”

The symbol $:=$ is used for assignment, which indicates that an action is taking place. The symbol $=$, which indicates equality, is used in statements. The assignment “let $a :=$ ___” changes the value of the variable a .

Example 1.3.1. We write “let a be 65536” or “let $a := 65536$ ” to assign the value 65536 to a . Both notations mean exactly the same thing.

After assigning this value to a , the statement $a = 65536$ is true.

However we can change the value of a . Let $a := 32$. Now $a = 65536$ is false and $a = 32$ is true.

1.3.2 Equality and Substitution

When we have a true equality statement, such as $a = 32$ at the end of Example 1.3.1, we can replace 32 with a (or a with 32) in other statements and expressions. This replacement is called substitution, and it is a fundamental principle in mathematics that we will use, often without explicitly mentioning that we have substituted one expression by another expression that is equal to the first. For example, we have already used substitution when evaluating the expressions in Example 1.2.12. We replaced $2 + 3$ with 5 because $2 + 3 = 5$ is a true equality.

Example 1.3.2. We give an example of using substitution. Let $a := 32$. Since $32 + 7 = 39$ is a true equality statement, $a + 7 = 39$ is also a true equality statement.

1.3.3 Variables in Definitions

In definitions we introduce new terminology for objects, properties, and operations.

First we state what kind of object we are talking about. To be able to refer to the object in the definition, we give it a variable name. Then we state the definition of the property using the variable name.

Example 1.3.3. We give an example of a definition of a property for a real-world object, a cup, using the language of mathematics.

Let c be a cup. When we say c is *full*, we mean that you cannot put anything else in c without spilling over.

The variable c is defined to be a cup (any cup). We are defining the property full, so full is in italics. We define full to mean that you cannot put anything else in the cup. If you had a cup, you could test to see if you could put anything else in it or not.

In the following definition we define the property *non-negative* for an integer. With the first sentence, “Let a be an integer,” we indicate for what kind of object we want to define a property. In the second sentence we refer to the integer a and give the condition under which it is called non-negative.

Definition 1.3.4. Let a be an integer. When we say a is *non-negative*, we mean that $a \geq 0$.

Now instead of saying “ a is an integer and $a \geq 0$,” we can say “ a is a non-negative integer.” In this example the statement that uses the definition is not much shorter than the explicit version that we were able to give before. As concepts become more complicated, it will become more convenient to use new vocabulary and notation that we introduce with definitions.

Now we define a new notation in the form of a new operation, namely the square of an integer.

Definition 1.3.5. Let a be an integer. We let $a^2 := a \cdot a$. We call a^2 the square of a and read a^2 as “ a squared.”

Example 1.3.6. We give examples for squares.

- (i) $5^2 = 5 \cdot 5 = 25$
- (ii) $(-11)^2 = (-11) \cdot (-11) = 121$ (remember a negative integer times a negative integer is a positive integer)
- (iii) $-(11^2) = -(11 \cdot 11) = -121$ (here the parentheses force us to square first and then negate)
- (iv) $(2^2)^2 = 4^2 = 16$

With variables we can define the product of a natural number and an integer using repeated addition as in Example 1.2.7. We introduce the objects under consideration, namely a natural number and an integer, and assign variable names. Then we use the variable names in the statement of the definition.

Definition 1.3.7. Let n be a natural number, and let a be an integer. We define the product of n and a as

$$n \cdot a := \underbrace{a + a + \dots + a}_{n \text{ times}}.$$

We illustrate this definition with an example.

Example 1.3.8. We have $5 \cdot 7 = 7 + 7 + 7 + 7 + 7 = 35$.

1.3.4 “For All” Statements

Statements are often applied to all possible mathematical objects of a specified type. When we use a variable without assigning it a concrete value, as in Definition 1.3.7, we specify what type of object we want the variable to be. “Let a be an integer” means that the variable a is an integer, and can be any integer. We also use the formulation “*for all*” when then give conditions on the properties of the objects.

Problem 1.3.9. *Decide whether the following statement is true or false. Say why.*

For all natural numbers n , we have $n > 0$.

Solution. The statement is true as the natural numbers are $1, 2, 3, 4, \dots$, which are all greater than 0.

It is not always this easy to decide whether a “for all” statement is true or false, as the statement often is claimed to be true for infinitely many numbers. We know that a “for all” statement is false when we have found one value for which the statement is wrong. This makes it easier to prove that a statement is false. Values for which a “for all” statement is false are called a *counterexample*.

Problem 1.3.10. *Decide whether the following statement is true or false. Say why.*

For all integers a , we have $a > 0$.

Solution. The statement is false, since -2 is an integer and $-2 > 0$ is false.

We only need to give one counterexample to show the statement is false even though a false “for all” statement may have many possible counterexamples.

We can also use the “for all” formulation for several variables. We formulate the *commutative property* of the addition of integers with “for all.” The statement is true for any choice of integer for the two variables.

Example 1.3.11. For all integers a and all integers b we have $a + b = b + a$. This is called the *commutative property of addition*.

We formulate the distributive property for integers with “for all.” The statement is true for any choice of integer for the three variables.

Example 1.3.12. For all integers a , b , and c we have $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$. This statement is called the *distributive property* for the addition and multiplication of integers.

Finally we combine the two previous examples and other properties of addition and multiplication.

Example 1.3.13. For all integers a , b , and c , we have:

- (i) $a + (b + c) = (a + b) + c$ (*associative property of addition*)
- (ii) $a + b = b + a$ (*commutative property of addition*)
- (iii) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ (*associative property of multiplication*)
- (iv) $a \cdot b = b \cdot a$ (*commutative property of multiplication*)
- (v) $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$ (*distributive property*)

1.3.5 For all, given any, and let

Instead of using “for all a ____” we sometimes choose a different approach for formulating statements. We write “*given any* a ____” or more commonly “let a be ____” followed by what type of object a is and some other statement or property. The statement that follows applies to any objects of the specified type.

Example 1.3.14. The following four statements all say the the same things. The first statement is the statement from Example 1.3.9.

- (i) For all natural numbers a we have $a > 0$.
- (ii) Given any natural number a we have $a > 0$.
- (iii) Let a be a natural number, then $a > 0$.
- (iv) If a is a natural number, then $a > 0$.

Example 1.3.15 (compare Example 1.3.11). Let a be an integer and let b be an integer. Then $a + b = b + a$. We call this the commutative property of addition.

Example 1.3.16 (compare Example 1.3.12). Let a be an integer, let b be an integer, and let c be an integer. Then $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$. We call this the distributive property.

1.3.6 There Exists

Many statements assert that there is a number with a certain property. In this case we use the formulation “*there exists*.”

Example 1.3.17. Consider the statement:

There exists an integer b such that $b + 2 = 0$.

The statement is true, because if $b = -2$ we have $(-2) + 2 = 0$, and -2 is an integer.

Example 1.3.18. Consider the statement:

There exists a natural number b such that $b + 2 = 0$.

The statement is false, because -2 is the only number with this property, and -2 is not a natural number.

We can also use “there exists” in definitions:

Definition 1.3.19. Let a be an integer. If there exists an integer b such that $a + b = 0$ then b is called the *additive inverse* of a .

1.3.7 Combining “For all” and “There Exists”

Combining the formulations “for all” and “there exists” allows us to formulate more complicated statements.

Since for all integers a we have $a + (-a) = 0$, the number $-a$ is the additive inverse of a . We formulate this as a theorem.

In our formulation of this result as a theorem, we combine the formulations “for all” and “there exists.”

Theorem 1.3.20. *For all integers there exists an additive inverse.*

Now we formulate a statement where the “there exists” comes before the “for all.”

Theorem 1.3.21. *There exists an integer a such that for all natural numbers n we have $a < n$.*

This theorem is easy to prove. When we set $a := -2$ the statement $a < n$ is clearly true for all natural numbers.

1.3.8 Evaluation

So far our use of variables has been in the formulation of statements. We now give a more hands-on use of them. When evaluating an expression we replace the variables by the values given for them and then compute.

Problem 1.3.22. *Evaluate $2 \cdot (b + 3)$ for $b := 7$.*

Solution. Replacing b by 7 we get $2 \cdot (7 + 3) = 2 \cdot 10 = 20$. Thus $2 \cdot (b + 3)$ for $b := 7$ is 20.

Problem 1.3.23. *Decide whether $a \cdot (-2) > 4$ is true for $a := 7$*

Solution. Replacing a by 7 the left hand side of the inequality becomes $7 \cdot (-2) = -14$. As $-14 > 4$ is false, the statement $a \cdot (-2) > 4$ is false for $a := 7$.

1.4 Exponentiation

In Definition 1.3.7, we introduced the concept of multiplication as repeated addition, and we build upon that idea here. We define *exponentiation* as repeated multiplication:

Definition 1.4.1. Let b be an integer and n be a positive integer. We define

$$b^n := \underbrace{b \cdot b \cdot \dots \cdot b}_{n \text{ copies of } b}.$$

We read b^n as “ b to the n -th power” or as “ b to the n -th.” We call b the *base* of b^n and n the *exponent* of b^n . If $n = 2$, then we usually say “ b squared,” instead of “ b to the 2nd,” (also see Definition 1.3.5) and if $n = 3$, we say “ b cubed” instead of “ b to the 3rd.”

First note that, by definition, $b^1 = b$ for all integers b .

Example 1.4.2. For examples of exponentiation we show how they are read.

- (i) $3^2 = 9$ is read “3 squared is equal to 9” or “3 to the 2nd is equal to 9”
- (ii) $2^3 = 8$ is read “2 cubed is equal to 8” or “2 to the third is equal to 8”
- (iii) $2^4 = 16$ is read “2 to the 4th is equal to 16”

Example 1.4.3. For examples of exponentiation we identify the base and exponent.

- (i) In 3^2 the base is 3 and the exponent is 2.
- (ii) In 2^3 the base is 2 and the exponent is 3.
- (iii) In 2^4 the base is 2 and the exponent is 4.

Example 1.4.4. We compute power using the definition.

- (i) $2^2 = 2 \cdot 2 = 4$
- (ii) $2^3 = 2 \cdot 2 \cdot 2 = 8$
- (iii) $2^4 = 2 \cdot 2 \cdot 2 \cdot 2 = 16$
- (iv) $3^2 = 3 \cdot 3 = 9$
- (v) $3^3 = 3 \cdot 3 \cdot 3 = 27$
- (vi) $(-2)^3 = (-2) \cdot (-2) \cdot (-2) = -8$
- (vii) $(-2)^4 = (-2) \cdot (-2) \cdot (-2) \cdot (-2) = 16$

Now, we provide properties of exponents and prove them using the idea that exponentiation is repeated multiplication.

Theorem 1.4.5. *Let a and b be integers, and let m and n be positive integers. Then, the following properties of exponents hold:*

- (i) $(b^m) \cdot (b^n) = b^{(m+n)}$
- (ii) $(b^m)^n = b^{(m \cdot n)}$

Instead of explicitly giving the order of operations with parentheses as in in (i) and (ii) we write b^{m+n} instead of $b^{(m+n)}$ and $b^{m \cdot n}$ instead of $b^{(m \cdot n)}$.

Proof. For integers a and b and positive integers m and n , we have the following:

$$\begin{aligned}
 \text{(i)} \quad (b^m) \cdot (b^n) &= \underbrace{b \cdot b \cdot \dots \cdot b}_m \cdot \underbrace{b \cdot b \cdot \dots \cdot b}_n = \underbrace{b \cdot b \cdot \dots \cdot b}_{m+n} = b^{m+n} \\
 \text{(ii)} \quad (b^m)^n &= (\underbrace{b \cdot b \cdot \dots \cdot b}_m)^n = \underbrace{\underbrace{b \cdot b \cdot \dots \cdot b}_m \cdot \dots \cdot \underbrace{b \cdot b \cdot \dots \cdot b}_m}_n = \underbrace{b \cdot b \cdot \dots \cdot b}_{m \cdot n} = b^{m \cdot n}
 \end{aligned}$$

□

Example 1.4.6. We illustrate the proof of the properties of exponents with an example.

$$(i) \quad (7^2) \cdot (7^3) = (7 \cdot 7) \cdot (7 \cdot 7 \cdot 7) = 7 \cdot 7 \cdot 7 \cdot 7 \cdot 7 = 7^5$$

$$(ii) \quad (7^2)^3 = (7^2) \cdot (7^2) \cdot (7^2) = (7 \cdot 7) \cdot (7 \cdot 7) \cdot (7 \cdot 7) = 7 \cdot 7 \cdot 7 \cdot 7 \cdot 7 \cdot 7 = 7^6$$

Another property of exponentiation follows from the commutative property of multiplication.

Theorem 1.4.7. *Let a and b be integers, and let n be a positive integer. Then $(a \cdot b)^n = (a^n) \cdot (b^n)$.*

Proof. We have $(a \cdot b)^n = \underbrace{(a \cdot b) \cdot (a \cdot b) \cdot \dots \cdot (a \cdot b)}_{n \text{ times}} = \underbrace{a \cdot a \cdot \dots \cdot a}_{n \text{ times}} \cdot \underbrace{b \cdot b \cdot \dots \cdot b}_{n \text{ times}} = a^n \cdot b^n$,

where the middle equal sign holds by the commutative property of multiplication. $\square$

Example 1.4.8. We have

$$(5 \cdot 7)^3 = (5 \cdot 7) \cdot (5 \cdot 7) \cdot (5 \cdot 7) = 5 \cdot 7 \cdot 5 \cdot 7 \cdot 5 \cdot 7 = 5 \cdot 5 \cdot 5 \cdot 7 \cdot 7 \cdot 7 = 5^3 \cdot 7^3$$

To extend our definition of exponentiation to all non-negative integer exponents, we must determine how to define the 0th power of an integer. We first consider an example.

Example 1.4.9. We try to find out what $(-6)^0$ should be. Our definition of $(-6)^0$ should be consistent with the properties of exponentiation in Theorem 1.4.5. In particular Theorem 1.4.5(ii), which states that for all natural numbers a and c we have

$$(-6)^a \cdot (-6)^c = (-6)^{(a+c)}$$

should also hold for $a = 0$. We want

$$(-6)^0 \cdot (-6)^c = (-6)^{(0+c)}$$

to be true. As for all natural number c we have $0 + c = c$ we get

$$(-6)^{(0+c)} = (-6)^c.$$

So the equality we want to be true can be written as

$$(-6)^0 \cdot (-6)^c = (-6)^c.$$

That is we want $(-6)^0$ multiplied by $(-6)^c$ to be equal to $(-6)^c$. The only number by which we can multiply a (non-zero) number and get the number as a result is 1. So for our equation to be true we must set

$$(-6)^0 := 1.$$

The argument in Example 1.4.9 holds not only for (-6) , but for all integers (except for 0). Let b be an integer. To extend our definition of exponentiation to all non-negative integer exponents, we must determine how to define b^0 . Let n be a positive integer. If we want the property in Theorem 1.4.5 (i) to include the possibility of an exponent of zero, we must have $b^0 \cdot b^n = b^{0+n} = b^n$. If $b \neq 0$, the only choice for b^0 that works is $b^0 = 1$.

When the base is 0, there are multiple possibilities for b^0 that would keep the properties in Theorem 1.4.5 correct. One possibility is defining $0^0 := 1$. As it does not break anything, that it does not build a contradiction into the system of mathematics, and it matches what we have found for non-zero bases, we go with this choice.

Definition 1.4.10. For all integers b we set $b^0 := 1$.

We remark that some authors leave 0^0 undefined, while with our definition we have $0^0 = 1$.

Problem 1.4.11. Use the properties of exponentiation to simplify $1256^3 \cdot 1256^{11}$.

Solution. We apply Theorem 1.4.5(i) which states that for all integers b and for all non-negative integers m and n we have $b^m \cdot b^n = b^{m+n}$. With $b = 1256$ and $m = 3$ and $n = 11$ we get

$$1256^3 \cdot 1256^{11} = 1256^{3+11} = 1256^{14}$$

Problem 1.4.12. Let d be an integer. Use the properties of exponentiation to simplify $d^9 \cdot d^7 \cdot d^3$.

Solution. Theorem 1.4.5(i) which states that for non-negative integers m and n we have $d^m \cdot d^n = d^{m+n}$. With $m = 9$ and $n = 7$ we get

$$d^9 \cdot d^7 \cdot d^3 = d^{9+7} \cdot d^3 = d^{16} \cdot d^3.$$

Applying the theorem again (this time with $m = 16$ and $n = 3$) we obtain

$$d^{16} \cdot d^3 = d^{16+3} = d^{19}.$$

We have found

$$d^9 \cdot d^7 \cdot d^3 = d^{9+7} \cdot d^3 = d^{16} \cdot d^3 = d^{16+3} = d^{19}.$$

Thus $d^9 \cdot d^7 \cdot d^3$ simplifies to d^{19} .

Problem 1.4.13. Let d be an integer. Use the properties of exponentiation to simplify $(d^3)^5$.

Solution. Apply Theorem 1.4.5(ii) we get

$$(d^3)^5 = d^{3 \cdot 5} = d^{15}$$

We end our discussion of exponentiation with a table of powers (Figure 1.4.1).

1.4.1 Square Roots

Definition 1.4.14. Let b be a non-negative integer. By the *square root* of b , written as $\sqrt{b}$, we mean the non-negative number a such that $a^2 = b$.

Some, but not all, square roots are integers. If the square root of b is an integer, we call b a *perfect square*.

Example 1.4.15. Some examples of perfect squares are $1 = 1^2$, $4 = 2^2$, $9 = 3^2$, and $16 = 4^2$. Their square roots are integers: $\sqrt{1} = 1$, $\sqrt{4} = 2$, $\sqrt{9} = 3$, and $\sqrt{16} = 4$.

If a number is given in a convenient form, it is easy to find its square root.

Figure 1.4.1: Powers of integers. The rows contain the base b for $0 \leq b \leq 10$ and the columns contain the exponent n for $0 \leq n \leq 9$.

b^n	0	1	2	3	4	5	6	7	8	9
0	1	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1
2	1	2	4	8	16	32	64	128	256	512
3	1	3	9	27	81	243	729	2 187	6 561	19 683
4	1	4	16	64	256	1 024	4 096	16 384	65 536	262 144
5	1	5	25	125	625	3 125	15 625	78 125	390 625	1 953 125
6	1	6	36	216	1 296	7 776	46 656	279 936	1 679 616	10 077 696
7	1	7	49	343	2 401	16 807	117 649	823 543	5 764 801	40 353 607
8	1	8	64	512	4 096	32 768	262 144	2 097 152	16 777 216	13 421 7728
9	1	9	81	729	6 561	59 049	531 441	4 782 969	43 046 721	387 420 489
10	1	10	100	1 000	10 000	100 000	1 000 000	10 000 000	100 000 000	1 000 000 000

Example 1.4.16. We give some more square roots of perfect squares.

- (i) $\sqrt{25} = \sqrt{5^2} = 5$.
- (ii) $\sqrt{144} = \sqrt{12^2} = 12$.
- (iii) $\sqrt{169} = \sqrt{13^2} = 13$
- (iv) $\sqrt{24372634816267643286^2} = 24372634816267643286$

When an integer is given as a square it is always easy to find its square root.

Problem 1.4.17. What is $\sqrt{77^2}$?

Solution. The square root of 77^2 is 77.

Problem 1.4.18. What is $\sqrt{667848628784687^2}$?

Solution. The square root of 667848628784687^2 is 667848628784687.

Chapter 2

Algorithms

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Recognize what simple algorithms return.
- (2) Compute the output values of an algorithm for given input values.

Algorithms are important to mathematics and other sciences. They give a structured way to explain a process in detail. Algorithms consist of instructions that can be executed by human beings or machines. Machines, usually computers, follow instructions they are given in a (limited) language called a programming language.

In this course we will not program computers but will nevertheless introduce a handful of instructions that we use to formulate our algorithms. These instructions will clarify how the algorithms should be executed.

2.1 Definition of an Algorithm

We give a formal definition of an algorithm, introduce the instructions that we will use, and end with an algorithm for computing powers of integers. Throughout this section we will give examples of algorithms.

Definition 2.1.1. An *algorithm* is a finite sequence of instructions for performing a task.

By finite we mean that there is an end to the sequence of instructions. The joke in Figure 2.5.1 refers to a misunderstood sequence of instructions with no end.

A recipe is a real-life example of an algorithm. The pancake recipe below is in the same format that we use to present algorithms. It already has some of the key features. Our algorithms always have an input, which contains all the ingredients needed to perform the task. In the pancake example we assume that kitchen hardware such as bowls and spoons are available, and we do not list them as input. The output is what the algorithm produces, or returns, when all instructions have been followed. We list the product of the algorithm after the instruction **return**. The pancake algorithm also contains a loop. You are instructed to repeat frying pancakes until there is no batter left.

Algorithm 2.1.2 (*Eierkuchen*).

Input: 1 cup of milk, 2 tablespoons of sugar, 1 cup of flour, 3 large eggs, 1 pinch of salt, and oil for the pan

Output: a stack of pancakes on a plate

- (1) Mix flour, eggs, sugar, and salt with an egg beater until the mixture is homogeneous.
- (2) Slowly add the milk while stirring the mixture.
- (3) Cool the batter in the refrigerator for an hour.
- (4) Heat a pan with oil.
- (5) **repeat** the following steps:
 - (a) Take a ladle of batter and pour into the pan.
 - (b) Fry pancake on one side.
 - (c) Flip pancake.
 - (d) Fry pancake on other side.
 - (e) Take pancake out of the pan.
 - (f) Put pancake onto plate.
- (6) **until** bowl is empty.
- (7) **return** plate with pancakes.

Although algorithms can consist of any kind of instructions (as illustrated in the recipe above), the algorithms in this course will be limited to computations with integers. Our formulation of algorithms consist of four parts:

- The word **Algorithm** is usually followed by a name that we can use to refer to the algorithm.
- The *Input* specifies what kind of numbers can be given to the algorithm. We often give the properties we expect them to have for the algorithm to work. We also include the variable names that are used to refer to these numbers.
- The *Output* tells us **what** the algorithm does by stating the properties of the numbers that the output of the algorithm.
- A **sequence of instructions** that yields the output. This sequence of instructions describes **how** the output is generated. We number the instructions and follow them in their numerical order.

In the pancake recipe under input we give the amounts of the ingredients and the size of the eggs. We formulate the input values as variables, so that we can refer to them in the algorithm. We also specify the output values and their properties.

For very simple algorithms declaring the output and giving the sequence of instructions will seem redundant. For slightly more complicated algorithms this is not the case. You will also see that there are different sequences of instructions that yield the same output.

In the formulation of algorithms we use the instructions **let**, **if**___**then**, **repeat**___**until**, and **return**, which we explain in detail in the following. In addition we use standard mathematical notation for computations. Each instruction in an algorithm consists of commands and mathematical expressions. We number the instructions in the algorithms and follow the instructions in this order. Each numbered instruction is called a *step*. The **repeat**___**until** loop can be used to change this order by telling us to repeat previous instructions. The conditional **if**___**then** also changes which commands are to be followed, depending on whether a statement is true or false.

Before we follow any specific instruction we always evaluate any mathematical expression first.

2.2 The Instruction return

When executing an algorithm, we follow the instructions in the algorithm until we encounter the instruction **return**. At **return** we leave the algorithm, and the values after the return statement are the output of the algorithm. The properties of these values must match what is given under *Output*. What the algorithm returns is the output of the algorithm. If an algorithm returns several values, we specify this by separating the output values by commas (see Algorithm 2.2.3 below). We give some examples of algorithms.

In these simple examples *Output* and the expression after **return** are essentially the same, which makes one of them seem redundant. In later examples it will become clearer that under *Output* we say **what** we are computing and that the sequence of instructions say **how** we compute it.

This algorithm has two integers as its input and returns their sum as its output:

Algorithm 2.2.1 (*Sum of two integers*).

Input: two integers a and b

Output: the sum of a and b

(1) **return** $a + b$

Example 2.2.2. We follow the instructions in Algorithm 2.2.1 for the input values $a := 57$ and $b := 8$.

Input: $a := 57$ and $b := 8$. As both 57 and 8 are integers this is a valid input for the algorithm.

(1) **return** $a + b$: The value of the variable a is 57 and the value of the variable b is 8. We compute $a + b = 57 + 8 = 65$. So the value after **return** is 65, it is the output of the algorithm.

Output: 65

So when the input is $a := 57$ and $b := 8$ the output of Algorithm 2.2.1 is 65.

The following algorithm has four output values. The input of the algorithm is an integer c ; its output are the powers c , c^2 , c^3 , and c^4 .

Algorithm 2.2.3 (*Four powers*).

Input: an integer c

Output: c , c^2 , c^3 , c^4

(1) **return** $c, c \cdot c, c \cdot c \cdot c, c \cdot c \cdot c \cdot c$

Example 2.2.4. We follow the instructions in Algorithm 2.2.3 for the input value $c := -3$.

Input: $c := -3$. As -3 is an integer this is a valid input for the algorithm.

(1) **return** $c, c \cdot c, c \cdot c \cdot c, c \cdot c \cdot c \cdot c$: The value of the variable c is -3 . We compute $c \cdot c = (-3) \cdot (-3) = 9$, $c \cdot c \cdot c = (-3) \cdot (-3) \cdot (-3) = -27$, and $c \cdot c \cdot c \cdot c = (-3) \cdot (-3) \cdot (-3) \cdot (-3) = 81$. Thus the algorithm returns the values -3 , 9 , -27 , and 81 .

Output: $-3, 9, -27, 81$.

So when the input is $c := -3$ the output of Algorithm 2.2.3 is $-3, 9, -27$, and 81 .

This algorithm does not have any input and always returns 42 as its output:

Algorithm 2.2.5 (*Fortytwo*).

Input: None

Output: the integer 42

(1) **return** 42

An algorithm without a **return** instruction does not have any output.

2.3 The Conditional **if** **then**

Conditionals are used to specify which instruction should be followed depending whether a statement is true or false.

The conditional has the two parts **if** and **then**. If the statement after **if** is true, the instruction that follows **then** is executed. If the statement after **if** is false, we do not execute the instruction that follows **then**, and instead we continue with the next instruction.

Consider the algorithm:

Algorithm 2.3.1 (*Maximum of two integers*).

Input: two integers a and b

Output: the maximum of a and b

(1) **if** $a \geq b$ **then return** a

(2) **return** b

Example 2.3.2. We follow Algorithm 2.3.1 for the input values $a := 2$ and $b := 5$. For each (numbered) line in the sequence of instructions we describe what we do.

Input: $a := 2$ and $b := 5$

(1) **if** $a \geq b$ **then return** a : We check whether the value of a is greater than the value of b . If this is true we follow the instruction after **then**, otherwise we continue with the next step. The value of a is 2 and the value of b is 5 . Because $2 \geq 5$ is false, we do not follow the instruction after **then** and continue with step (2).

(2) **return** b – The algorithm returns the value of the variable b which is 5 .

Output: 5

So when the input is $a = 2$ and $b = 5$ the output of Algorithm 2.3.1 is 5 .

Example 2.3.3. We follow Algorithm 2.3.1 for the input $a := 3$ and $b := 1$.

Input: $a := 3$ and $b := 1$

- (1) **if** $a \geq b$ **then return** a : We have $a = 3$ and $b = 1$. This $a \geq b$ is true. We follow the instruction after **then** and return 3.

Output: 3

So when the input is $a = 3$ and $b = 1$ the output of Algorithm 2.3.1 is 3.

Example 2.3.4. We follow Algorithm 2.3.1 for the input $a := 11$ and $b := 11$.

Input: $a := 11$ and $b := 11$

- (1) Because $11 \geq 11$ is true. Thus we follow the instruction after **then** and return 3.

Output: 11

So when the input is $a = 11$ and $b = 11$ the output of Algorithm 2.3.1 is 11.

Absolute Value

Our next goal is the formulation of an algorithm that returns the absolute value of an integer. We start with a definition of the absolute value of an integer.

Definition 2.3.5. The *absolute value* of an integer b is its distance from zero. We denote the absolute value of an integer a by $|a|$.

A distance between two integers is always a non-negative integer. So the absolute value of an integer is a non-negative integer.

Example 2.3.6. We give examples of absolute values.

- (i) The absolute value of 2 is 2, as the distance of 2 from 0 (on the number line) is 2. We write $|2| = 2$.
- (ii) The absolute value of -2 is 2, as the distance of -2 from 0 (on the number line) is 2. We write $|-2| = 2$.
- (iii) The absolute value of 0 is 0, as the distance of 0 from 0 (on the number line) is 0. We write $|0| = 0$.

If an integer b is positive, then its absolute value is the integer b itself. When b is negative, its distance from 0 still is positive. We can casually describe finding the absolute value of b as removing the negative sign. This is easy to describe in words, but not easy in mathematical notation. To write this using mathematical operations, we say that if b is negative, the absolute value of b is $-b$, which is positive.

Example 2.3.7. We compute the absolute values of some integers.

- (i) $|0| = 0$
- (ii) $|8| = 8$
- (iii) $|-10| = -(-10) = 10$

We are now ready to formulate an algorithms that returns the absolute value of an integer.

Algorithm 2.3.8 (*Absolute value*).

Input: an integer b

Output: the absolute value of b

- (1) **if** $b \geq 0$ **then return** b
- (2) **return** $-b$

Example 2.3.9. We follow the instructions in Algorithm 2.3.8 for the input $b := 7$.

Input: $b := 7$

- (1) **if** $b \geq 0$ **then return** b : As $b = 7$ the statement $b \geq 0$ is true. Hence we follow the instruction after **then** and return the value of b which is 7.

Output: 7

So when the input is $b := 7$ the output of Algorithm 2.3.8 is 7.

Example 2.3.10. We follow the instructions in Algorithm 2.3.8 for the input $b := -6$.

Input: $b := -6$

- (1) **if** $b \geq 0$ **then return** b : As $b = -6$ the statement $-6 \geq 0$ is false. Hence we continue with the next instruction.
- (2) **return** $-b$ We compute $-b$ which is $-(-6) = 6$ and return 6.

Output: 6

So when the input is $b := -6$ the output of Algorithm 2.3.8 is 6.

2.4 The Assignment **let** $_{_} :=$

The instruction **let** $a := b$ assigns the value of b to a . For example after the instruction **let** $a := 5$, the variable a has the value 5. Assume that the variable a has the value 5; then after **let** $a := a + 1$, the variable a has the value 6. This occurs because we first evaluate that $a + 1$ is 6 (since $a = 5$) and then assign 6 to the variable a .

Example 2.4.1. We give examples for the use of the instruction **let** $_{_} :=$.

- (i) The easiest use of **let** is to assign a value to a variable, for example we can use it to assign the value 0 to i .

let $i := 0$

After this instruction the value of the variable i is 0.

- (ii) We can also use **let** to assign the value of an expression to a variable. Assume that the value of the variable c is 10. In the instruction

let $d := c + 35$

first the expression on the right is evaluated. As $c = 10$ we get that $c + 35 = 10 + 35 = 45$. Then the computed value 45 is assigned to d , so that the value of d after this instruction is 45.

- (iii) It is more confusing when the same variable shows up on both sides of the $:=$ of a **let** instruction. Assume that the value of i is 5 and consider the instruction:

let $i := i + 1$

First the expression on the right is evaluated, we obtain $i + 1 = 5 + 1 = 6$. Then the result of this computation, namely 6, is assigned to i , so that the new value of i is 6.

The next algorithm computes the same values as Algorithm 2.2.3, namely c , c^2 , c^3 , and c^4 .

Algorithm 2.4.2 (*Four powers fast*).

Input: an integer c

Output: c, c^2, c^3, c^4

- (1) **let** $d := c \cdot c$
- (2) **let** $e := c \cdot d$
- (3) **let** $f := d \cdot d$
- (4) **return** c, d, e, f

The *Input* and *Output* of Algorithm 2.2.3 and Algorithm 2.4.2 are the same, but the results are obtained in different ways. In particular the computation effort differs.

- Algorithm 2.2.3 needs one multiplication to compute $c^2 = c \cdot c$, two multiplications to compute $c^3 = c \cdot c \cdot c$, and three multiplications to compute $c^4 = c \cdot c \cdot c \cdot c$ (count the multiplication symbols ‘ $\cdot$ ’). This is a total of 6 multiplications.
- Algorithm 2.4.2 computes the same values as $c^2 = d = c \cdot c$, $c^3 = e = c \cdot d$, and $c^4 = f = d \cdot d$. It only needs 3 multiplications to compute all four values.

So we can consider Algorithm 2.4.2 to be about twice as fast as Algorithm 2.2.3.

Example 2.4.3. We follow the steps of the algorithm for the input $c := -3$

Input $c := -3$

- (1) **let** $d := c \cdot c$: We have $c = -3$. We compute $c \cdot c = (-3) \cdot (-3) = 9$ and assign this value to the variable d . Now $d = 9$.
- (2) **let** $e := c \cdot d$: We have $c = -3$ and $d = 9$. We compute $c \cdot d = (-3) \cdot 9 = -27$ and assign the value to the variable e . Now $e = -27$.
- (3) **let** $f := d \cdot d$: We have $d = 9$. We compute $d \cdot d = 9 \cdot 9 = 81$ and assign this value to the variable f .
- (4) **return** c, d, e, f : We return the values of c, d, e , and f , namely $-3, 9, -27$, and 81

Output $-3, 9, -27, 81$

So when the input to Algorithm 2.4.2 is $c := -3$ then the output is $-3, 9, -27, 81$.

2.5 The Loop **repeat_until**

Loops allow us to repeat sequences of instructions. In a **repeat__until**-loop a sequence of instructions is repeatedly followed until a specified statement is true.

A **repeat__until**-loop starts with a **repeat** instruction and ends with a **until** instruction. The instructions between **repeat** and **until** are followed (in order) until the statement after **until** is true. If you follow the algorithm and get to the instruction **until** and the statement after **until** is false, you jump back to **repeat** and execute the first instruction after **repeat**.

In our next example, given a natural number n , computes the sum of the first n natural numbers. In the algorithm, s is the sum so far and i is incremented in each iteration of the **repeat__until** loop. The algorithm computes $1 + 2 + 3 + \cdots + (n - 2) + (n - 1) + n$.

Our first algorithm with a **repeat__until**-loop subtracts 2 from a given natural number n until we get number that is less than 2. When the number is even the output is 0, when the number the output is 1.

Algorithm 2.5.1 (*Even or odd*).

Input: a natural number n greater than 1

Output: 0 if n is even, 1 otherwise

- (1) **repeat**
 - (a) **let** $n := n - 2$
- (2) **until** $n < 2$
- (3) **return** n

Example 2.5.2. We follow the instructions of Algorithm 2.5.1 for the input $n := 5$. *Input:* $n := 5$

- (1) **repeat** : A **repeat__until**-loop starts here
 - (a) **let** $n := n - 2$: We have $n = 5$. We compute $n - 2 = 5 - 2 = 3$ and assign this value to n . So now $n = 3$.
- (2) **until** $n < 2$: We have $n = 3$. Since the statement $n < 2$ is false we repeat the instructions in the loop by continuing with step (1)(a).
 - (a) **let** $n := n - 2$: We have $n = 3$. We compute $n - 2 = 3 - 2 = 1$ and assign this value to n . So now $n = 1$.
- (2) **until** $n < 2$: We have $n = 1$. Since the statement $n < 2$ is true we leave the loop and continue with step (3).
- (3) **return** n : We return the value of n which is 1.

Output: $n = 1$

The following algorithm returns the sum of the first n natural numbers.

Algorithm 2.5.3 (*Sum up to*).

Input: a natural number n

Output: the sum of the first n natural numbers

- (1) **let** $i := 0$
- (2) **let** $s := 0$
- (3) **repeat**
 - (a) **let** $i := i + 1$

- (b) **let** $s := s + i$
- (4) **until** $i = n$
- (5) **return** s

Example 2.5.4. We follow the steps of Algorithm 2.5.3 for the input $n := 4$. For each step of the algorithm, we give the new values of the variables that change values in that step.

Input: $n := 4$

- (1) $i = 0$
- (2) $s = 0$
- (3) (a) $i = 0 + 1 = 1$
- (b) $s = 0 + 1 = 1$
- (4) As $i = 1$ and $n = 4$ the statement $i = n$ is false. We repeat the loop and continue with step (3)(a).
- (3) (a) $i = 1 + 1 = 2$
- (b) $s = 1 + 2 = 3$
- (4) As $i = 2$ and $n = 4$ the statement $i = n$ is false. We repeat the loop and continue with step (3)(a).
- (3) (a) $i = 2 + 1 = 3$
- (b) $s = 3 + 3 = 6$
- (4) As $i = 3$ and $n = 4$ the statement $i = n$ is false. We repeat the loop continue with step (3)(a).
- (3) (a) $i = 3 + 1 = 4$
- (b) $s = 6 + 4 = 10$
- (4) As $i = 4$ and $n = 4$ the statement $i = n$ is true. We continue with step (5).
- (5) The algorithm returns $s = 10$.

Output: 10

Figure 2.5.1: A repeat joke

Question: Why did the programmer go into the shower and never come out ?

Answer: He read the instructions on the shampoo bottle: Massage into wet hair. Lather. Rinse. Repeat.

A **repeat__until**-loop with a statement that is always false in the **repeat** instruction yields a never ending loop. A sequence of instructions (even if this is only the case for certain input values) that contains such a loop is not an algorithm.

Example 2.5.5. We give an example of a sequence of instructions that gets caught in a never ending loop for certain input values.

Input: an integer a

- (1) **let** $m := 1$
- (2) **repeat**
 - (a) **let** $m := m \cdot a$
 - (b) **let** $a := a - 1$
- (3) **until** $a = 0$
- (4) **return** m

When the input a is 0 or a negative integer, this sequence of commands does not end. It never reaches the instruction **return** m since a will never be 0. In this case we do not have a finite sequence of instructions, so it does not match the definition of an algorithm (Definition 2.1.1).

Problem 2.5.6. *With the algorithm below answer the following questions:*

- (i) *Follow the steps of the algorithm for $b := 5$ and $n := 3$.*
- (ii) *What does the algorithm return for the input $b := -2$ and $n := 6$.*
- (iii) *What does the algorithm compute?*

Algorithm

Input : An integer b and a natural number n

- (1) **let** $c := 0$
- (2) **let** $i := 0$
- (3) **repeat**
 - (a) **let** $c := c + b$
 - (b) **let** $i := i + 1$
- (4) **until** $i = n$
- (5) **return** c

Solution. (i) We follow the algorithm for the input $b := 5$ and $n := 3$.

Input: $b := 5$ and $n := 3$

- (1) $c = 0$
- (2) $i = 0$
- (3) (a) $c = 5$
(b) $i = 1$
- (4) As $i = 1$ and $n = 3$ the statement $i = n$ is false. We repeat the loop and continue with step (3)(a).
- (3) (a) $c = 10$
(b) $i = 2$
- (4) As $i = 2$ and $n = 3$ the statement $i = n$ is false. We repeat the loop and continue with step (3)(a).
- (3) (a) $c = 15$
(b) $i = 3$
- (4) As $i = 3$ and $n = 3$ the statement $i = n$ is true. We exit the loop and continue with step (5).
- (5) We return the value of c which is 15.

Output: 15

- (ii) Proceeding as above we find that for the input $b = -2$ and $n = 6$ the algorithm returns

−12.

- (iii) Until $i < n$ the algorithm adds b to c and adds 1 to n . Thus the number of times b is added to c is n . As initially i is set to 0 (see step (2)) the number of times b is added to 0 is n . So the output of the algorithm is $n \cdot b$. Also compare Definition 1.3.7 where we had defined multiplication as repeated addition.

Problem 2.5.7. *With the algorithm below answer the following questions:*

- (i) *Follow the steps of the algorithm for the input $n := 3$.*
(ii) *What does the algorithm compute?*

Algorithm

Input : A natural number n

- (1) **let** $f := 1$
(2) **repeat**
 (a) **let** $f := f \cdot n$
 (b) **let** $n := n - 1$
(3) **until** $n = 0$
(4) **return** f

Solution. (i) We follow the algorithm for the input $n := 3$.

Input: $n := 3$.

- (1) $f := 1$
(2) (a) $f := 1 \cdot 3 = 3$
 (b) $n := 3 - 1 = 2$
(3) As $n = 2$ the statement $n = 0$ is false. To repeat the loop we continue with step (2)(a).
(2) (a) $f := 3 \cdot 2 = 6$
 (b) $n := 2 - 1 = 1$
(3) As $n = 1$ the statement $n = 0$ is false. To repeat the loop we continue with step (2)(a).
(2) (a) $f := 6 \cdot 1 = 6$
 (b) $n := 1 - 1 = 0$
(3) As $n = 0$ the statement $n = 0$ is true. We exit the loop and continue with step (4).
(4) We return the value of f which is 6.

Output: 6

- (ii) For the input value n , the output of the algorithm is $n \cdot (n - 1) \cdot \dots \cdot 2 \cdot 1$. So, the algorithm computes the product of the first n natural numbers.

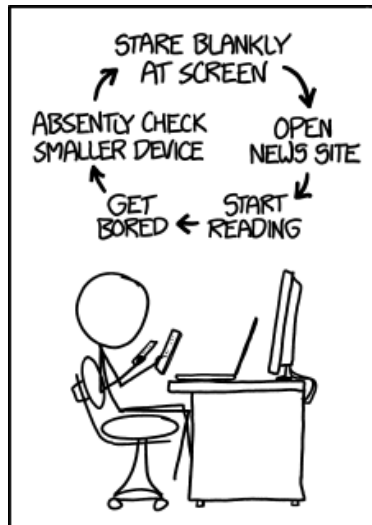
The product of the first n natural numbers is important enough to have a name.

Definition 2.5.8. Let n be a natural number. The product

$$1 \cdot 2 \cdot 3 \cdot \dots \cdot (n - 1) \cdot n$$

is called *n factorial*. We denote *n factorial* by $n!$.

Figure 2.5.2: *Loop* by R. Munroe (<https://xkcd.com/1411>).



Ugh, today's kids are forgetting the old-fashioned art of absentmindedly reading the same half-page of a book over and over and then letting your attention wander and picking up another book.

2.6 Exponentiation Algorithm

We present an algorithm for computing a power of an integer. We call this algorithm the *Naive Exponentiation* algorithm, since there is a more clever way of calculating powers which we will present with Algorithm 15.3.6.

Algorithm 2.6.1 (*Naive Exponentiation*).

Input: An integer b and a non-negative integer n

Output: b^n

- (1) **if** $n = 0$ **then return** 1
- (2) **let** $c := 1$
- (3) **let** $i := 0$
- (4) **repeat**
 - (a) **let** $c := c \cdot b$
 - (b) **let** $i := i + 1$
- (5) **until** $i = n$
- (6) **return** c

In this algorithm, the number of steps in the sequence of computations, n , is directly given as one of the inputs. We demonstrate this algorithm with a numerical example.

Example 2.6.2. We compute 5^3 with Algorithm 2.6.1.

Input: $b = 5$ and $n = 3$

- (1) As $3 \neq 0$ we continue with step (2)

- (2) $c := 1$
- (3) $i := 0$
- (4) (a) $c := 1 \cdot 5 = 5$
 (b) $i := 0 + 1 = 1$
- (5) As $i = 1$ and $n = 3$ the statement $i = n$ is false. We continue with step (4)(a).
- (4) (a) $c := 5 \cdot 5 = 25$
 (b) $i := 1 + 1 = 2$
- (5) As $i = 2$ and $n = 3$ the statement $i = n$ is false. We continue with step (4)(a).
- (4) (a) $c := 5 \cdot 25 = 125$
 (b) $i := 2 + 1 = 3$
- (5) As $i = 3$ and $n = 3$ we have $i = n$. We continue with step (6).
- (6) We return the value of c which is 125.

Output: 125

We have computed $5^3 = 125$.

Chapter 3

Division

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Compute quotients and remainders using the division algorithm.
- (2) Recognize the division algorithm and algorithms that compute only the quotients or remainders.
- (3) Compute quotients and remainders of larger numbers using a calculator.
- (4) Apply properties of the operation mod.
- (5) Apply the operation mod in real world problems.
- (6) Apply the operation mod in the validation of ISBN numbers.

Division yields the answer of the question

How often does a natural number b go into another natural number a ?

The answer to this question is called the *quotient* of the division of a by b , which we often denote by q .

3.1 Quotients and Remainders

In this course we restrict our considerations to integers, so we only allow q to also have integer values. One way of determining how often b goes into a is to repeatedly subtract b from a . The quotient (of the division of a by b) is then the number of times we have subtracted b . Whatever is leftover from a after subtracting b as often as possible is called the remainder (of the division of a by b).

Example 3.1.1. To find out how often 7 goes into 26 we repeatedly subtract 7 from 26. We stop before subtracting 7 again would give us a negative number. In other words we stop when we get to a number that is less than seven, because then 7 would not go another time into the number. We get:

$$26 - 7 = 19$$

$$19 - 7 = 12$$

$$12 - 7 = 5$$

We stop here because 7 does not go into 5. We have subtracted 7 three times, which means that 7 goes into 26 three times. We have 5 leftover. This is the remainder. We have found that

$$26 - 3 \cdot 7 = 5 \text{ or } 26 = 3 \cdot 7 + 5.$$

The goal of this section is fully understand what division in the integers means and to give applications.

We start by formalizing the procedure introduced above with the Division Algorithm and generalize it to negative integers. We introduce the operations `div` and `mod` as notation for quotients and remainders and describe how integer division can be performed with a calculator. We investigate properties of the operation `mod`, and give an example for its use in the validation of ISBN numbers.

3.2 Division Algorithm

Division can be thought of as “undoing” multiplication. Since we defined multiplication as repeated addition, we will first perform division through repeated subtraction. In the division algorithm we start with non-negative integer a and keep subtracting a natural number b until we end up with a number that is less than b and greater than or equal to 0. We call the number of times that we can subtract b from a the *quotient* of the division of a by b . The remaining number is called the *remainder* of the division of a by b .

We typically use the variable q for the quotient and the variable r for the remainder.

We have

$$r = a - \underbrace{b - b - \dots - b}_{q \text{ times}} = a - \underbrace{(b + b + \dots + b)}_{q \text{ times}} = a - (b \cdot q).$$

As the division algorithm computes the quotient as well as the remainder, **return** is followed by two values separated by a comma.

If $a < b$ then we cannot subtract b from a and end up with a number greater than or equal to b . Thus, in this case, the quotient is 0 and the remainder is a . We catch this case in step (1) of the algorithm.

Algorithm 3.2.1 (*Division for positive numbers*).

Input: a natural number a and a natural number b

Output: Two integers q and r such that $a = (b \cdot q) + r$ and $0 \leq r < b$

- (1) **if** $a < b$ **then return** 0, a
- (2) **let** $q := 0$
- (3) **let** $r := a$
- (4) **repeat**
 - (a) **let** $r := r - b$
 - (b) **let** $q := q + 1$
- (5) **until** $r < b$
- (6) **return** q, r

Example 3.2.2. We find the output values of Algorithm 3.2.1 for the input values $a := 4$ and $b := 7$.

Input: $a := 4$ and $b := 7$

- (1) As $a = 4$ and $b = 7$ statement $a < b$ is true. So we follow the instruction after **then** and return the values of q and r , namely 0 and 4.

Output: 0,4

Thus the quotient of the division of 4 by 7 is 0 and the remainder is 4.

Example 3.2.3. We find the output values of the Algorithm 3.2.1 for the input values $a := 30$ and $b := 8$.

Input: $a := 30$ and $b := 8$

- (1) As $a = 30$ and $b = 8$ the statement $a < b$ is false. So we continue with step (2).
- (2) $q := 0$
- (3) $r := 30$
- (4) (a) $r := 30 - 8 = 22$
 (b) $q := 0 + 1 = 1$
- (5) As $r = 22$ and $b = 8$ the statement $r < b$ is false. So we continue with step (4)(a).
- (4) (a) $r := 22 - 8 = 14$
 (b) $q := 1 + 1 = 2$
- (5) As $r = 14$ and $b = 8$ the statement $r < b$ is false. So we continue with step (4)(a).
- (4) (a) $r := 14 - 8 = 6$
 (b) $q := 2 + 1 = 3$
- (5) As $r = 6$ and $b = 8$ the statement $r < b$ is true. So we continue with step (6).
- (5) We return the quotient $q = 3$ and the remainder $r = 6$

Output: 3, 6

Thus the quotient of the division of 30 by 8 is 3 and the remainder is 6.

If $a > 0$, then Algorithm 3.2.1 returns the quotient and remainder of the division of a by b . If we try to use Algorithm 3.2.1 when a is negative, the algorithm always returns 0, a which does not satisfy the condition $0 \leq r$ for the output since $r = a < 0$. So we need a different algorithm for the case $a < 0$.

3.2.1 Division of negative numbers

When $a < 0$, we still want find q and r such that $a = (b \cdot q) + r$ with $0 \leq r < b$. We get a positive remainder when a is negative by repeated addition of b . This is the same as repeatedly subtracting $-b$. Let s be the number of times we have to add b to a in order to get $0 \leq r < b$. After s additions of b to a we have

$$r = a + \underbrace{b + b + \dots + b}_{s \text{ times}} = a + (b \cdot s).$$

If we let $q := -s$, we get $r = a - (b \cdot q)$ (compare this to what we wanted). We stop when $0 \leq r < b$. We repeatedly add b to negative numbers until $0 \leq r < b$ is true. Since a negative number plus b is always less than b and we check the value of r after every addition, it is sufficient to check whether $0 \leq r$.

Example 3.2.4. We illustrate the process of dividing a negative number by dividing -33 by 9 . We repeatedly add 9 until we get a number from 0 to $9-1=8$. That number is the remainder. The negative of the number of times we add 9 is the quotient.

$$-33 + 9 = -24$$

$$-24 + 9 = -15$$

$$-15 + 9 = -6$$

$$-6 + 9 = 3$$

As $0 \leq 3 < 9$ we are done. The remainder is 3 . We have added 9 four times, so the quotient is -4 . We have

$$-33 + 9 \cdot 4 + 3 \text{ or } -33 = -(9 \cdot 4) + 3 \text{ or } -33 = 9 \cdot (-4) + 3.$$

We now formalize this procedure in an algorithm.

Algorithm 3.2.5 (*Division for negative numbers*).

Input: A negative integer a and a natural number b

Output: Two integers q and r such that $a = (b \cdot q) + r$ and $0 \leq r < b$

- (1) **let** $q := 0$
- (2) **let** $r := a$
- (3) **repeat**
 - (a) **let** $r := r + b$
 - (b) **let** $q := q - 1$
- (4) **until** $r \geq 0$
- (5) **return** q, r

Example 3.2.6. We find the output values of the Division Algorithm (Algorithm 3.2.5) for the input values $a := -20$ and $b := 7$.

Input: $a := -20$ and $b := 7$

- (1) $q := 0$
- (2) $r := a = -20$
- (3) (a) $r := -20 + 7 = -13$
(b) $q := 0 - 1 = -1$
- (4) As $r = -20$ the statement $r \geq 0$ is false. So we continue with step (3)(a).
- (3) (a) $r := -13 + 7 = -6$
(b) $q := -1 - 1 = -2$
- (4) As $r = -6$ the statement $r \geq 0$ is false. So we continue with step (3)(a).
- (3) (a) $r := -6 + 7 = 1$
(b) $q := -2 - 1 = -3$
- (4) As $r = 1$ the statement $r \geq 0$ is false. So we continue with step (5).
- (5) We return the quotient $q = -3$ and the remainder $r = 1$

Output: $-3, 1$

Thus the quotient or the division of -20 by 7 is -3 and the remainder is 1 .

We give some more examples.

Example 3.2.7. With the values from Examples 3.2.2, 3.2.3, and 3.2.6 we get:

- (i) For $a := 4$ and $b := 7$, we have $q = 0$ and $r = 4$, and write $4 = (7 \cdot 0) + 4$.
- (ii) For $a := 30$ and $b := 8$, we have $q = 3$ and $r = 6$, and write $30 = (8 \cdot 3) + 6$.
- (iii) For $a := -20$ and $b := 7$, we have $q = -3$ and $r = 1$, and write $-20 = (7 \cdot (-3)) + 1$.

Problem 3.2.8. For the given values of a and b , determine the quotient q and remainder r of the division of a by b , and write the equality $a = (b \cdot q) + r$.

- (i) $a := 7$ and $b := 3$
- (ii) $a := 7$ and $b := 8$
- (iii) $a := 20$ and $b := 4$
- (iv) $a := -13$ and $b := 3$

Solution. The answers are provided here, but details for the solution are omitted.

- (i) For $a := 7$ and $b := 3$, we have $q = 2$ and $r = 1$, and write $7 = (3 \cdot 2) + 1$.
- (ii) For $a := 7$ and $b := 8$, we have $q = 0$ and $r = 7$, and write $7 = (8 \cdot 0) + 7$.
- (iii) For $a := 20$ and $b := 4$, we have $q = 5$ and $r = 0$, and write $20 = (4 \cdot 5) + 0$.
- (iv) For $a := -13$ and $b := 3$, we have $q = -5$ and $r = 2$, and write $-13 = (3 \cdot (-5)) + 2$.

Using Algorithm 3.2.1 or Algorithm 3.2.5, we can compute the quotient and remainder of the division of any integer a by any natural number b . For $a := 0$ and any natural number b we have $a = (b \cdot q) + r$ and $0 \leq r < b$ when $q = 0$ and $r = 0$.

Thus for all integers a and all natural numbers b we can find integers q and r such that $a = (b \cdot q) + r$ and $0 \leq r < b$. We call the combination of the two algorithms the *division algorithm*.

The construction of q and r in those algorithms yields a proof of the following theorem.

Theorem 3.2.9. Let a be an integer and b be a natural number. Then, there exist unique integers q and r with $0 \leq r < b$ such that

$$a = (b \cdot q) + r.$$

Next we introduce notation for the the quotient and remainder of the division of an integer by a natural number.

Definition 3.2.10. Let a be an integer and b be a natural number, and let q and r be the unique integers such that $0 \leq r < b$ and $a = (b \cdot q) + r$.

- (i) We denote the *quotient* q of the division of a by b by $a \operatorname{div} b$.
- (ii) We denote the *remainder* of the division of a by b by $a \operatorname{mod} b$.

Example 3.2.11. We demonstrate the division with remainder notation with the numbers from Example 3.2.7.

- (i) $4 \operatorname{div} 7 = 0$ and $4 \operatorname{mod} 7 = 4$
- (ii) $30 \operatorname{div} 8 = 3$ and $30 \operatorname{mod} 8 = 6$
- (iii) $-20 \operatorname{div} 7 = -3$ and $-20 \operatorname{mod} 7 = 1$

3.3 Long Division

Algorithms 3.2.1 and 3.2.5 provide a way to determine the quotient and remainder of a division problem by repeatedly subtracting or adding a fixed value, and that process has a very different feel from the process of long division that is often introduced in early mathematics. While the steps provided in the Division Algorithm always produce the correct quotient and remainder for the division problem given by valid input values, the process could be quite long if the number of steps required is very big. So, we now provide a second (potentially more practical) way to use a basic calculator to get the quotient and remainder of a division problem. Rather than laboriously writing out the algorithmic steps of long division, we will demonstrate the process by example.

Example 3.3.1. Let $a := 300$ and $b := 16$. We perform the following long division:

$$\begin{array}{r} 18 \\ 16 \overline{)300} \\ \underline{16} \\ 140 \\ \underline{128} \\ 12 \end{array}$$

Then, $q = 18$ and $r = 12$, and we write $300 \operatorname{div} 16 = 18$ and $300 \operatorname{mod} 16 = 12$.

With a calculator the process of finding $q = a \operatorname{div} b$ and $r = a \operatorname{mod} b$ can be shortened.

Strategy 3.3.2 (Calculator Long Division). Suppose we are given an integer a and a natural number b . We give a strategy for finding the quotient $a \operatorname{div} b$ and the remainder $a \operatorname{mod} b$.

- (1) To find $a \operatorname{div} b$ and $a \operatorname{mod} b$ we set up the long division problem:

$$b \overline{)a}$$

- (2) Compute $a \div b$ with a calculator.
- (3) The quotient q is the biggest integer that is less than or equal to the numerical value of $a \div b$. If $a \div b$ is an integer, then $q := a \div b$, otherwise, q is the integer to the left of $a \div b$ on the number line. Place the entire quotient q on top of the long division:

$$\begin{array}{r} q \\ b \overline{)a} \end{array}$$

- (4) Multiply $b \cdot q$ and place that value under the long division:

$$\begin{array}{r} q \\ b \overline{)a} \\ b \cdot q \end{array}$$

(5) Subtract to get the remainder $r = a - (b \cdot q)$ (using Theorem 3.2.9):

$$\begin{array}{r} q \\ b \overline{)a} \\ \underline{b \cdot q} \\ r \end{array}$$

Example 3.3.3. (Example 3.3.1 – Revisited) We use Strategy 3.3.2 to compute the quotient q and remainder r of the division of 300 by 16.

(1) Set up the long division problem:

$$16 \overline{)300}$$

(2) A calculator gives us that $300 \div 16 = 18.75$.

(3) The integer on the number line to the left of 18.75 is $q := 18$. Place the entire quotient on top of the long division:

$$\begin{array}{r} 18 \\ 16 \overline{)300} \end{array}$$

(4) Multiply $16 \cdot 18 = 288$ and place that value under the long division:

$$\begin{array}{r} 18 \\ 16 \overline{)300} \\ 288 \end{array}$$

(5) Subtract to get the remainder $r = 300 - 288 = 12$:

$$\begin{array}{r} 18 \\ 16 \overline{)300} \\ 288 \\ \hline 12 \end{array}$$

Then, $q = 18$ and $r = 12$, and we write $300 \text{ div } 16 = 18$ and $300 \text{ mod } 16 = 12$.

Example 3.3.4. We use Strategy 3.3.2 to compute the quotient q and remainder r of the division of $a = -457$ by $b = 24$.

(1) Set up the long division problem:

$$24 \overline{)-457}$$

(2) A calculator gives us that $-457 \div 24 = -19.0416 \dots$

(3) The integer on the number line to the left of $-19.0416 \dots$ is $q := -20$. Place the entire quotient on top of the long division:

$$\begin{array}{r} -20 \\ 24 \overline{)-457} \end{array}$$

(4) Multiply $24 \cdot (-20) = -480$ and place that value under the long division:

$$\begin{array}{r} -20 \\ 24 \overline{)-457} \\ -480 \end{array}$$

(5) Subtract to get the remainder $r = -457 - (-480) = 23$:

$$\begin{array}{r} -20 \\ 24 \overline{) -457} \\ \underline{-480} \\ 23 \end{array}$$

Then $q = -20$ and $r = 23$. So we have $-457 \operatorname{div} 24 = -20$ and $-457 \bmod 24 = 23$.

Example 3.3.5. We find $10 \operatorname{div} 55$ and $10 \bmod 55$. Let $a = 10$ and $b = 55$. We use Strategy 3.3.2 to compute the quotient $q = 10 \operatorname{div} 55$ and remainder $r = 10 \bmod 55$ of the division of 10 by 55.

(1) Set up the long division problem:

$$55 \overline{) 10}$$

(2) A calculator gives us that $10 \div 55 = 0.18 \dots$

(3) The integer on the number line to the left of $0.18 \dots$ is $q = 0$. Place the entire quotient on top of the long division:

$$\begin{array}{r} 0 \\ 55 \overline{) 10} \end{array}$$

(4) Multiply $55 \cdot 0 = 0$ and place that value under the long division:

$$\begin{array}{r} 0 \\ 55 \overline{) 10} \\ 0 \end{array}$$

(5) Subtract to get the remainder $r = 10 - 0 = 10$:

$$\begin{array}{r} 0 \\ 55 \overline{) 10} \\ \underline{0} \\ 10 \end{array}$$

Then, $q = 0$ and $r = 10$, and we have $10 \operatorname{div} 55 = 0$ and $10 \bmod 55 = 10$.

When $a \div b$ is an integer then the quotient $a \operatorname{div} b$ is equal to $a \div b$ and the remainder is zero. We demonstrate that we also obtain this result using Strategy 3.3.2.

Example 3.3.6. We find $480 \operatorname{div} 160$ and $480 \bmod 160$. We use Strategy 3.3.2.

(1) Set up the long division problem:

$$160 \overline{) 480}$$

(2) A calculator gives us that $480 \div 160 = 3$.

(3) As $480 \div 160$ is an integer we get $q := 480 \div 160 = 3$. Place the entire quotient on top of the long division:

$$\begin{array}{r} 3 \\ 160 \overline{) 480} \end{array}$$

(4) Multiply $160 \cdot 3 = 480$ and place that value under the long division:

$$\begin{array}{r} 3 \\ 160 \overline{)480} \\ 480 \end{array}$$

(5) Subtract to get the remainder $r := 480 - 480 = 0$:

$$\begin{array}{r} 3 \\ 160 \overline{)480} \\ \underline{480} \\ 0 \end{array}$$

So $q = 3$ and $r = 0$, and we have $480 \operatorname{div} 160 = 3$ and $480 \bmod 160 = 0$.

In practice we often do not explicitly write down all steps of the strategy.

Example 3.3.7. We compute $-107 \bmod 72$ and $-107 \operatorname{div} 72$ with a calculator. We have

$$-107 \div 72 = -1.48611\dots$$

The integer on the number line to the left of $-1.48611\dots$ is -2 (starting at -1.48611 we go left on the number line until we find an integer). Thus -2 is the quotient.

Now we compute the remainder by subtracting the quotient -2 times 72 from -107 .

$$-107 - (-2) \cdot 72 = -107 - (-144) = -107 + 144 = 37$$

We have computed $-107 \operatorname{div} 72 = -2$ and $-107 \bmod 72 = 37$.

Example 3.3.8. We compute $9087 \operatorname{div} 87$ and $9087 \bmod 87$. A calculator gives us:

$$9087 \div 87 = 104.4482\dots$$

The closest integer to the left of $104.4482\dots$ on the number line is 104 . This is the quotient. Now we use it to compute the remainder:

$$9087 - 104 \cdot 87 = 39$$

We have found $9087 \operatorname{div} 87 = 104$ and $9087 \bmod 87 = 39$.

3.4 The Operation mod

We now investigate the operation $\bmod$ further. Recall that $a \bmod b$ is the remainder of the division of a by b (see Definition 3.2.10). We have established that the division algorithm (Algorithms 3.2.1 and 3.2.5) produces the quotient and remainder of a particular division as its output values. As an alternative method we have presented (calculator) long division. In the following example we compute the remainder by inspection and using these two methods.

Example 3.4.1. We compute $41 \bmod 13$ in three different ways. The number $41 \bmod 13$ is the remainder of the division of 41 by 13.

Method 1 the trained eye The largest multiple of 13 that is less than 41 is 39. The difference between 41 and 39 is 2, this is the remainder of the division of 41 by 39. Thus $41 \bmod 13 = 2$.

Method 2 calculator long division We have $41 \div 13 = 3.153\dots$, thus the quotient of the division of 41 by 13 is 3. The remainder is $41 - 3 \cdot 13 = 41 - 39 = 2$. Thus $41 \bmod 13 = 2$.

Method 3 division algorithm We subtract 13 until we get a number in the remainder target range from 0 to $13 - 1 = 12$:

$$41 - 13 = 28$$

$$28 - 13 = 15$$

$$15 - 13 = 2$$

As 2 is in the remainder target range from 0 to $13 - 1 = 12$ it is the remainder. Thus $41 \bmod 13 = 2$.

Example 3.4.2. We provide this example to point out that the remainders ‘wrap around’.

$$0 \bmod 3 = 0$$

$$1 \bmod 3 = 1$$

$$2 \bmod 3 = 2$$

$$3 \bmod 3 = 0$$

$$4 \bmod 3 = 1$$

$$5 \bmod 3 = 2$$

$$6 \bmod 3 = 0$$

$$7 \bmod 3 = 1$$

$$8 \bmod 3 = 2$$

The remainder of division by 2 is either 0 or 1. We use this to define two familiar terms.

Definition 3.4.3. An integer n is *even* means that $n \bmod 2 = 0$.

Definition 3.4.4. An integer n is *odd* means that $n \bmod 2 = 1$.

We now investigate some properties of the operation $\bmod$. In particular, we are interested in the behavior of $\bmod$ in sums and products.

We build upon our observations in the example to formulate statements about addition and multiplication in combination with the operation $\bmod$.

Theorem 3.4.5. *Let a and b be integers, and let m be a natural number. Then*

$$(a + b) \bmod m = ((a \bmod m) + (b \bmod m)) \bmod m$$

.

Proof. We have that

$$a = (a \bmod m) + (s \cdot m) \text{ for some integer } s, \text{ and}$$

$$b = (b \bmod m) + (t \cdot m) \text{ for some integer } t.$$

With the notation above, we have

$$\begin{aligned}
 (a + b) \bmod m &= (((a \bmod m) + (s \cdot m)) + ((b \bmod m) + (t \cdot m))) \bmod m \\
 &= ((a \bmod m) + (b \bmod m) + (s + t) \cdot m) \bmod m \\
 &= ((a \bmod m) + (b \bmod m)) \bmod m.
 \end{aligned}$$

□

Example 3.4.6. We illustrate Theorem 3.4.5 with an example. We compute $(10+20) \bmod 7$ in two ways, namely directly and applying the theorem.

- (i) $(10 + 20) \bmod 7 = 30 \bmod 7 = 2$
- (ii) $(10 + 20) \bmod 7 = ((10 \bmod 7) + (20 \bmod 7)) \bmod 7 = (3 + 6) \bmod 7 = 9 \bmod 7 = 2$

Using the theorem may seem more awkward right now. When calculations get more involved its value will become more apparent. The theorem also can be used to evaluate expressions when we only know the remainders.

Problem 3.4.7. Let a and b be integers with $a \bmod 113 = 29$ and $b \bmod 113 = 100$. Compute $(a + b) \bmod 113$.

Solution. By Theorem 3.4.5 we have

$$(a + b) \bmod 113 = ((a \bmod 113) + (b \bmod 113)) \bmod 113$$

As we know that $a \bmod 113 = 29$ and $b \bmod 113 = 100$ we can replace $a \bmod 113$ by 29 and $b \bmod 113$ by 100. Copying what we have so far and evaluating we get:

$$\begin{aligned}
 (a + b) \bmod 113 &= ((a \bmod 113) + (b \bmod 113)) \bmod 113 \\
 &= (29 + 100) \bmod 113 = 129 \bmod 113 = 16.
 \end{aligned}$$

The operation mod also behaves nicely under multiplication.

Theorem 3.4.8. Let a and b be integers, and let m be a natural number. Then

$$(a \cdot b) \bmod m = ((a \bmod m) \cdot (b \bmod m)) \bmod m.$$

Proof. We have that

$$\begin{aligned}
 a &= (a \bmod m) + (s \cdot m) \text{ for some integer } s, \text{ and} \\
 b &= (b \bmod m) + (t \cdot m) \text{ for some integer } t.
 \end{aligned}$$

$$\begin{aligned}
 (a \cdot b) \bmod m &= (((a \bmod m) + s \cdot m) \cdot ((b \bmod m) + t \cdot m)) \bmod m \\
 &= ((a \bmod m) \cdot (b \bmod m) \\
 &\quad + (a \bmod m) \cdot t \cdot m + s \cdot m \cdot (b \bmod m) + s \cdot m \cdot t \cdot m) \bmod m \\
 &= ((a \bmod m) \cdot (b \bmod m) \\
 &\quad + ((a \bmod m) \cdot t + s \cdot (b \bmod m) + s \cdot m \cdot t) \cdot m) \bmod m \\
 &= ((a \bmod m) \cdot (b \bmod m)) \bmod m.
 \end{aligned}$$

□

Theorems 3.4.5 and 3.4.8 is particularly useful when computing mod with larger numbers.

Example 3.4.9. We apply Theorem 3.4.8 to compute $(20 \cdot 10) \bmod 7$ in two ways.

$$(20 \cdot 10) \bmod 7 = ((20 \bmod 7) \cdot (10 \bmod 7)) \bmod 7 = (6 \cdot 3) \bmod 7 = 18 \bmod 7 = 4$$

is longer but easier to compute than

$$(10 \cdot 20) \bmod 7 = 200 \bmod 7 = 4,$$

since we avoid the computation of $200 \bmod 7$.

Example 3.4.10. We apply Theorems 3.4.5 and 3.4.8 to compute $(61 + (9 \cdot 8)) \bmod 11$.

$$(61 + (9 \cdot 8)) \bmod 11 = (61 \bmod 11 + 72 \bmod 11) \bmod 11 = (6 + 6) \bmod 11 = 12 \bmod 11 = 1$$

Example 3.4.11. With Calculator long division we compute:

- (i) $8082 \bmod 17 = 7$
- (ii) $4540 \bmod 17 = 1$
- (iii) $4496 \bmod 17 = 8$

Knowing these numbers and applying Theorems 3.4.5 and 3.4.8 we find the following remainders without having to add or multiply large numbers.

- (i) $(8082 \cdot 4540) \bmod 17 = ((8082 \bmod 17) \cdot (4540 \bmod 17)) \bmod 17 = (7 \cdot 1) \bmod 17 = 7$
- (ii) $(4540 + 4496) \bmod 17 = ((4540 \bmod 17) + (4496 \bmod 17)) \bmod 17 = (1 + 8) \bmod 17 = 9$
- (iii) $(8082 \cdot 4540 + 4496) \bmod 17 = (((8082 \cdot 4540) \bmod 17) + (4496 \bmod 17)) \bmod 17 = (7 + 8) \bmod 17 = 0$
- (iv) $(8082 + 4540 + 4496) \bmod 17 = ((8082 \bmod 17) + ((4540 + 4496) \bmod 17)) \bmod 17 = (7 + 9) \bmod 17 = 16 \bmod 17 = 16$

In the following we make use of the decimal representation of numbers, to find remainders of divisions of numbers that would be too large to handle for most calculators. If you need a refresher on decimal numbers, read section 11.1.

Problem 3.4.12. Find $23829913346008023471 \bmod 20$.

Solution. As 20 is a divisor of 100 Theorem 3.4.5 helps considerably. First notice that

$$23829913346008023471 = (238299133460080234 \cdot 100) + 71.$$

With Theorem 3.4.5 we get:

$$\begin{aligned} 23829913346008023471 \bmod 20 &= ((238299133460080234 \cdot 100) + 71) \bmod 20 \\ &= (((238299133460080234 \cdot 100) \bmod 20) + (71 \bmod 20)) \bmod 20 \\ &= (0 + 71) \bmod 20 = 71 \bmod 20 = 11 \end{aligned}$$

Problem 3.4.13. Find $23829913346008023471 \bmod 5$.

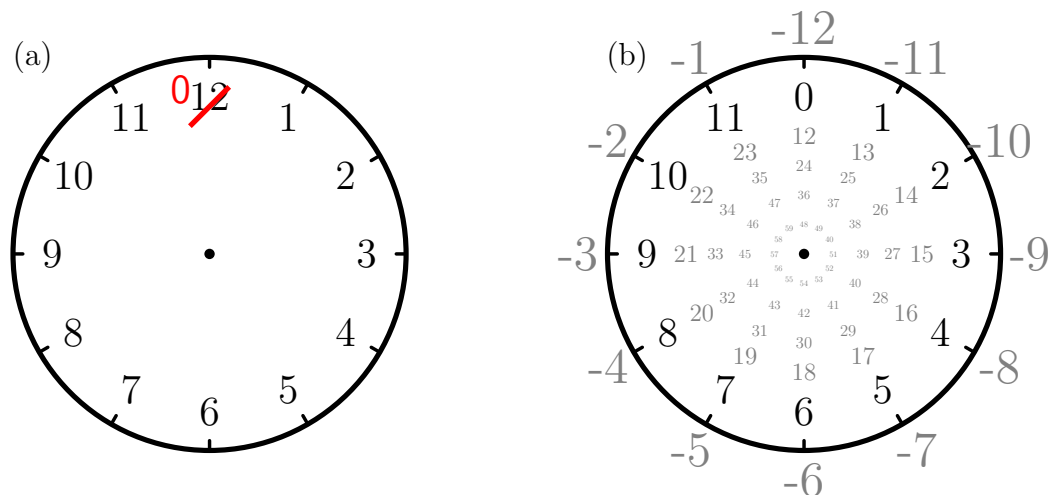
Solution. As 5 divides 10 we write

$$23829913346008023471 = (2382991334600802347 \cdot 10) + 1.$$

As $10 \bmod 5 = 0$ we immediately see that

$$23829913346008023471 \bmod 5 = (0 + 1) \bmod 5 = 1.$$

Figure 3.5.1: Two ways of picturing arithmetic modulo 12. (a) We conduct the familiar addition of hours and replace 12 on the clock face by 0. (b) We wrap the number line (compare Figure 1.1.1) around a circle such that all numbers with the same remainder from division by 12 are in the same position.



3.5 Clock Arithmetic

In the following we give applications of the combination of the operation mod and the addition of integers called clock arithmetic.

Recall that the operation mod yields the remainder of integer division. That is, for an integer a and a natural number b the number $r = a \bmod b$ is the number such that $a = (b \cdot q) + r$ for some integer q and r is a non-negative integer and $r < b$.

Example 3.5.1. We give some examples of remainders.

- (i) $0 \bmod 7 = 0$
- (ii) $1 \bmod 7 = 1$
- (iii) $-1 \bmod 7 = 6$
- (iv) $8 \bmod 7 = 1$
- (v) $7 \bmod 7 = 0$
- (vi) $10 \bmod 7 = 3$
- (vii) $14 \bmod 7 = 0$
- (viii) $100 \bmod 7 = 2$

We give some applications of the operation mod, namely the arithmetic of hours, days of the week, and months. adding hours, days of the week, and months. We start with examples of adding hours and then relate this to using addition and the operation mod.

Example 3.5.2. When using the 12 hour clock we have:

- (i) An hour after 11 o'clock it is 12 o'clock.
- (ii) Two hours after 11 o'clock it is 1 o'clock.

- (iii) 10 hours after 11 o'clock it is 9 o'clock.
- (iv) 20 hours after 11 o'clock it is 7 o'clock.
- (v) 25 hours after 11 o'clock it is 12 o'clock.

These operations can be considered as adding hours to a time. To compute these additions we add the hours and then subtract 12 as many times as necessary to obtain a number between 1 and 12. With a similar method we had computed the remainder in Algorithm 3.2.1. The main difference between the two approaches is that using the 12 hour clock we obtain numbers between 1 and 12 and when computing remainders we obtain numbers between 0 and 11. That is, we replace 12 by 0, compare Figure 3.5.1 (a). We call this arithmetic modulo 12. Figure 3.5.1 (b) illustrates how the number line wraps around the clock face in arithmetic modulo 12. The remainder modulo 12 of two numbers is the same if they differ by a multiple of 12.

Example 3.5.3. We formulate the computations from Example 3.5.2 using remainders. Recall that we denoted the remainder of the division of a by 12 by $a \bmod 12$.

- (i) $(11 + 1) \bmod 12 = 12 \bmod 12 = 0$
- (ii) $(11 + 2) \bmod 12 = 13 \bmod 12 = 1$
- (iii) $(11 + 10) \bmod 12 = 21 \bmod 12 = 9$
- (iv) $(11 + 20) \bmod 12 = 31 \bmod 12 = 7$
- (v) $(11 + 25) \bmod 12 = 36 \bmod 12 = 0$

By Theorem 3.4.5 that we can add first or take the remainder first, and we will get the same answer, so in (iv) and (v) above, we could have done the following:

- (i) $(11 + 20) \bmod 12 = ((11 \bmod 12) + (20 \bmod 12)) \bmod 12 = (11 + 8) \bmod 12 = 19 \bmod 12 = 7$
- (ii) $(11 + 25) \bmod 12 = ((11 \bmod 12) + (25 \bmod 12)) \bmod 12 = (11 + 1) \bmod 12 = 12 \bmod 12 = 0$

The second computation appears to have more steps, but the arithmetic can be much simpler. In practice we combine both approaches.

Problem 3.5.4. *What time is it in 79 hours if it is 4 o'clock now ?*

Solution. We have $(4 + 79) \bmod 12 = 83 \bmod 12 = 11$. Thus 79 hours from now, it is 11 o'clock.

The same result can also be obtained by first computing $79 \bmod 12 = 7$ and then $(4 + 7) \bmod 12 = 11 \bmod 12 = 11$.

Everything we have done for hours above also works for other counts that wrap around.

Problem 3.5.5. *Which day of the week is it in 110 days from today if today is Friday ?*

Solution. The days of the week wrap around after seven days. When adding days and we want the result as a weekday, any multiples of 7 do not change the day of the week. Instead of adding 110 days we add $110 \bmod 7 = 5$ days. So 110 days after Friday is the same day of the week as 5 days after Friday, namely Wednesday.

Problem 3.5.6. Which month is it 721 months from now if this month is November?

Solution. Months wrap around after 12 months. We have $721 \bmod 12 = 1$. Since December is one month after November, 721 months from now, it will be December.

3.6 Application: ISBN

An *ISBN* (International Standard Book Number) is a number that uniquely identifies a book. Until 2007, the assigned ISBNs were 10 digits long, but the newly-assigned ISBNs are 13 digits long. As a practical application of the operation mod, we discuss the check digit of 10-digit ISBNs.

This is an example of an application of an area of mathematics called *coding theory*. The code used for the ISBN-10 is an error detecting code. That means it can be detect whether a common error, such as a wrong digit or two swapped digits, was made in handling the number.

We write a 10-digit ISBN-10 as

$$x_1 - x_2 x_3 x_4 - x_5 x_6 x_7 x_8 x_9 - x_{10}.$$

Here $x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9$, and x_{10} are variables, that is, placeholders for numbers. The purpose of the numbers in the subscript is to distinguish the variables. The choice of numbering also hints on the order. The dashes “-” do not mean subtraction but are there to group the numbers.

Each of the first 9 digits x_1 to x_9 is an integer from 0 to 9. The 10th digit x_{10} is called a *check digit*, used to detect certain errors in ISBN-10s. The possible values for the last digit x_{10} are integers from 0 to 9 and the letter X, interpreted as the value 10.

As there would be too many parentheses in the following computations, we amend our conventions concerning the *order of operations*. We evaluate multiplication before addition, so that instead of $(a \cdot b) + c$ we can write $a \cdot b + c$.

Definition 3.6.1. The digits

$$x_1 - x_2 x_3 x_4 - x_5 x_6 x_7 x_8 x_9 - x_{10}.$$

form a valid ISBN-10 if

$$x_{10} = (1 \cdot x_1 + 2 \cdot x_2 + 3 \cdot x_3 + 4 \cdot x_4 + 5 \cdot x_5 + 6 \cdot x_6 + 7 \cdot x_7 + 8 \cdot x_8 + 9 \cdot x_9) \bmod 11.$$

Problem 3.6.2. Determine whether 0-345-45374-3 is a valid ISBN-10.

Solution. We use the notation from Definition 3.6.1. We have $x_1 = 0, x_2 = 3, x_3 = 4, x_4 = 5, x_5 = 4, x_6 = 5, x_7 = 3, x_8 = 7$, and $x_9 = 4$. By Definition 3.6.1

$$x_1 - x_2 x_3 x_4 - x_5 x_6 x_7 x_8 x_9 - x_{10}$$

is an ISBN-10 when

$$x_{10} = (1 \cdot x_1 + 2 \cdot x_2 + 3 \cdot x_3 + 4 \cdot x_4 + 5 \cdot x_5 + 6 \cdot x_6 + 7 \cdot x_7 + 8 \cdot x_8 + 9 \cdot x_9) \bmod 11.$$

We have

$$\begin{aligned} x_{10} &= (1 \cdot x_1 + 2 \cdot x_2 + 3 \cdot x_3 + 4 \cdot x_4 + 5 \cdot x_5 + 6 \cdot x_6 + 7 \cdot x_7 + 8 \cdot x_8 + 9 \cdot x_9) \bmod 11 \\ &= (1 \cdot 0 + 2 \cdot 3 + 3 \cdot 4 + 4 \cdot 5 + 5 \cdot 4 + 6 \cdot 5 + 7 \cdot 3 + 8 \cdot 7 + 9 \cdot 4) \bmod 11 \\ &= (0 + 6 + 12 + 20 + 20 + 30 + 21 + 56 + 36) \bmod 11 \\ &= (0 + 6 + 1 + 9 + 9 + 8 + 10 + 1 + 3) \bmod 11 = 47 \bmod 11 = 3. \end{aligned}$$

As the last digit 0-345-45374-3 is 3 and we have computed that $x_{10} = 3$ we conclude that 0-345-45374-3 is a valid ISBN-10.

Problem 3.6.3. *Determine whether 0-475-02548-7 is a valid ISBN-10.*

Solution. We use the notation from Definition 3.6.1. We have $x_1 = 0$, $x_2 = 4$, $x_3 = 7$, $x_4 = 5$, $x_5 = 0$, $x_6 = 2$, $x_7 = 5$, $x_8 = 4$, and $x_9 = 8$. So the last digit of the ISBN-10 0-475-02548- x_{10} is

$$\begin{aligned} x_{10} &= (1 \cdot x_1 + 2 \cdot x_2 + 3 \cdot x_3 + 4 \cdot x_4 + 5 \cdot x_5 + 6 \cdot x_6 + 7 \cdot x_7 + 8 \cdot x_8 + 9 \cdot x_9) \bmod 11 \\ &= (1 \cdot 0 + 2 \cdot 4 + 3 \cdot 7 + 4 \cdot 5 + 5 \cdot 0 + 6 \cdot 2 + 7 \cdot 5 + 8 \cdot 4 + 9 \cdot 8) \bmod 11 \\ &= (0 + 8 + 21 + 20 + 0 + 12 + 35 + 32 + 72) \bmod 11 \\ &= (0 + 8 + 10 + 9 + 0 + 1 + 2 + 10 + 6) \bmod 11 = 46 \bmod 11 = 2 \end{aligned}$$

As the last digit of 0-475-02548-7 is 7 and we have computed that $x_{10} = 2$ we conclude that 0-475-02548-7 is not a valid ISBN-10.

Problem 3.6.4. *The ISBN-10 of Euclid's The Thirteen Books of the Elements, Vol. 1: Books 1-2 is 0-486-60088- x_{10} , where the check digit x_{10} is missing. Determine the complete ISBN-10.*

Solution. We use the formula from Definition 3.6.1

$$x_{10} = (1 \cdot x_1 + 2 \cdot x_2 + 3 \cdot x_3 + 4 \cdot x_4 + 5 \cdot x_5 + 6 \cdot x_6 + 7 \cdot x_7 + 8 \cdot x_8 + 9 \cdot x_9) \bmod 11$$

to compute the missing check digit x_{10} .

$$\begin{aligned} x_{10} &= (1 \cdot 0 + 2 \cdot 4 + 3 \cdot 8 + 4 \cdot 6 + 5 \cdot 6 + 6 \cdot 0 + 7 \cdot 0 + 8 \cdot 8 + 9 \cdot 8) \bmod 11 \\ &= (8 + 24 + 24 + 30 + 64 + 72) \bmod 11 = (8 + 2 + 2 + 8 + 9 + 6) \bmod 11 = 2 \end{aligned}$$

So, the complete ISBN-10 is 0-486-60088-2.

The ISBN is constructed in such a way that certain common errors that occur can be detected. For example, multiplying the first digit by 10, the second by 9, and so on, makes it possible to detect whether two digits an ISBN-10 have been inadvertently swapped. The final example of this section demonstrates the detection of such an error. Additionally, notice that we apply Theorem 3.4.5 to help us reduce each entry in the sum prior to adding so that we can work with smaller numbers.

Example 3.6.5. Marion tried to enter the ISBN-10 of a book she wanted to purchase on a booksellers web page. Instead of entering the number 3-540-13140-X she entered 3-450-13140-X. The booksellers web server computes

$$\begin{aligned}x_{10} &= (1 \cdot 3 + 2 \cdot 4 + 3 \cdot 5 + 4 \cdot 0 + 5 \cdot 1 + 6 \cdot 3 + 7 \cdot 1 + 8 \cdot 4 + 9 \cdot 0) \bmod 11 \\&= (3 + 8 + 15 + 0 + 5 + 18 + 7 + 32 + 0) \bmod 11 \\&= (3 + 8 + 4 + 0 + 5 + 7 + 7 + 10) \bmod 11 = 33 \bmod 11 = 0.\end{aligned}$$

As the tenth digit entered by Marion is X which stands for 10 and $10 \neq 0$ the web server sends the message, that there is no book with the requested ISBN-10 number.

We have seen that the operation mod is used in the authentication of ISBN numbers. Similar methods are used in the authentication of other numbers such as serial numbers of banknotes and credit card numbers,.

Chapter 4

Divisors

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Recognize the Euclidean algorithm.
- (2) Compute greatest common divisors.
- (3) Compute the cofactors from a special case of Bézout's identity.

In this section, we introduce another important algorithm, the Euclidean Algorithm (Algorithm 4.3.1). This algorithm gives us a way to systematically determine the greatest common divisor of two natural numbers. Then, we show how to use the computations in the Euclidean Algorithm (Algorithm 4.3.1) to determine the integers whose existence is guaranteed by Bézout's Identity (Theorem 4.4.1).

4.1 Divisibility

We begin by introducing terminology.

Definition 4.1.1. Suppose that for integers a and b , there is an integer q such that $a = b \cdot q$. Then b *divides* a .

There are several other formulations for b *divides* a , for example

- (i) a is *divisible* by b
- (ii) a is a *multiple* of b
- (iii) b *divides* a
- (iv) b is a *divisor* of a
- (v) b is a *factor* of a

By Definition 4.1.1 if b divides a , then $a = b \cdot q$ for some integer q . Then we have $a = b \cdot q + 0$ so that in particular $a \bmod b = 0$. If b does not divide a , then $a \bmod b \neq 0$. It follows immediately that if b divides a then $b \leq a$.

Problem 4.1.2. For the given values of a and b , determine whether or not b divides a . If b divides a , determine the integer q such that $a = b \cdot q$.

- (i) $a = 30$ and $b = 10$
- (ii) $a = 2$ and $b = 46$
- (iii) $a = 29$ and $b = 4$

Solution. In each case we consider the remainder $a \bmod b$ of the division a and b .

- (i) We compute $30 \bmod 10 = 0$. So 10 divides 30. Furthermore, we have that $30 \operatorname{div} 10 = 3$ so $30 = 10 \cdot 3$.
- (ii) We compute $2 \bmod 46 = 2 \neq 0$. So 46 does not divide 2. (Be careful not to mix up a and b during the division or the conclusion. The order matters. It turns out that 2 does divide 46 since $46 = 2 \cdot 23$.)
- (iii) We compute $29 \bmod 4 = 1 \neq 0$. So 4 does not divide 29.

If a number divides two other numbers, it divides their sum.

Theorem 4.1.3. *Let b be a natural number and let a and c be integers. If b divides a and b divides c , then b divides $a + c$.*

Proof. As b divides a , there is an integer q such that $a = b \cdot q$. As b divides c , there is an integer s such that $c = b \cdot s$. With substitution and the distributive property we obtain

$$a + c = (b \cdot q) + (b \cdot s) = b \cdot (q + s).$$

Thus $a + c$ is a multiple of b which means that b divides $a + c$. □

Example 4.1.4. Let $b := 10$ and $a := 100$ and $c := 1000$. Then b divides a and b divides c . Also b divides $a + c = 1100$.

4.2 Greatest Common Divisors

Definition 4.2.1. Let a and b be integers. The greatest natural number g that divides both a and b is called the *greatest common divisor* of a and b and is denoted by $\gcd(a, b)$. We say a and b are *coprime* if $\gcd(a, b) = 1$.

In the definition the order of a and b does not matter. We get:

Theorem 4.2.2. *Let a and b be integers. Then $\gcd(a, b) = \gcd(b, a)$.*

Example 4.2.3. We find the greatest common divisor of 12 and 42. As all divisors of 12 are less than or equal to 12, we only have to check numbers less than 12.

- 1 divides both 12 and 42
- 2 divides both 12 and 42
- 3 divides both 12 and 42
- 4 divides 12 but not 42
- 5 does not divide 12 or 42
- 6 divides both 12 and 42
- 7 does not divide 12, but does divide 42
- 8 does not divide 12 or 42
- 9 does not divide 12 or 42
- 10 does not divide 12 or 42
- 11 does not divide 12 or 42

- 12 divides 12 but not 42

Thus the greatest integer that divides both 12 and 42 is 6, that is, $\gcd(12, 42) = 6$.

Example 4.2.4. We give the greatest common divisor in some special cases. Let a be a natural number. Then

- (i) $\gcd(a, a) = a$, as the largest natural number that divides a is a .
- (ii) $\gcd(0, a) = a$, as 0 is divisible by all natural numbers a .
- (iii) $\gcd(1, a) = 1$, as the largest divisor of 1 is 1 and all natural numbers a are divisible by 1.

Our next goal is to find an efficient method for finding greatest common divisors. Recall that remainder of division of a and b is $a - (b \cdot q)$ where $q := a \operatorname{div} b$. Theorem 4.2.5 tells us that we can use the remainder to help us find the greatest common divisor.

Theorem 4.2.5. Let g be a natural number and let a , b , and q be integers.

- (i) If g divides a and g divides b then g also divides $a - (b \cdot q)$.
- (ii) If g divides $a - (b \cdot q)$ and g divides b then g also divides a .
- (iii) $\gcd(a - (b \cdot q), b) = \gcd(a, b)$.
- (iv) $\gcd(a \bmod b, b) = \gcd(a, b)$.

Proof. In the proof of (i) and (ii) we follow an approach similar to that of the proof of Theorem 4.1.3. We apply (i) and (ii) in the proof of (iii) and use (iii) to prove (iv).

- (i) As g divides a there exists an integer s such that $a = g \cdot s$ and as g divides b there exists an integer t such that $b = g \cdot t$. We now have

$$a - (b \cdot q) = (g \cdot s) - ((g \cdot t) \cdot q) = g \cdot (s - (t \cdot q)).$$

Thus $a - (b \cdot q)$ is a multiple of g which means that g divides $a - (b \cdot q)$.

- (ii) As g divides $a - (b \cdot q)$ there exists an integer s such that $a - (b \cdot q) = g \cdot s$ and as g divides b , there exists an integer t such that $b = g \cdot t$. We now have

$$a = (a - (b \cdot q)) + (b \cdot q) = (g \cdot s) + ((g \cdot t) \cdot q) = g \cdot (s + (t \cdot q))$$

Thus a is a multiple of g which means that g divides a .

- (iii) By (i) all natural numbers g that divide a and b also divide $a - (b \cdot q)$. By (ii) all natural numbers g that divide $a - (b \cdot q)$ and b also divide a . Thus the common divisors of a and b and the common divisors of $a - (b \cdot q)$ are the same. So, in particular, the greatest common divisor of a and b is equal to the greatest common divisor of $a - (b \cdot q)$ and b .

- (iv) For $q := a \operatorname{div} b$ we have $a \bmod b = a - (b \cdot q)$. With (iii) we get

$$\gcd(a \bmod b, b) = \gcd(a - (b \cdot q), b) = \gcd(a, b).$$

□

We now repeatedly apply Theorem 4.2.5(iv) to find the greatest common divisor of two integers.

Example 4.2.6. Let $a := 51$ and $b := 15$. We find $\gcd(a, b)$. With Theorem 4.2.5(iv) we get

$$\gcd(51, 15) = \gcd(51 \bmod 15, 15) = \gcd(6, 15).$$

By Theorem 4.2.2 we have

$$\gcd(6, 15) = \gcd(15, 6).$$

Applying Theorem 4.2.5 (iv) we obtain

$$\gcd(15, 6) = \gcd(15 \bmod 6, 6) = \gcd(3, 6).$$

By Theorem 4.2.2 we have

$$\gcd(3, 6) = \gcd(6, 3).$$

With Theorem 4.2.5(iv) we get

$$\gcd(6, 3) = \gcd(6 \bmod 3, 3) = \gcd(0, 3).$$

By Example 4.2.4

$$\gcd(0, 3) = 3.$$

So we have found that

$$\gcd(51, 15) = \gcd(51 - (15 \cdot 3), 15) = \gcd(6, 15) = \gcd(15, 6) = \gcd(3, 6) = \gcd(6, 3) = \gcd(0, 3) = 3.$$

That is, $\gcd(51, 15) = 3$

4.3 The Euclidean Algorithm

We formulate an algorithm for computing greatest common divisors that follows the strategy we used in Example 4.2.6. As in the example we repeatedly apply Theorem 4.2.5(iv) to reduce the computation of $\gcd(a, b)$ to the $\gcd(a \bmod b, b)$. This makes the numbers of which we compute the greatest common divisor smaller in every step, until the remainder $a \bmod b$ is zero.

The algorithm is named after the Greek mathematician Euclid, who first described it in Book 7 of his *Elements* (around 300 BC)¹. To make the representation of the algorithm easier, we only allow natural numbers (positive integers) as inputs.

Algorithm 4.3.1 (*Euclidean*).

Input: Two natural numbers a and b with $a > b$

Output: The greatest common divisor $\gcd(a, b)$ of a and b

- (1) **repeat**
 - (a) **let** $r := a \bmod b$
 - (b) **let** $a := b$

¹euclid.

- (c) **let** $b := r$
- (2) **until** $r = 0$
- (3) **return** a

In the algorithm, r becomes smaller in each iteration of the loop. As r is a non-negative integer, it has to become zero eventually. Thus after finitely many steps the algorithm returns a result.

Example 4.3.2. We compute the greatest common divisor of 612 and 56 with Algorithm 4.3.1.

Input: $a := 612$ and $b := 56$

- (1) (a) $r := 612 \bmod 56 = 52$
- (b) $a := 56$
- (a) $b := 52$
- (2) As $r = 52$ the statement $r = 0$ is false. So we continue with (1).
- (1) (a) $r := 56 \bmod 52 = 4$
- (b) $a := 52$
- (a) $b := 4$
- (2) As $r = 4$ the statement $r = 0$ is false. So we continue with (1).
- (1) (a) $r := 52 \bmod 4 = 0$
- (b) $a := 4$
- (a) $b := 0$
- (2) As $r = 0$ the statement $r = 0$ is false. So we continue with (1).
- (3) We return the value of a which is 4.

Output: 4

We have found that the greatest common divisor of 612 and 52 is 4.

Next we repeat the previous example. Instead of explicitly writing down what happens in every step, we write down the value of each variable at the end of step (1). This notation is more suitable for the computation of greatest common divisors by hand.

Example 4.3.3. We compute the greatest common divisor of 612 and 56 with Algorithm 4.3.1.

In the table we give the values of the variables at the end of step (1) in each iteration of the loop.

step	r	a	b
<i>Input</i>		612	56
(1)	$612 \bmod 56 = 52$	56	52
(1)	$56 \bmod 52 = 4$	52	4
(1)	$52 \bmod 4 = 0$	4	0
<i>Output</i>		4	

Thus the output is $\gcd(612, 56) = 4$.

More abstractly, we use the Euclidean Algorithm (Algorithm 4.3.1) to prove the next result.

Theorem 4.3.4. *Consecutive natural numbers $n + 1$ and n are coprime.*

Proof. Let n be a natural number. We compute $\gcd(n + 1, n)$ using the Euclidean Algorithm, with Algorithm 4.3.1. In the table we give the values of the variables after step (1) in each iteration of the loop.

step	r	a	b
<i>Input</i>		$n + 1$	n
(1)	$(n + 1) \bmod n = 1$	n	1
(1)	$n \bmod 1 = 0$	1	0
<i>Output</i>		1	

Thus, $\gcd(n + 1, n) = 1$, and we conclude that $n + 1$ and n are coprime. $\square$

We illustrate the proof of the theorem with a numerical example.

Example 4.3.5. We compute the greatest common divisor of 238 and 237 with Algorithm 4.3.1. In the table we give the values of the variables after step (1) in each iteration of the loop.

step	r	a	b
<i>Input</i>		238	237
(1)	$238 \bmod 237 = 1$	237	1
(1)	$237 \bmod 1 = 0$	1	0
<i>Output</i>		1	

Thus the output is $\gcd(238, 237) = 1$.

4.4 Bézout's Identity

The following theorem follows from the Euclidean Algorithm (Algorithm 4.3.1) and Theorem 3.2.9.

Theorem 4.4.1 (Bézout's Identity). *For all natural numbers a and b there exist integers s and t with $(s \cdot a) + (t \cdot b) = \gcd(a, b)$.*

The values s and t from Theorem 4.4.1 are called the *cofactors* of a and b . To find s and t for any a and b , we would use repeated substitutions on the results of the Euclidean Algorithm (Algorithm 4.3.1). This works because the algorithm connects a and b to the $\gcd(a, b)$ by a series of related equations.

When $\gcd(a, b) = a \bmod b$, we can easily find the values of s and t from Theorem 4.4.1. In this course we limit our computations to this case. We demonstrate this in the following examples.

Example 4.4.2. We find values for s and t from Theorem 4.4.1 for $a := 28$ and $b := 12$.

First, we compute the $\gcd(28, 12)$ using the Euclidean Algorithm (Algorithm 4.3.1). In the table we give the values of the variables at the end of step (1) in each iteration of the loop.

step	r	a	b
<i>Input</i>		28	12
(1)	$28 \bmod 12 = 4$	12	4
(1)	$12 \bmod 4 = 0$	1	0
<i>Output</i>		4	

So the $\gcd(28, 12) = 28 \bmod 12 = 4$. To find s and t with $(s \cdot 28) + (t \cdot 12) = \gcd(28, 12) = 4$ we need

- the remainder from the first iteration of the loop $r := a \bmod b = 28 \bmod 12 = 4$ and
- the quotient $q := a \operatorname{div} b = 28 \operatorname{div} 12 = 2$.

Now we can write a in the form $a = b \cdot q + r$:

$$28 = 12 \cdot 2 + 4$$

We write $a = (b \cdot q) + r$ in slightly more complicated way, namely as $(1 \cdot a) = (q \cdot b) + r$. Solving $(1 \cdot a) = (q \cdot b) + r$ for r we get $(1 \cdot a) - (q \cdot b) = r$. To bring this into the desired form $(s \cdot a) + (t \cdot b) = \gcd(a, b)$ we write $-(q \cdot b)$ as $+((-q) \cdot b)$ and obtain

$$(1 \cdot a) + ((-q) \cdot b) = r$$

Plugging in our values for a , b , q , and r we obtain

$$(1 \cdot 28) + ((-2) \cdot 12) = 4$$

So $s = 1$ and $t = -2$.

Note, that we obtain $s = 1$ as the Euclidean algorithm only needed two steps to compute the greatest common divisor. The cofactors s and t are not unique. Using the numbers from the example above, we could also have gotten $(s \cdot 28) + (t \cdot 12) = 4$ for $s = -5$ and $t = 12$.

Problem 4.4.3. Find integers s and t such that $s \cdot 5 + t \cdot 2 = \gcd(5, 2)$.

Solution. Although it is easy to see that the greatest common divisor of 5 and 2 is 1, we need some of the intermediate result from the Euclidean algorithm to find s and t . Following the Euclidean algorithm (Algorithm 4.3.1) for the input values $a := 5$ and $b := 2$ we get:

step	r	a	b
<i>Input</i>		5	2
(1)	$5 \bmod 2 = 1$	2	1
(1)	$2 \bmod 1 = 0$	1	0
<i>Output</i>		1	

We have confirmed that $\gcd(5, 2) = 1$. Since the Euclidean algorithm terminated after 2 iterations we can use the same trick as in Example 4.4.2. We get

$$r := 5 \bmod 2 = 1$$

and

$$q := 5 \operatorname{div} 2 = 2$$

Plugging these into the formula

$$(1 \cdot a) + ((-q) \cdot b) = r$$

we get

$$(1 \cdot 5) + ((-2) \cdot 2) = 1.$$

We read off the values $s := 1$ and $t := -2$. Note that $t = -(5 \operatorname{div} 2)$.

The observation made at the end of the last example can be generalized. We obtain the following theorem.

Theorem 4.4.4. *Let a and b be natural numbers. If the Euclidean algorithm for computing the greatest common divisor of a and b returns $\gcd(a, b)$ after only running through the **repeat-until** loop twice then $s \cdot a + t \cdot b = \gcd(a, b)$ with $s = 1$ and $t = -(a \operatorname{div} b)$.*

Problem 4.4.5. *For $a = 63$ and $b = 14$ find integers s and t such that $s \cdot a + t \cdot b = \gcd(a, b)$.*

Solution. We find the greatest common divisor of 63 and 14 using the Euclidean Algorithm.

$$(1) \quad 63 \bmod 14 = 7$$

$$(1) \quad 14 \bmod 7 = 0$$

So the Euclidean Algorithm ends after running through the loop twice and returns $\gcd(63, 14) = 7$. By Theorem 4.4.4 we have $s = 1$ and $t = -(63 \operatorname{div} 14) = -4$.

We check whether the result is correct:

$$1 \cdot 63 + (-4) \cdot 14 = 63 + (-56) = 7.$$

Part II

Sets and Functions

Sets are one of the fundamental structures in mathematics. We present the basic notation and definitions for working with sets, including the important notion of the equality of sets, in Chapter 5. In Chapter 6 we introduce subsets and explain the construction of sets as Cartesian products of sets. Functions are another fundamental objects in mathematics. Functions assign each element in one set to an element in another set. Often they are used to change the representation of objects. We investigate properties such as equality and invertibility of functions and combine functions to obtain new functions (Chapter 7). In Chapter 8 we apply functions to the encoding of characters into numbers and to the encryption of text.

Chapter 5

Sets

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Recognize whether a collection of objects is well-defined.
- (2) Recognize whether an element is in a set.
- (3) Rewrite a given set in roster form.
- (4) Recognize when two sets are equal.

We talk about sets every day – a set of books, a set of dishes, a set of rules. In mathematics, we often need to establish which objects we are using for a particular problem. Although we avoided using the term *set*, we have already discussed an important mathematical set – the set of integers.

We start by formally defining the term set and giving different ways of specifying sets. Furthermore we introduce notation for indicating membership in a set and saying when two sets are equal. We also present symbols for special sets such as the empty set and the set of integers.

5.1 Definition of a Set

We introduce sets as collections of objects. However, not all descriptions of a collection of objects are necessarily interpreted in the same way by everyone. For example, “the last four letters of the alphabet” could be interpreted differently by speakers of different languages. So, a more precise way to describe the collection of the letters **w**, **x**, **y**, and **z** might be “the last four letters of the English alphabet.” To distinguish letters from variables we write them in a typewriter font.

When there is no such ambiguity in the description of an object, then we say it is well-defined. We extend this to collections of objects and call them *well-defined* if their contents can be clearly determined.

Example 5.1.1. We give descriptions of collections of objects that are well-defined or not well-defined.

- (i) The collection of Greek letters is well-defined.
- (ii) The collection of South American countries is well-defined.
- (iii) The collection of cute animals is not well-defined. The characteristics that make an animal “cute” are a matter of opinion.
- (iv) The collection of best math teachers is not well-defined. The meaning of the word “best” here is up for interpretation.

Definition 5.1.2. A *set* is a well-defined collection of distinct objects. The objects in a set are called *elements* of the set.

When we use variables as placeholders for sets, we often use capital letters such as A or B . Sets may be described in various ways. For example, the set consisting of the letters w, x, y , and z might also be described as the set consisting of the last four letters of the English alphabet. So far, we have indicated sets by giving a verbal description of the contents of the set. Two additional methods we will use to indicate a set are *roster form* and *set-builder notation*.

5.2 Roster Form

Definition 5.2.1. The contents of a set can be described by listing the elements of the set, separated by commas, inside a set of curly brackets. This way of describing a set is called *roster form*.

Example 5.2.2. We give examples of sets in roster form.

- (i) $\{1, 2, 3, 4\}$ is the set containing the numbers 1, 2, 3, and 4.
- (ii) $\{w, x, y, z\}$ is the set containing the letters w, x, y , and z .
- (iii) $\{\text{red, yellow, blue}\}$ is the set containing red, yellow, and blue.
- (iv) $\{6\}$ is the set containing the number 6.
- (v) $\{3, -3, 11\}$ is the set containing the numbers 3, -3, and 11.
- (vi) $\{5, 3, w\}$ is the set containing the numbers 5 and 3 and the letter w .

Recall that an ellipsis (...) indicates that the pattern is continued. We can use an ellipsis when writing a set in roster form instead of listing every element.

Example 5.2.3. We give examples of sets written in roster form that use ellipses.

- (i) $\{1, 2, 3, \dots, 100\}$ is the set of integers from 1 to 100.
- (ii) $\{2, 3, 4, \dots, 99\}$ is the set of integers from 2 to 99.
- (iii) $\{c, d, e, \dots, n\}$ is the set of letters from c to n .

Roster form also allows us to formulate a set that does not contain any elements by writing $\{\}$.

Definition 5.2.4. The *empty set* (also called the *null set*) is the set that consists of no elements. It is denoted by $\{\}$.

The empty set contains no elements. In particular it does not contain the number 0.

5.3 Membership and Equality

The basic relationship between a set and an object is whether or not the object is an element of the set. We can only ask whether an element is in a set or not. There is also no ordering of the elements in the set.

Definition 5.3.1. The symbol $\in$, read as “is an element of” or “*is in*,” indicates membership in a set. The symbol $\notin$, read as “is not an element of” or “is not in,” indicates lack of membership in a set.

Example 5.3.2. We give examples of how to read the symbols $\in$ and $\notin$.

- (i) $3 \in \{1, 2, 3, 4\}$ is read as “3 is an element of the set containing 1, 2, 3, and 4” or “3 is in the set containing 1, 2, 3, and 4.”
- (ii) $5 \notin \{1, 2, 3, 4\}$ is read as “5 is not an element of the set $\{1, 2, 3, 4\}$ ” or “5 is not in the set $\{1, 2, 3, 4\}$.”
- (iii) $y \notin \{a, e, i, o, u\}$ is read as “y is not an element of the set containing a, e, i, o, and u.”

Definition 5.3.3. Two sets A and B are *equal* if each element in A is in B and if each element in B is in A . If two sets A and B are equal, we write $A = B$. If two sets A and B are not equal, we write $A \neq B$.

Example 5.3.4. We give examples of the correct usage of the symbols $=$ and $\neq$.

- (i) $\{1, 2, 3\} = \{2, 1, 3\}$, as each element, namely 1, 2, and 3, of $\{1, 2, 3\}$ is in $\{2, 1, 3\}$ and vice versa. The order in which the elements are listed in roster form does not change the set.
- (ii) $\{1, 2\} \neq \{1, 2, 3\}$, as 3 is not in $\{1, 2\}$.
- (iii) $\{a, b, c\} \neq \{1, 2, 3\}$, as a is not in $\{1, 2, 3\}$.
- (iv) $\{1, 2, \{3, 4\}\} \neq \{1, 2, 3, 4\}$, as the element $\{3, 4\}$ of $\{1, 2, \{3, 4\}\}$ is not contained in $\{1, 2, 3, 4\}$.
- (v) $\{\} \neq \{1\}$ as the number 1 is not contained in the empty set.

Problem 5.3.5. Let $C := \{1, 3, 5, 6\}$. For each statement indicate whether it is true or false.

- (i) $\{3\} = C$
- (ii) $C = \{6\}$
- (iii) $\{5, 3, 1, 6\} = C$
- (iv) $\{5\} \neq C$

Solution. Recall that two sets are equal if all elements in the first set are in the second set and if all elements of the second set are in the first set.

- (i) The number 1 is in the set C on the right but not in the set $\{3\}$ on the left. So $\{3\}$ is not equal to C ; the statement is false.
- (ii) The number 1 is in the set C on the left but not in the set $\{6\}$ on the right. So C is not equal to $\{6\}$; the statement is false.
- (iii) We first check whether every element of the set $\{5, 3, 1, 6\}$ is in the set C on the right.

The number 5 is in the set C .

The number 3 is in the set C .

The number 1 is in the set C .

The number 6 is in the set C .

Now we are halfway done. Next we check whether every element of $C = \{1, 3, 5, 6\}$ is in $\{5, 3, 1, 6\}$.

The number 1 is in the set $\{1, 3, 5, 6\}$.
The number 3 is in the set $\{1, 3, 5, 6\}$.
The number 5 is in the set $\{1, 3, 5, 6\}$.
The number 6 is in the set $\{1, 3, 5, 6\}$.

We conclude that $\{5, 3, 1, 6\} = C$. So the statement is true.

- (iv) The number 1 is in the set C on the right but not in the set $\{6\}$ on the left. So $\{5\}$ is not equal to C , in symbols: $\{5\} \neq C$. The statement is true.

5.4 Special Sets

Next, we introduce notation for some sets that we will use throughout this course. To make the notation unique and recognizable, we denote some special sets using specific capital letters $\mathbb{A}$, $\mathbb{N}$, $\mathbb{P}$, $\mathbb{W}$, and $\mathbb{Z}$ in a font called *blackboard bold*.

Definition 5.4.1. We define the following sets:

- (i) The set $\mathbb{Z} := \{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$ is the set of *integers*.
- (ii) The set $\mathbb{N} := \{1, 2, 3, \dots\}$ is the set of *natural numbers*.
- (iii) The set $\mathbb{P}$ is the set of prime numbers.
- (iv) The set $\mathbb{W} = \{0, 1, 2, 3, \dots\}$ is the set of *whole numbers*.
- (v) For $n \in \mathbb{N}$ we define $\mathbb{Z}_n := \{0, 1, 2, \dots, n-1\}$. We read $\mathbb{Z}_n$ as “z n.”
- (vi) For $n \in \mathbb{N}$ we define $\mathbb{Z}_n^\otimes := \{1, 2, \dots, n-1\}$. We read $\mathbb{Z}_n^\otimes$ as “z n without zero.”
- (vii) The set $\mathbb{A} := \{-, \mathbf{a}, \mathbf{b}, \mathbf{c}, \dots, \mathbf{x}, \mathbf{y}, \mathbf{z}\}$ is the set of *characters*¹.

Example 5.4.2. We give an example of the sets $\mathbb{Z}_n$ and $\mathbb{Z}_n^\otimes$, where $n = 7$.

- (i) $\mathbb{Z}_7 = \{0, 1, 2, 3, 4, 5, 6\}$ is read “z 7 is equal to the set containing 0,1,2,3,4,5, and 6.”
- (ii) $\mathbb{Z}_7^\otimes = \{1, 2, 3, 4, 5, 6\}$ is read “z 7 without 0 is equal to the set containing 1,2,3,4,5, and 6.”

Example 5.4.3. It is always interesting to try out the extreme cases of a definition. For $\mathbb{Z}_n$ and $\mathbb{Z}_n^\otimes$ and $n \in \{1, 2\}$ (this means we will look at the cases when $n = 1$ and when $n = 2$):

- (i) $\mathbb{Z}_1 = \{0\}$
- (ii) $\mathbb{Z}_1^\otimes = \{\}$
- (iii) $\mathbb{Z}_2 = \{0, 1\}$
- (iv) $\mathbb{Z}_2^\otimes = \{1\}$

5.4.1 Formulating Statements with Sets

We can use set notation and the special sets defined above to give shorter formulations of statements from Section 1.3. Essentially we are replacing “let a be an integer” by “let $a \in \mathbb{Z}$ ”.

Example 5.4.4. For all $n \in \mathbb{N}$ we have $n > 0$. (compare Problem 1.3.9)

¹for technical reasons we use the symbol -instead of the character space to separate words

The commutative property of addition for integers from Example 1.3.11 becomes:

Example 5.4.5. For all $a \in \mathbb{Z}$ and $b \in \mathbb{Z}$ we have $a + b = b + a$.

Likewise the distributive property (compare Example 1.3.12) can be written as:

Example 5.4.6. For all $a \in \mathbb{Z}$, $b \in \mathbb{Z}$, and $c \in \mathbb{Z}$ we have $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$.

Theorem 1.3.20 states that for all integers there exists an additive inverse. We can write this theorem as:

Theorem 5.4.7. For all $a \in \mathbb{Z}$ there is a $b \in \mathbb{Z}$ such that $a + b = 0$.

Finally we reformulate Theorems 1.4.5 and 1.4.7 using set notation

Theorem 5.4.8. Let $a \in \mathbb{Z}$ and $b \in \mathbb{Z}$ and let $m \in \mathbb{N}$ and $n \in \mathbb{N}$. Then

- (i) $(b^m) \cdot (b^n) = b^{(m+n)}$
- (ii) $(b^m)^n = b^{(m \cdot n)}$
- (iii) $(a \cdot b)^n = (a^n) \cdot (b^n)$

5.5 Set-Builder Notation

Set-builder notation can be used to specify a set by describing the properties of its elements. In set-builder notation we write sets in the form

$$\{x \mid (\text{properties of } x)\},$$

where (properties of x) is replaced by conditions that fully describe the elements of the set. The bar ($\mid$) is used to separate the elements and properties. The bar is read as “such that,” and all together we read this set as “the set of all elements x such that (properties of x).” We use a variable (here x) to formulate the properties on the elements in the set.

Example 5.5.1. We read $A = \{x \mid x \in \mathbb{N} \text{ and } x < 6\}$ as “ A is equal to the set of all elements x such that x is a natural number and x is less than 6.” Notice how we use x to formulate the properties of the elements in the set. In roster form we write $A = \{1, 2, 3, 4, 5\}$.

Example 5.5.2. There are many ways of describing the same set using set-builder notation:

- (i) $\{x \mid x \text{ is a natural number from 4 to 8}\} = \{4, 5, 6, 7, 8\}$
- (ii) $\{x \mid x \in \mathbb{N} \text{ and } x > 3 \text{ and } x < 9\} = \{4, 5, 6, 7, 8\}$
- (iii) $\{x \mid x \in \mathbb{N} \text{ and } x \geq 4 \text{ and } x \leq 8\} = \{4, 5, 6, 7, 8\}$

Example 5.5.3. We formulate some familiar sets in set-builder notation.

- (i) $\mathbb{N} = \{x \mid x \in \mathbb{Z} \text{ and } x > 0\}$ is the set of natural numbers.
- (ii) $\{x \mid x \in \mathbb{N} \text{ and } x \bmod 2 = 0\} = \{2, 4, 6, 8, \dots\}$ is the set of even natural numbers.
- (iii) $\{x \mid x \in \mathbb{N} \text{ and } x \bmod 2 = 1\} = \{1, 3, 5, 7, \dots\}$ is the set of odd natural numbers.

(iv) $\{\}$ = $\{x \mid x \in \mathbb{N} \text{ and } x < 0\}$ as there are no natural numbers that are less than zero.

Example 5.5.4. Let $m \in \mathbb{N}$. We give special sets from the previous section in set builder notation.

(i) $\mathbb{Z}_m = \{x \mid x \in \mathbb{Z} \text{ and } x \geq 0 \text{ and } x < m\}$.

(ii) $\mathbb{Z}_m^\otimes = \{x \mid x \in \mathbb{Z} \text{ and } x > 0 \text{ and } x < m\}$.

Chapter 6

More on Sets

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Recognize whether a set is a subset of another set.
- (2) List the elements of a Cartesian product.
- (3) Recognize whether an element is in a Cartesian product.
- (4) Rewrite an image as a Cartesian product.
- (5) Convert a Cartesian product into an image.

Now that we have defined sets, it is important to be able to identify a basic relationship between sets. This is the idea of a subset. We also introduce a way of obtaining a new set from existing sets by constructing the Cartesian product of the sets.

6.1 Subsets

It is often helpful to break down large sets into smaller, more manageable sets. We introduce relations that allow us to formulate statements about the containment of the elements of one set in another set. The subset relation allows us to compare sets beyond only equality.

Definition 6.1.1. A set A is a *subset* of a set B if each element in A is also an element in B . If A is a subset of B , we write $A \subseteq B$. If there is at least one element in A that is not an element in B , then A is not a subset of B . If A is not a subset of B , we write $A \not\subseteq B$.

We read $A \subseteq B$ as “ A is a subset of B ” and $A \not\subseteq B$ as “ A is not a subset of B .”

Example 6.1.2. We give some examples for the use of the relations $\subseteq$ and $\not\subseteq$.

- (i) $\{1, 2\} \subseteq \{1, 2, 4, 9\}$
- (ii) $\{1, 2\} \subseteq \mathbb{N}$
- (iii) $\{1, 2\} \not\subseteq \{1, 3, 4, 9\}$
- (iv) $\{2\} \subseteq \{2, 3\}$
- (v) $\{2, 3\} \subseteq \{2, 3\}$

The relations $\in$ and $\subseteq$ may seem similar, but we have to consider that $\subseteq$ compares two sets while $\in$ is used to express that an element is in a set. So we cannot write $3 \subseteq \{1, 2, 3\}$ or $3 \not\subseteq \{1, 2, 3\}$ because 3 is not a set.

Example 6.1.3. We give some examples for the use of the relations $\in$, $\notin$, $\subseteq$, and $\not\subseteq$.

- (i) $3 \in \{1, 2, 3\}$, as the number 3 is in the set containing the numbers 1, 2, and 3.

- (ii) $\{3\} \subseteq \{1, 2, 3\}$, as each element, namely the number 3, of the set $\{3\}$ is in the set containing the numbers 1, 2, and 3.
- (iii) $\{3\} \not\subseteq \{1, 2, 3\}$, as the set $\{3\}$ is not in the set containing the numbers 1, 2, and 3.
- (iv) $\{3\} \not\subseteq \{\{1\}, \{2\}, \{3\}\}$, as the number 3 is not element of the set containing the sets $\{1\}$, $\{2\}$, and $\{3\}$.
- (v) $\{1, 2\} \subseteq \{1, 2, 3, 4\}$, as the numbers 1 and 2 are in the set containing the numbers 1 and 2 and 3 and 4.

The empty set $\{\}$ does not contain any elements. So when checking whether the empty set is a subset of another set, we do not have any elements to check. So it is true that each element in $\{\}$ is also an element of any other set. This means that the empty set is a subset of every set.

Theorem 6.1.4. *For all sets A we have: $\{\} \subseteq A$.*

Example 6.1.5. We give examples of subset relations involving the empty set.

- (i) $\{\} \subseteq \{2, 3\}$
- (ii) $\{\} \subseteq \{\}$

For any set A each element in A is also an element of A .

Theorem 6.1.6. *For all sets A we have: $A \subseteq A$.*

Furthermore, if two sets are both subsets of each other, they contain the same elements and hence are equal.

Theorem 6.1.7. *For all sets A and B we have: If $A \subseteq B$ and $B \subseteq A$ then $A = B$.*

6.2 Cartesian Products

A Cartesian product of two sets is a new set that is constructed from the two sets. In order to define Cartesian products, we need to define a mathematical object called an ordered pair.

Definition 6.2.1. An *ordered pair* is an ordered list of two mathematical objects, a and b , written as (a, b) . The objects in an ordered pair are called *components*. The object a is the first component of (a, b) , and the object b is the second component of (a, b) .

Definition 6.2.2. Let A and B be sets. The *Cartesian product* of A and B , denoted $A \times B$, is the set of *ordered pairs* (a, b) , where $a \in A$ and $b \in B$.

The Cartesian product of two sets A and B , formulated in set-builder notation, is

$$A \times B := \{(a, b) \mid a \in A \text{ and } b \in B\}.$$

To form the Cartesian product $A \times B$, we pair each element of A , placed in the first component of the ordered pair, with each element of B , placed in the second component of the ordered pair.

Example 6.2.3. Let $A = \{0, 1\}$, and let $B = \{4, 5, 6\}$. Then,

$$A \times B = \{(0, 4), (0, 5), (0, 6), (1, 4), (1, 5), (1, 6)\},$$

and

$$B \times A = \{(4, 0), (4, 1), (5, 0), (5, 1), (6, 0), (6, 1)\}.$$

Problem 6.2.4. Let $A = \{1, 2, 3\}$ and let $B = \{-50\}$. Give the set $A \times B$ in roster form.

Solution. The set $A \times B$ contains all ordered pairs whose first entry is an element of the set A and whose second entry is an element of the set B . We write ordered pairs whose first entry is c and whose second entry is d as (c, d) . We get

$$A \times B = \{(1, -50), (2, -50), (3, -50)\}$$

In the next problem a Cartesian product is given in set builder notation.

Problem 6.2.5. Let $A = \{12, 13, 34\}$. Give $\{(a, a \bmod 5) \mid a \in A\}$ in roster form.

Solution. We find all pairs whose first entry is an element a of the set A and whose second entry is $a \bmod 5$. We get

$$\begin{aligned} \{(a, a \bmod 5) \mid a \in A\} &= \{(12, 12 \bmod 5), (13, 13 \bmod 5), (34, 34 \bmod 5)\} \\ &= \{(12, 2), (13, 3), (34, 4)\}. \end{aligned}$$

Definition 6.2.6. Let A and B be sets. Saying that $(a, b) \in A \times B$ and $(c, d) \in A \times B$ are *equal* means that $a = c$ and $b = d$. If (a, b) and (c, d) are equal, we write $(a, b) = (c, d)$.

So, two ordered pairs are equal if they have matching first components and matching second components. The fact that the elements (a, b) of $A \times B$ are called *ordered pairs* indicates that we must pay attention to order for Cartesian products. In comparison, recall that the order of the elements in a set given in roster form does not matter. (See Example 5.3.4.)

Example 6.2.7. As sets, $\{1, 2\} = \{2, 1\}$. However, as ordered pairs, $(1, 2) \neq (2, 1)$.

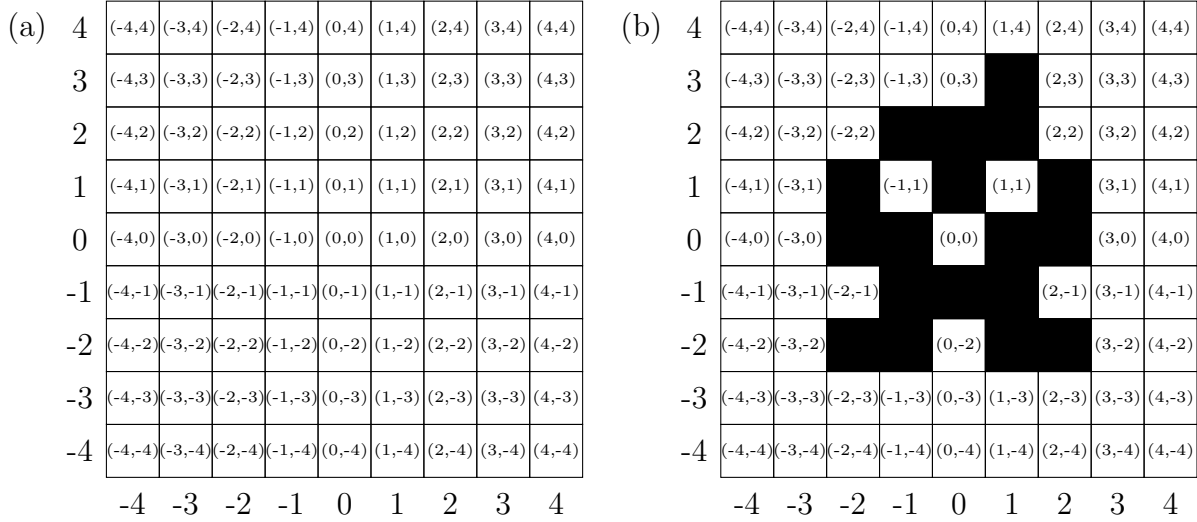
Since the empty set $\{\}$ does not contain any elements, there are no elements to be placed into the second component of the Cartesian product $A \times \{\}$. So, we have that $A \times \{\} = \{\}$ for any set A . Similarly, $\{\} \times B = \{\}$ for any set B .

6.3 Applications of Cartesian Products

We can visualize a Cartesian product of two sets as a *raster* – a rectangular pattern of points. In Figure 6.3.2 we represent the set $\{0, 1, 2\} \times \{0, 1, 2, 3, 4\}$ in this way.

There are many applications of such a representation of Cartesian products, or rather many real life objects can be represented by Cartesian products.

Figure 6.3.1: The sets from Example 6.3.2. (a) the set G as a raster of pixels and (b) the subset I of G as black pixels that produce an image in the raster



Example 6.3.1. The squares on a chess board are represented by elements of the Cartesian product $\{a, b, c, d, e, f, g, h\} \times \{1, 2, 3, 4, 5, 6, 7, 8\}$.

Most computers display images as a raster of points called *pixels* that can be addressed by their coordinates. These coordinates are ordered pairs and hence elements of a Cartesian product. We represent an image by coloring in the points that correspond to elements of a subset of a Cartesian product in the raster that represents the Cartesian product.

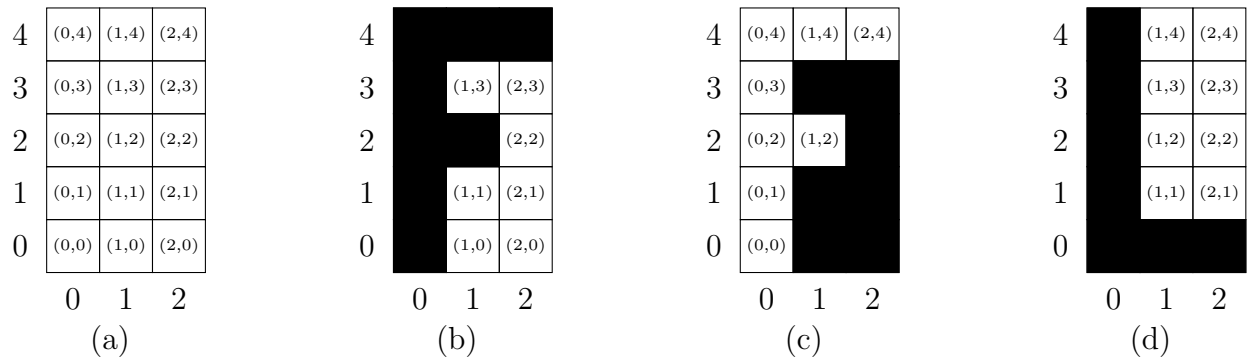
Example 6.3.2. Figure 6.3.1 (a) is a graphical representation of the Cartesian product $G = \{-4, \dots, 4\} \times \{-4, \dots, 4\}$ as a raster of rectangles, called pixels, with one pixel for each element of G .

Figure 6.3.1 (b) is an example of an image that could be displayed on a computer screen. The image of the “alien” is formed by black pixels in the raster. Let $I := \{(1, 3), (-1, 2), (0, 2), (1, 2), (-2, 1), (0, 1), (2, 1), (-2, 0), (-1, 0), (1, 0), (2, 0), (-1, -1), (0, -1), (1, -1), (-2, -2), (-1, -2), (1, -2), (2, -2)\}$. Then, the subset I of G defines the set of black pixels that forms the image in the raster.

Example 6.3.3. In Figure 6.3.2, we represent the set $G = \{0, 1, 2\} \times \{0, 1, 2, 3, 4\}$ as a raster with the elements of various subsets given in black.

- We start with the raster representing the set G .
- Let $F := \{(0, 4), (1, 4), (2, 4), (0, 3), (0, 2), (1, 2), (0, 1), (0, 0)\}$. Then, the subset F of G forms a picture of the letter F.
- Let the set H consist of all elements of G that are not in F . This is the set of pairs $H := \{(1, 0), (2, 0), (1, 1), (2, 1), (2, 2), (1, 3), (2, 3)\}$.
- Let $L := \{(0, 4), (0, 3), (0, 2), (0, 1), (0, 0), (1, 0), (2, 0)\}$. Then, the subset L of G looks like the letter L.

Figure 6.3.2: Graphical representation of the sets from Example 6.3.3



Chapter 7

Functions

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Evaluate functions.
- (2) Compute the image of a function.
- (3) Demonstrate that two functions are equal.
- (4) Evaluate the composite of two functions.
- (5) Recognize the identity function on a set.
- (6) Recognize whether a function is invertible or not.
- (7) Demonstrate that a function is the inverse of another function.

Just like sets, functions are a basic and important idea in mathematics. Functions give us a tool for manipulating numbers and other data and for switching between different representations of objects.

In this section we introduce functions and their properties. We encounter some special functions, create new functions as composites of other functions, and consider invertible functions, which we will use in the following sections to switch between different representations of data.

7.1 Definition of a Function

A function has three parts, a set of inputs, a set of outputs, and a rule that relates the elements of the set of inputs to the elements of the set of outputs in such a way that each input is assigned exactly one output.

Although for brevity, functions are often identified by a one-letter name, such as f , many common functions are identified by multi-letter names. We will see that gcd that we saw in the previous chapter is actually an example of a function. Many others you can find on the keys of your calculator, for example: cos, exp, ln, log, sin, tan, and so on.

Definition 7.1.1. Let A and B be nonempty sets.

- A *function* f from A to B assigns exactly one element of B to each element of A . We denote a function f from A to B by $f: A \rightarrow B$ and we write $f(a) = b$ if b is the unique element of B that is assigned to the element $a \in A$ by f .
- We read $f: A \rightarrow B$ as “the function f from A to B .” We read $f(a) = b$ as “ f of a is b ” or “ f evaluated at a is b .”
- The set A is called the *domain* of f , and the set B is called the *codomain* of f .
- Suppose that $f(a) = b$. Then the element b is the *image* of the element a under the function f , and the element a is a *preimage* of the element b under the function f .

Figure 7.1.1: The function $\text{studentid} : N \rightarrow I$ where $N = \{\text{Aaron, Alice, Bob, Eve, James, Nathan, Oscar, Sandi}\}$ is the set of students in MAT 112 and $I = \{1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008\}$ the set of student identification numbers defined by a table.

n	$\text{studentid}(n)$
Aaron	1006
Alice	1001
Bob	1002
Eve	1003
James	1005
Nathan	1007
Oscar	1004
Sandi	1008

Example 7.1.2. Let $N = \{\text{Aaron, Alice, Bob, Eve, James, Nathan, Oscar, Sandi}\}$ be the set of students in MAT 112 and $I = \{1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008\}$ the set of student identification numbers.

In Figure 7.1.1 we define a function $\text{studentid} : N \rightarrow I$ that assigns an identification number in the set I to each student in the set S . The set N of student names is the domain of the function studentid and the set I of student identification numbers is the codomain of studentid .

We have $\text{studentid}(\text{Alice}) = 1001$. So 1001 is the image of Alice under the function studentid . Alice is the preimage of 1001 under the function studentid .

Example 7.1.3. Let $I = \{1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008\}$ be the identification numbers of the students in MAT 112. The teacher of MAT 112 posts the table from Figure 7.1.2 on her door, so that the students can look up their grades their without posting the students names.

The set I of student identification numbers is the domain of the function grade and the set $G = \{A, B, C, D, F\}$ is the codomain of grade .

We have $\text{grade}(1001) = B$. Which means that the grade of the student with the identification number 1001 is B . So the image of 1001 under the function grade is B and 1001 is a preimage of B under the function grade .

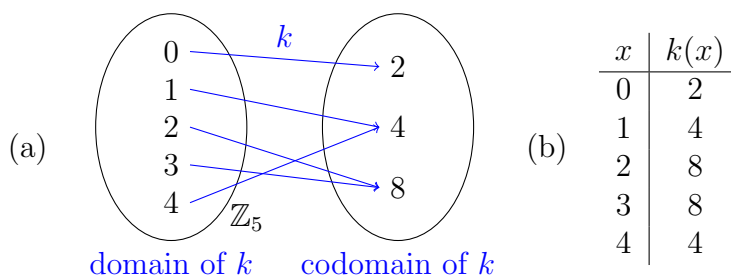
The images of the elements under a function can be specified in different ways. If we can write down all of the elements of the domain of a function, the function can be specified by explicitly giving the images of the elements of the domain. This can, for example, be done with a diagram or a table.

Example 7.1.4. The function k in Figure 7.1.3 is given by a diagram in (a) and by a table in (b).

Figure 7.1.2: The function $\text{grade} : I \rightarrow G$ where $I = \{1001, 1002, 1003, 1004, 1005, 1006, 1007, 1008\}$ is the set of student identification numbers and $G := \{A, B, C, D, F\}$ is the set of possible grades.

i	$\text{grade}(i)$
1001	B
1002	C
1003	D
1004	F
1005	A
1006	A
1007	A
1008	A

Figure 7.1.3: Two ways of specifying the function k from the set $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$ to the set $\{2, 4, 8\}$ used in Example 7.1.4: (a) by a diagram and (b) by a table



Functions can also be specified by an algebraic rule. For a function f , we may specify that $f(x)$ is equal to an algebraic expression in the variable x .

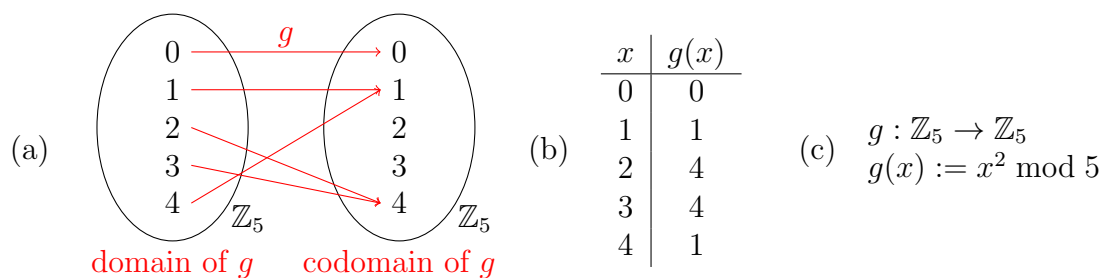
Example 7.1.5. The function g in Figure 7.1.4 is given by a diagram in (a), by a table in (b), and by an algebraic rule in (c).

Example 7.1.6. The function $s : \mathbb{N} \rightarrow \mathbb{N}$ given by $s(n) := n^2$ is the function that assigns to each natural number n its square n^2 . (Note that we are using $:=$ because we are defining the output of the function s .)

We have $s(1) = 1^2 = 1$, $s(2) = 2^2 = 4$, $s(3) = 3^2 = 9$ and so on. Thus the image of 1 is 1, the image of 2 is 4, and the image of 3 is 9. A preimage of 1 is 1, a preimage of 4 is 2, and a preimage of 9 is 3. The number 2 does not have a preimage, since it is not a square of a natural number.

We give an example of a function, under which each element in the codomain has (infinitely) many preimages.

Figure 7.1.4: Three ways of specifying the function g used in 7.1.5 from the set $\mathbb{Z}_5$ to the set $\mathbb{Z}_5$: (a) by a diagram, (b) by a table, and (c) by an algebraic rule.



Example 7.1.7. Let $\mathbb{N}$ be the set of natural numbers and $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$. Consider the function $m : \mathbb{N} \rightarrow \mathbb{Z}_5$ given by $m(a) := a \bmod 5$. We have $m(1) = 1$, $m(2) = 2$, $m(3) = 3$, $m(4) = 4$, $m(5) = 0$, $m(6) = 1$, and so on.

In the table below, we explicitly give the images under m for a few elements in the domain.

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	...
$m(a)$	1	2	3	4	0	1	2	3	4	0	1	2	3	4	0	1	2	3	4	...

Notice that the image of 1 is 1. But the number 1 has many preimages. For example, the elements 1, 6, 11, and 16 in $\mathbb{N}$ are all preimages of the element 1 in $\mathbb{Z}_5$. There are infinitely many preimages of 1, namely all numbers in the set $\{a \mid a \in \mathbb{N} \text{ and } a \bmod 5 = 1\}$.

A function can also be described by an algorithm.

Example 7.1.8. The greatest common divisor function takes each pair of natural numbers and assigns to it the natural number that is the greatest common divisor of the two numbers in the pair. So, the greatest common divisor function has domain $\mathbb{N} \times \mathbb{N}$ and codomain $\mathbb{N}$, and we may write

$$\text{gcd} : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}.$$

The function gcd is explicitly specified by the Euclidean Algorithm that has a pair of natural numbers as the input and provides a single natural number as the output.

7.2 Graphs of Functions

In Definition ?? we had introduced the image of an element of the domain under a function. For x in the domain of a function f we call $f(x)$ the image of x under f . The set of all these images is called the image of the function.

Definition 7.2.1 (Image of a function). The *image of a function* $f : A \rightarrow B$ is

$$f(A) = \{f(x) \mid x \in A\}.$$

For all x in the domain of $f : A \rightarrow B$ we have $f(x) \in B$, thus the image $f(A)$ of f is a subset of codomain B of f .

Example 7.2.2. Let $f : \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ be given by $f(x) = x^2 \bmod 5$. Then

$$\begin{aligned} f(0) &= 0^2 \bmod 5 = 0 \bmod 5 = 0 \\ f(1) &= 1^2 \bmod 5 = 1 \bmod 5 = 1 \\ f(2) &= 2^2 \bmod 5 = 4 \bmod 5 = 4 \\ f(3) &= 3^2 \bmod 5 = 9 \bmod 5 = 4 \\ f(4) &= 4^2 \bmod 5 = 16 \bmod 5 = 1 \end{aligned}$$

Thus the image $f(\mathbb{Z}_5)$ of f is

$$f(\mathbb{Z}_5) = \{f(x) \mid x \in \mathbb{Z}_5\} = \{0, 1, 4\}.$$

Note that this differs from the codomain of f which is $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$.

In Section 7.1 above we defined functions by algebraic expressions or tables of charts. Another way of representing a function is its graph. Instead of organizing the values in a table we consider them as elements of a Cartesian product.

Definition 7.2.3 (Graph of function). The *graph of a function* $f : A \rightarrow B$ is

$$\{(x, f(x)) \mid x \in A\} \subseteq A \times B.$$

Example 7.2.4. Let $f : \mathbb{Z}_6 \rightarrow \mathbb{Z}_5$ be given by $f(x) = 2^x \bmod 5$.

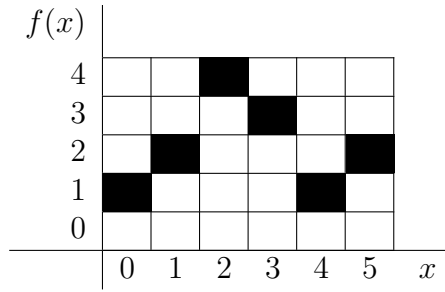
We have

$$\begin{aligned} f(0) &= 2^0 \bmod 5 = 1 \bmod 5 = 1 \\ f(1) &= 2^1 \bmod 5 = 2 \bmod 5 = 2 \\ f(2) &= 2^2 \bmod 5 = 4 \bmod 5 = 4 \\ f(3) &= 2^3 \bmod 5 = 8 \bmod 5 = 3 \\ f(4) &= 2^4 \bmod 5 = 16 \bmod 5 = 1 \\ f(5) &= 2^5 \bmod 5 = 32 \bmod 5 = 2 \end{aligned}$$

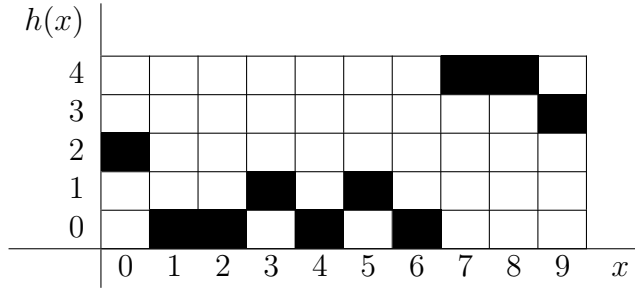
Thus the graph of f is:

$$\begin{aligned} \{(x, f(x)) \mid x \in \mathbb{Z}_6\} &= \{(0, f(0)), (1, f(1)), (2, f(2)), (3, f(3)), (4, f(4)), (5, f(5))\} \\ &= \{(0, 1), (1, 2), (2, 4), (3, 3), (4, 1), (5, 2)\} \subseteq \mathbb{Z}_6 \times \mathbb{Z}_5 \end{aligned}$$

The graphical representation of the graph of f as a subset of $\mathbb{Z}_6 \times \mathbb{Z}_5$ where black pixel represent the elements of the graph of f is:



Example 7.2.5. Suppose that the graph of the function h is given by



The values on the horizontal axis of the plot are the elements of domain of h . So the domain of h is $\mathbb{Z}_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. The codomain are the values on the vertical axis of the plot. Thus the codomain of h is $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$.

The graph of h are the elements of the Cartesian product $\mathbb{Z}_{10} \times \mathbb{Z}_5$ which are represented by the black pixels in the plot. We find that the graph of h is

$$\{(x, h(x)) \mid x \in A\} = \{(0, 2), (1, 0), (2, 0), (3, 1), (4, 0), (5, 1), (6, 0), (7, 4), (8, 4), (9, 3)\}.$$

Because the graph of h consists of the pairs $(x, h(x))$ where x is an element of the domain of h , we can read off the values $h(x)$ easily. We get

$$\begin{aligned} h(0) = 2, h(1) = 0, h(2) = 0, h(3) = 1, h(4) = 0, \\ h(5) = 1, h(6) = 0, h(7) = 4, h(8) = 4, h(9) = 0 \end{aligned}$$

These values can also be directly read of the plot by finding the vertical coordinate of the black pixel in the column of each value on the horizontal axis.

7.3 Equality of Functions

Two functions are equal if they have the same domain and codomain and their values are the same for all elements of the domain.

Definition 7.3.1. Let A and B be sets and $f : A \rightarrow B$ and $g : A \rightarrow B$ be functions. We say that f and g are *equal* and write $f = g$ if $f(a) = g(a)$ for all $a \in A$. If f and g are not equal, we write $f \neq g$.

In our definition of the equality of functions, we have assumed that the two functions have the same domain and codomain. Two functions that do not have the same domain and codomain are not equal.

Example 7.3.2. Let $f : \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ be given by $f(a) := (a + 1) \bmod 5$ and $g : \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ be given by $g(a) := (a - 4) \bmod 5$. First, note that the domains of f and g are the same and the codomains of f and g are the same. We show that $f = g$ by evaluating both f and g at each element of the common domain $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$ and then comparing the function values for the same elements.

$$\begin{array}{ll} f(0) = (0 + 1) \bmod 5 = 1 \bmod 5 = 1 & g(0) = (0 - 4) \bmod 5 = (-4) \bmod 5 = 1 \\ f(1) = (1 + 1) \bmod 5 = 2 \bmod 5 = 2 & g(1) = (1 - 4) \bmod 5 = (-3) \bmod 5 = 2 \\ f(2) = (2 + 1) \bmod 5 = 3 \bmod 5 = 3 & g(2) = (2 - 4) \bmod 5 = (-2) \bmod 5 = 3 \\ f(3) = (3 + 1) \bmod 5 = 4 \bmod 5 = 4 & g(3) = (3 - 4) \bmod 5 = (-1) \bmod 5 = 4 \\ f(4) = (4 + 1) \bmod 5 = 5 \bmod 5 = 0 & g(4) = (4 - 4) \bmod 5 = 0 \bmod 5 = 0 \end{array}$$

Since $f(a) = g(a)$ for all $a \in \mathbb{Z}_5$, we have $f = g$.

Problem 7.3.3. *Decide whether the two functions*

$$\begin{array}{l} f : \mathbb{Z}_3 \rightarrow \mathbb{Z}_3 \text{ given by } f(x) = (x^2 + 1) \bmod 3 \\ g : \mathbb{Z}_3 \rightarrow \mathbb{Z}_3 \text{ given by } f(x) = (x - 2) \bmod 3 \end{array}$$

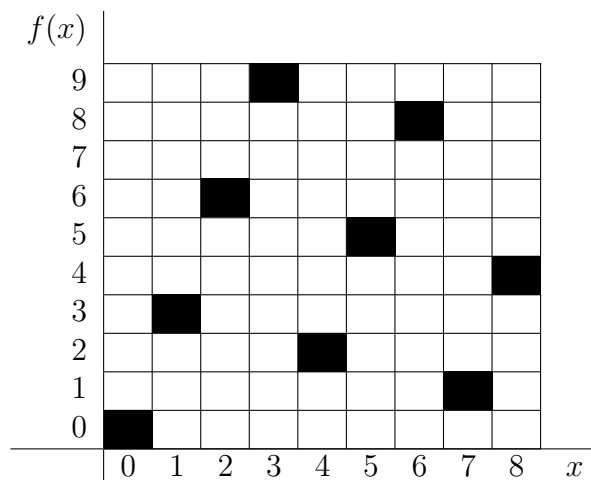
are equal.

Solution. We evaluate f and g at elements of their domain $\mathbb{Z}_3 = \{0, 1, 2\}$. If $f(x) = g(x)$ for all $x \in \mathbb{Z}_3$ then $f = g$.

$$\begin{array}{ll} f(0) = (0^2 + 1) \bmod 3 = 1 \bmod 3 = 1 & g(0) = (0 - 2) \bmod 3 = (-2) \bmod 3 = 1 \\ f(1) = (1^2 + 1) \bmod 3 = 2 \bmod 3 = 2 & g(1) = (1 - 2) \bmod 3 = (-1) \bmod 3 = 2 \\ f(2) = (2^2 + 1) \bmod 3 = 5 \bmod 3 = 2 & g(2) = (2 - 2) \bmod 3 = 0 \bmod 3 = 0 \end{array}$$

As $f(2) \neq g(2)$ the two functions f and g are not equal.

Example 7.3.4. Let the graph of a function f be given by



The graph immediately yields that the domain of f is $\mathbb{Z}_9$ and that the codomain of f is $\mathbb{Z}_{10}$. We read off the images of the elements of the domain f :

$$f(0) = 0, f(1) = 3, f(2) = 6, f(3) = 9, f(4) = 2, f(5) = 5, f(6) = 8, f(7) = 1, f(8) = 4$$

Let $g : \mathbb{Z}_9 \rightarrow \mathbb{Z}_{10}$ be given by $g(x) = 3 \cdot x \bmod 10$.

We now determine whether the functions f and g are equal. We have already found $f(x)$ for all $x \in \mathbb{Z}_9$. Now we compute $g(x)$ for all $x \in \mathbb{Z}_9$. We get:

$$g(0) = 0 \quad g(1) = 3 \quad g(2) = 6 \quad g(3) = 9 \quad g(4) = 2 \quad g(5) = 5 \quad g(6) = 8 \quad g(7) = 1 \quad g(8) = 4$$

Because f and g have the same domain and codomain and because $f(x) = g(x)$ for all $x \in \mathbb{Z}_9$ we conclude that the two functions f and g are equal.

7.4 Composite Functions

We combine two functions to get a new function by using function composition. Given two functions f and g we create a new function such that the image of a in the domain of f is $g(f(a))$. To compute $g(f(a))$ we first apply f to determine $f(a)$, and then apply g to the result. This only works if $f(a)$ is in the domain of g .

Definition 7.4.1. Let $f : A \rightarrow B$, and let $g : B \rightarrow C$. The *composite function* $g \circ f$, is the function $g \circ f : A \rightarrow C$ defined by

$$(g \circ f)(x) = g(f(x)).$$

We read $g \circ f$ as “the composite of (the functions) g and f .” We read $(g \circ f)(x)$ as “the composite of g and f of x ” or as “ g of f of x .”

We can soften the conditions on the domain and codomain of f and g by only requiring that the codomain of f is a subset of the domain of g .

Example 7.4.2. We use the functions $\text{studentid} : N \rightarrow I$ and $\text{grade} : I \rightarrow G$ from Examples 7.1.2 and 7.1.3 given by the tables in Figures 7.1.1 and 7.1.2 respectively.

To find the grade of a student, we first need to look up the student’s identification number in the table from Figure 7.1.1 and then with the identification number look up the grade in the table from Figure 7.1.1.

So to find Alice’s grade we first look up her identification number in Figure 7.1.1 and find that it is 1001. From Figure 7.1.2 we get that the grade of the student with identification number 1001 is a B . Thus Alice’s grade in MAT 112 is a B .

Now we formulate this process in terms of function composition. The composite function

$$\text{grade} \circ \text{studentid}$$

given a student’s name yields the student’s grade. The domain of $\text{grade} \circ \text{studentid}$ is the set of student names and the codomain of $\text{grade} \circ \text{studentid}$ is the set $G = \{A, B, C, D, F\}$ of grades. We get

$$(\text{grade} \circ \text{studentid})(\text{Alice}) = \text{grade}(\text{studentid}(\text{Alice})) = \text{grade}(1001) = B.$$

In Figure 7.4.1, we give an example of the composite of two functions that are given by a diagram.

Example 7.4.3. Let $s : \mathbb{N} \rightarrow \mathbb{N}$ be given by $s(n) := n^2$ as in Example 7.1.6, and let $m : \mathbb{N} \rightarrow \mathbb{Z}_5$ be given by $m(a) := a \bmod 5$ as in Example 7.1.7. The composite function $m \circ s$ is a function from $\mathbb{N}$ to $\mathbb{Z}_5$, and we have that $(m \circ s)(n) = m(s(n)) = m(n^2) = n^2 \bmod 5$ for each $n \in \mathbb{N}$. Notice that the algebraic rule for $m \circ s$ is the same as the algebraic rule for the function g in Figure 7.1.4. However, $m \circ s \neq g$ since the domain of $m \circ s$ is $\mathbb{N}$ and the domain of g is $\mathbb{Z}_5$.

The order in which the functions are composed matters, that is, there are functions f and g such that $g \circ f \neq f \circ g$.

Example 7.4.4. We show that the order of the composition of function matters. Let $f : \mathbb{N} \rightarrow \mathbb{N}$ given by $f(n) := 2 \cdot n$ and $g : \mathbb{N} \rightarrow \mathbb{N}$ given by $g(m) := m^2$. The domains of f and g allow us to form the composites $g \circ f$ and $f \circ g$. To show that $f \circ g$ is not equal to $g \circ f$ we only need to find an $b \in \mathbb{N}$ with $(g \circ f)(b) \neq (f \circ g)(b)$. For $b = 3$ we have

$$(g \circ f)(3) = g(f(3)) = g(2 \cdot 3) = g(6) = 6^2 = 36$$

and

$$(f \circ g)(3) = f(g(3)) = f(3^2) = f(9) = 2 \cdot 9 = 18,$$

So the functions $g \circ f$ and $f \circ g$ are not equal.

Problem 7.4.5. Consider the two functions

$$\begin{aligned} g : \{-1, 0, 1\} &\rightarrow \{0, 1, 2\} \text{ given by } g(x) := x^2, \text{ and} \\ h : \{0, 1, 2\} &\rightarrow \{2, 3, 4\} \text{ given by } h(y) := y + 2. \end{aligned}$$

- (i) Does the composite function $h \circ g$ exist? If yes, specify $h \circ g$.
- (ii) Does the composite function $g \circ h$ exist? If yes, specify $g \circ h$.

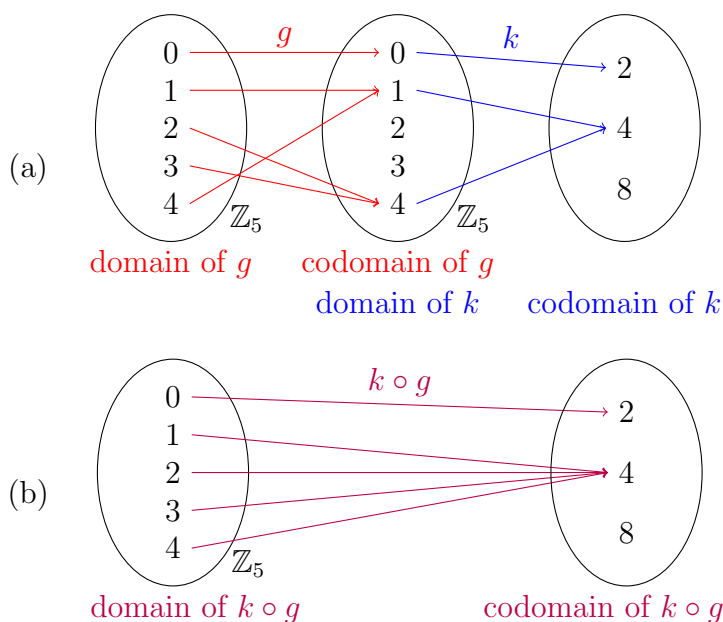
Solution.

- (i) The codomain of g is equal to the domain of h , so the composite $h \circ g$ exists. We specify $h \circ g : \{-1, 0, 1\} \rightarrow \{2, 3, 4\}$ by evaluating it at all elements of its domain. We have:

$$\begin{aligned} (h \circ g)(-1) &= h(g(-1)) = h(1) = 3 \\ (h \circ g)(0) &= h(g(0)) = h(0) = 2 \\ (h \circ g)(1) &= h(g(1)) = h(1) = 3 \end{aligned}$$

- (ii) The codomain of h is not equal to the domain of g , so $g \circ h$ does not exist.

Figure 7.4.1: (a) The functions $k : \mathbb{Z}_5 \rightarrow \{2, 4, 8\}$ from Figure 7.1.3 and $g : \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ from Figure 7.1.4 composed. (b) The composite $k \circ g : \mathbb{Z}_5 \rightarrow \{2, 4, 8\}$.



7.5 Identity Functions

Definition 7.5.1. For any set A , the function $\text{id}_A : A \rightarrow A$ given by $\text{id}_A(b) = b$ for all $b \in A$ is the *identity function* on A .

Example 7.5.2. The identity function on $\mathbb{Z}_3 = \{0, 1, 2\}$ is the function $\text{id}_{\mathbb{Z}_3} : \mathbb{Z}_3 \rightarrow \mathbb{Z}_3$ given by:

$$\text{id}_{\mathbb{Z}_3}(0) = 0 \quad \text{id}_{\mathbb{Z}_3}(1) = 1 \quad \text{id}_{\mathbb{Z}_3}(2) = 2$$

We examine the behavior of the identity function with respect to composition. Let A and B be sets, and let $f : A \rightarrow B$ be a function. Let $a \in A$. Then,

$$(f \circ \text{id}_A)(a) = f(\text{id}_A(a)) = f(a),$$

implying that $f \circ \text{id}_A = f$. Furthermore, $\text{id}_B(b) = b$ for all $b \in B$. So in particular,

$$(\text{id}_B \circ f)(a) = \text{id}_B(f(a)) = f(a),$$

implying that $\text{id}_B \circ f = f$. We have proven:

Theorem 7.5.3. Let A and B be sets, and let $f : A \rightarrow B$ be a function. Then $f \circ \text{id}_A = f$ and $\text{id}_B \circ f = f$.

Problem 7.5.4. Decide whether the function $f : \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ given by $f(x) = (x^5) \bmod 5$ is the identity function on $\mathbb{Z}_5$.

Solution. We evaluate the function f at all elements of its domain $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$. If $f(x) = x$ for all $x \in \mathbb{Z}_5$ then f is the identity function on $\mathbb{Z}_5$.

$$\begin{aligned} f(0) &= (0^5) \bmod 5 = 0 \bmod 5 = 0 \\ f(1) &= (1^5) \bmod 5 = 1 \bmod 5 = 1 \\ f(2) &= (2^5) \bmod 5 = 32 \bmod 5 = 2 \\ f(3) &= (3^5) \bmod 5 = 243 \bmod 3 = 3 \\ f(4) &= (4^5) \bmod 5 = 1024 \bmod 5 = 4 \end{aligned}$$

As $f(x) = x$ for all $x \in \mathbb{Z}_5$ it is the identity function on $\mathbb{Z}_5$.

Problem 7.5.5. Decide whether the function $f : \mathbb{Z}_4 \rightarrow \mathbb{Z}_4$ given by $f(x) = (x^2) \bmod 4$ is the identity function on $\mathbb{Z}_4$.

Solution. We evaluate the function f at all elements of its domain $\mathbb{Z}_4 = \{0, 1, 2, 3\}$. If $f(x) = x$ for all $x \in \mathbb{Z}_4$ then f is the identity function on $\mathbb{Z}_4$.

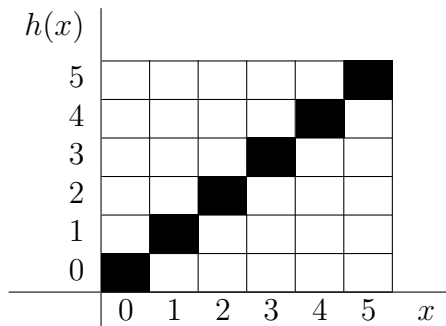
$$\begin{aligned} f(0) &= (0^2) \bmod 4 = 0 \bmod 4 = 0 \\ f(1) &= (1^2) \bmod 4 = 1 \bmod 4 = 1 \\ f(2) &= (2^2) \bmod 4 = 4 \bmod 4 = 0 \end{aligned}$$

We have found that $f(2) = 0 \neq 2$. So f is not the identity function on $\mathbb{Z}_4$.

Example 7.5.6. Recall that $\mathbb{Z}_6 = \{0, 1, 2, 3, 4, 5\}$. The function $h : \mathbb{Z}_6 \rightarrow \mathbb{Z}_6$ be given by $h(x) = x$ is the identity function on $\mathbb{Z}_6$. The graph of h is

$$\{(x, h(x)) \mid x \in \mathbb{Z}_6\} = \{(x, x) \mid x \in \mathbb{Z}_6\} = \{(0, 0), (1, 1), (2, 2), (3, 3), (4, 4), (5, 5)\}$$

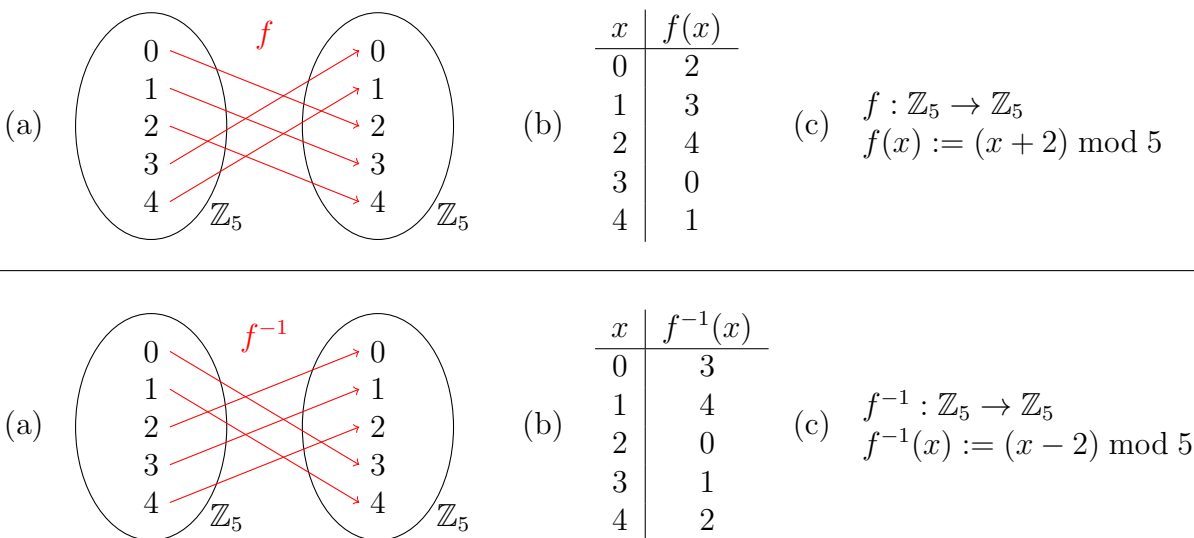
and its graphical representation where elements of the set above are represented by black pixels is:



7.6 Inverse Functions

Sometimes it is possible to “undo” the effect of a function. In these cases, we can define a function that reverses the effects of another function. A function that we can “undo” is called *invertible*.

Figure 7.6.1: We specify an invertible function $f : \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ and its inverse f^{-1} by a graph in (a), by a table in (b), and by an algebraic rule in (c).



Definition 7.6.1. Let A and B be non-empty sets. We say that a function $f : A \rightarrow B$ is *invertible* if for every $b \in B$ there is exactly one $a \in A$ such that $f(a) = b$. The *inverse* of an invertible function $f : A \rightarrow B$, denoted by f^{-1} , is the function $f^{-1} : B \rightarrow A$ that assigns to each element $b \in B$ the unique element $a \in A$ such that $f(a) = b$.

In other words, a function $f : A \rightarrow B$ is invertible if every $b \in B$ has exactly one preimage $a \in A$. So if $f(a) = b$, then $f^{-1}(b) = a$.

Example 7.6.2. We use the functions $\text{studentid} : N \rightarrow I$ and $\text{grade} : I \rightarrow G$ from Examples 7.1.2 and 7.1.3 given by the tables in Figures 7.1.1 and 7.1.2 respectively.

The function $\text{studentid} : N \rightarrow I$ where I is the set of student identification numbers and N is the set of student names is invertible as long as every student has a different name. The function $\text{studentid}^{-1} : I \rightarrow N$ is the function that tells us the student's name for a given identification number. With the table in Figure 7.1.1 we get

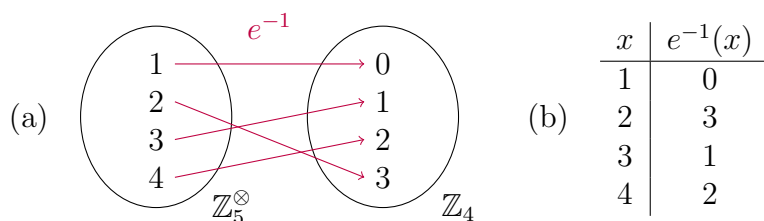
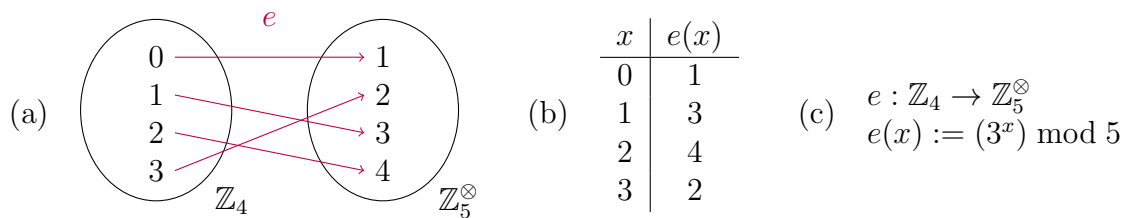
$$\text{studentid}(\text{Alice}) = 1001, \text{ so } \text{studentid}^{-1}(0001) = \text{Alice}.$$

Example 7.6.3. Recall the function $\text{grade} : I \rightarrow G$ where I is the set of identification numbers and G is the set of grades is not invertible since many students may earn the same grade in a class. Both the students with the identification number 1007 and 1008 earn an A in MAT 112. We have $\text{grade}(1007) = A$ and $\text{grade}(1008) = A$, and we would not be able to uniquely define $\text{grade}^{-1}(A)$.

In Figure 7.6.1 we give an example of an invertible function from $\mathbb{Z}_5$ to $\mathbb{Z}_5$ and its inverse. The function e in Figure 7.6.2 illustrates that for an invertible function the domain and codomain do not have to be the same.

Example 7.6.4. Consider the function $b : \mathbb{Z}_4 \rightarrow \mathbb{Z}_2 \times \mathbb{Z}_2$ given by $b(0) := (0, 0)$, $b(1) := (0, 1)$, $b(2) := (1, 0)$, and $b(3) := (1, 1)$. Since every element in $\mathbb{Z}_2 \times \mathbb{Z}_2$ has exactly one preimage

Figure 7.6.2: We specify an invertible function $e : \mathbb{Z}_4 \rightarrow \mathbb{Z}_5^\otimes$ and its inverse e^{-1} by a graph in (a) and by a table in (b). Additionally, the algebraic rule that defines e is given in (c).



under b , the function b is invertible. The inverse $b^{-1} : \mathbb{Z}_2 \times \mathbb{Z}_2 \rightarrow \mathbb{Z}_4$ of the function b is given by $b^{-1}((0, 0)) := 0$, $b^{-1}((0, 1)) := 1$, $b^{-1}((1, 0)) := 2$, and $b^{-1}((1, 1)) := 3$.

Suppose that $f : A \rightarrow B$ is invertible, and let $f^{-1} : B \rightarrow A$ be its inverse. If $f(a) = b$, then $f^{-1}(b) = a$. Thus for the composition functions $f^{-1} \circ f$ and $f \circ f^{-1}$, we have

$$(f^{-1} \circ f)(a) = f^{-1}(f(a)) = f^{-1}(b) = a.$$

Hence $f^{-1} \circ f = \text{id}_A$ where id_A denotes the identity function on A . Similarly

$$(f \circ f^{-1})(b) = f(f^{-1}(b)) = f(a) = b$$

and so $f \circ f^{-1} = \text{id}_B$.

For $f^{-1} : B \rightarrow A$ the inverse is the function $(f^{-1})^{-1} : A \rightarrow B$ that assigns to $a \in A$ the element $b \in B$ such that $f^{-1}(b) = a$. This element b is equal to $f(a)$. Thus $(f^{-1})^{-1}(a) = f(a)$. Therefore $(f^{-1})^{-1} = f$. So f is the inverse of f^{-1} .

Theorem 7.6.5. *If $f : A \rightarrow B$ is an invertible function and $f^{-1} : B \rightarrow A$ is its inverse, then*

- (i) f is the inverse of f^{-1} ,
- (ii) $f \circ f^{-1} = \text{id}_B$, and
- (iii) $f^{-1} \circ f = \text{id}_A$.

Assume that $f : A \rightarrow B$ is a function and there is a function $g : B \rightarrow A$ such that $g \circ f = \text{id}_A$. Let $a \in A$ and $b = f(a)$. Now $g(b) = g(f(a)) = (g \circ f)(a) = \text{id}_A(a) = a$. This assignment $g(b) = a$ is unique, as g is a function. So g is the inverse of f . We have proven:

Theorem 7.6.6. Let $f : A \rightarrow B$ be a function. If there is a function $g : B \rightarrow A$ such that $g \circ f = \text{id}_A$ then g is the inverse of f .

Problem 7.6.7. Let g and h as in Problem 7.4.5:

$$g : \{-1, 0, 1\} \rightarrow \{0, 1, 2\} \text{ be given by } g(x) := x^2, \text{ and}$$

$$h : \{0, 1, 2\} \rightarrow \{2, 3, 4\} \text{ be given by } h(y) := y + 2.$$

- (i) Is g invertible? If yes, specify g^{-1} .
- (ii) Is h invertible? If yes, specify h^{-1} .

Solution.

- (i) The function g is not invertible since $2 \in \{0, 1, 2\}$ has no preimage in $\{-1, 0, 1\}$. (Note that g fails to be invertible for multiple reasons, since $g(-1) = 1$ and $g(1) = 1$.)
- (ii) The function h is invertible since each element in the domain is assigned to a distinct element in the codomain. We have $h^{-1} : \{2, 3, 4\} \rightarrow \{0, 1, 2\}$ defined by $h^{-1}(z) = z - 2$. In particular, $h^{-1}(2) = 0$, $h^{-1}(3) = 1$, and $h^{-1}(4) = 2$.

Example 7.6.8. Let $f : \mathbb{Z}_7 \rightarrow \mathbb{Z}_7$ be given by $f(x) = 3(x + 1) \bmod 7$. We have

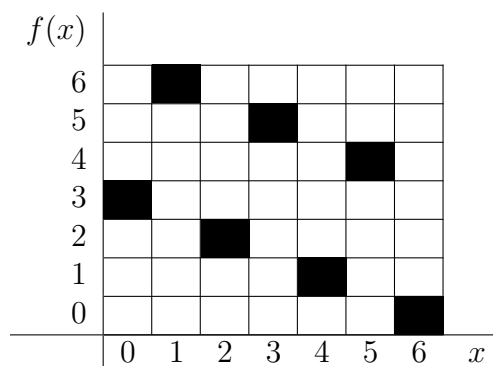
$$\begin{aligned} f(0) &= 3(0 + 1) \bmod 7 = 3 \bmod 7 = 3 \\ f(1) &= 3(1 + 1) \bmod 7 = 6 \bmod 7 = 6 \\ f(2) &= 3(2 + 1) \bmod 7 = 9 \bmod 7 = 2 \\ f(3) &= 3(3 + 1) \bmod 7 = 12 \bmod 7 = 5 \\ f(4) &= 3(4 + 1) \bmod 7 = 15 \bmod 7 = 1 \\ f(5) &= 3(5 + 1) \bmod 7 = 18 \bmod 7 = 4 \\ f(6) &= 3(6 + 1) \bmod 7 = 21 \bmod 7 = 0 \end{aligned}$$

Thus $f(\mathbb{Z}_7) = \mathbb{Z}_7$ and no element in the codomain has the same preimage. Hence the function f is invertible.

The graph of f is

$$\{(x, f(x)) \mid x \in \mathbb{Z}_7\} = \{(0, 3), (1, 6), (2, 2), (3, 5), (4, 1), (5, 4), (6, 0)\} \quad (7.1)$$

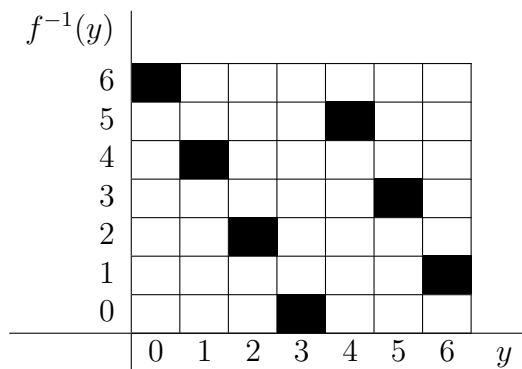
and its graphical representation where the elements of the set are represented by black pixels is:



We obtain the graph of the inverse f^{-1} by swapping the order of the numbers in the ordered pairs in the graph of f . Thus the graph of f^{-1} is

$$\{(3, 0), (6, 1), (2, 2), (5, 3), (1, 4), (4, 5), (0, 6)\}$$

and its graphical representation is:



Chapter 8

Codes

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Convert text into a sequence of numbers.
- (2) Apply substitution ciphers to encrypt and decrypt texts.
- (3) Apply frequency analysis to break Caesar ciphers.

In this section we apply functions in the encoding and encryption of texts. We first introduce a function that we use to represent characters by numbers in Section 8.1. This is followed by a short introduction to symmetric key cryptography in Section 8.2. and the description of simple symmetric key cryptosystems, namely Caesar ciphers in Section 8.3 and other substitution ciphers in Section 8.4. We conclude this section with an attack on substitution ciphers called frequency analysis in Section 8.5.

8.1 Character Encoding

A *code* is a system of rules to convert information from one form to another. When we convert given information into another representation, we are *encoding*. When we convert back to the original representation, we are *decoding*. We represent the rules for encoding and decoding by functions. To be able to recover the original information through decoding, the encoding function must be invertible.

In this section we convert text into a sequence of numbers. Commonly used character encodings are *ASCII* (*American Standard Code for Information Interchange*) and *Unicode*. ASCII uses 128 printable and control characters and was standardized in 1963 by ASA (American Standards Association). Unicode can handle the characters in most of the world's writing systems.

We use a simpler code that only encodes the characters in the set

$$\mathbb{A} = \{-, a, b, c, \dots, z\}$$

into a sequence of numbers in $\mathbb{Z}_{27} = \{0, 1, 2, \dots, 26\}$. We use the encoding function C and its inverse the decoding function C^{-1} given in Figure 8.1.1. Note that when we write texts with the characters in $\mathbb{A}$ we write - instead of the character space.

Problem 8.1.1. Encode the word `cookies` with the encoding function C .

Solution. We have $C(c) = 3$, $C(o) = 15$, $C(k) = 11$, $C(i) = 9$, $C(e) = 5$, and $C(s) = 19$. Thus `cookies` is encoded as the numbers

$$3, 15, 15, 11, 9, 5, 19$$

Figure 8.1.1: Tables that specify the encoding function $C : \mathbb{A} \rightarrow \mathbb{Z}_{27}$ and its inverse the decoding function $C^{-1} : \mathbb{Z}_{27} \rightarrow \mathbb{A}$

x	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
$C(x)$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

y	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
$C^{-1}(y)$	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z

Problem 8.1.2. Decode 20, 15, 15, 0, 5, 1, 19, 25 with the decoding function C^{-1} .

Solution. We have $C^{-1}(20) = \mathfrak{t}$, $C^{-1}(15) = \mathfrak{o}$, $C^{-1}(0) = -$, $C^{-1}(5) = \mathfrak{e}$, $C^{-1}(1) = \mathfrak{a}$, $C^{-1}(19) = \mathfrak{s}$, and $C^{-1}(25) = \mathfrak{y}$. Thus we obtain the words too-easy.

Problem 8.1.3. Encode the text

and-therefore-never-send-to-know-for-whom-the-bell-tolls-
it-tolls-for-thee ¹

with the function C from Figure 8.1.1.

Solution. We obtain the sequence of elements of $\mathbb{Z}_{27}$:

1, 14, 4, 0, 20, 8, 5, 18, 5, 6, 15, 18, 5, 0, 14, 5, 22, 5, 18, 0, 19, 5, 14, 4, 0, 20,
15, 0, 11, 14, 15, 23, 0, 6, 15, 18, 0, 23, 8, 15, 13, 0, 20, 8, 5, 0, 2, 5, 12, 12, 0,
20, 15, 12, 12, 19, 0, 9, 20, 0, 20, 15, 12, 12, 19, 0, 6, 15, 18, 0, 20, 8, 5, 5

Problem 8.1.4. Decode the sequence of elements of $\mathbb{Z}_{27}$

12, 5, 20, 0, 13, 5, 0, 14, 15, 20, 0, 20, 15, 0, 20, 8, 5, 0, 13, 1, 18, 18, 9, 1, 7,
5, 0, 15, 6, 0, 20, 18, 21, 5, 0, 13, 9, 14, 4, 19, 0, 1, 4, 13, 9, 20, 0, 9, 13, 16,
5, 4, 9, 13, 5, 14, 20, 19

with the function C^{-1} from Figure 8.1.1.

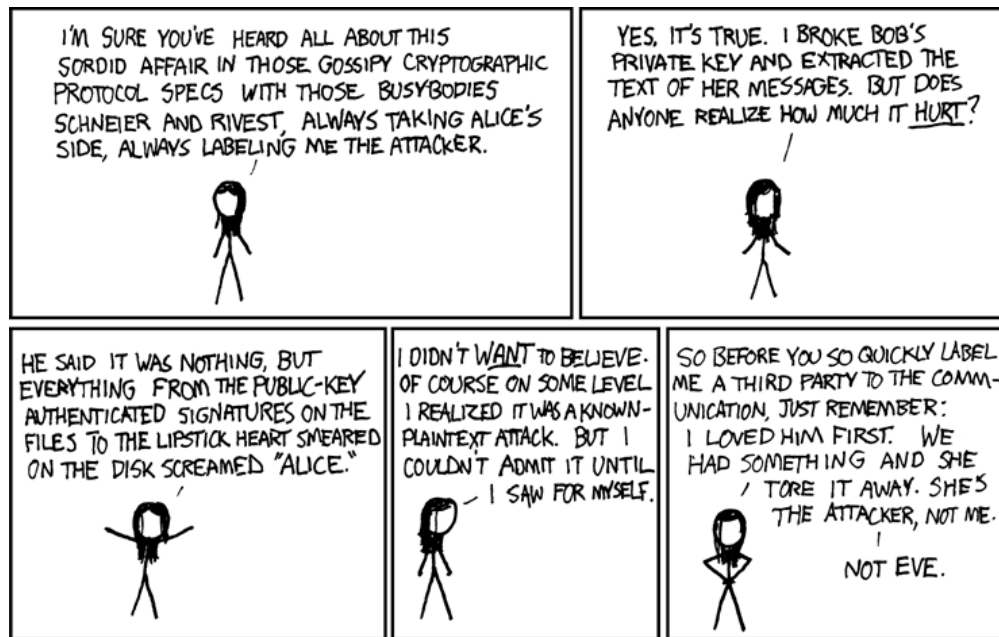
Solution. We obtain the text:

let-me-not-to-the-marriage-of-true-minds-admit-impediments ²

¹from the poem *Devotions upon Emergent Occasions* by John Donne, 1624

²from Shakespeare's Sonnet 116, 1609

Figure 8.1.2: *Alice and Bob* by R. Munroe (<https://xkcd.com/177>).



Yet one more reason I'm barred from speaking at crypto conferences.

8.2 Symmetric Key Cryptography

Symmetric key cryptography is another application of functions. An *encryption* function turns readable *plain text* into unreadable *cipher text*, and the corresponding *decryption* function turns the cipher text back into the original plain text.

Descriptions of cryptographic protocols are commonly phrased as interactions between *Alice*, *Bob*, and *Eve*. Alice sends a message to Bob, and the eavesdropper Eve listens in on their conversation and tries to break their encryption (Figure 8.2.1). In a symmetric key encryption scheme, Alice and Bob first have to agree on a common shared key. Alice uses the key to encrypt a message and sends the encrypted message to Bob. Then, Bob uses the key to decrypt the encrypted message that was sent by Alice in order to obtain the message in its original form (Figure 8.2.2).

8.3 Caesar Cipher

One of the earliest known approaches to symmetric key cryptography was applied by *Julius Caesar* (100 BC to 44 BC) and is now called the *Caesar cipher*. Caesar cyclically shifted the alphabet by n letters, where n is a natural number. Caesar did not encrypt the character space and most other authors also follow that convention.

Instead of just shifting the letters of the alphabet, we will shift our set of characters that contains the 26 letters as well as the space character $-$.

Figure 8.2.1: Alice sends a message to Bob, and Eve eavesdrops on their conversation.

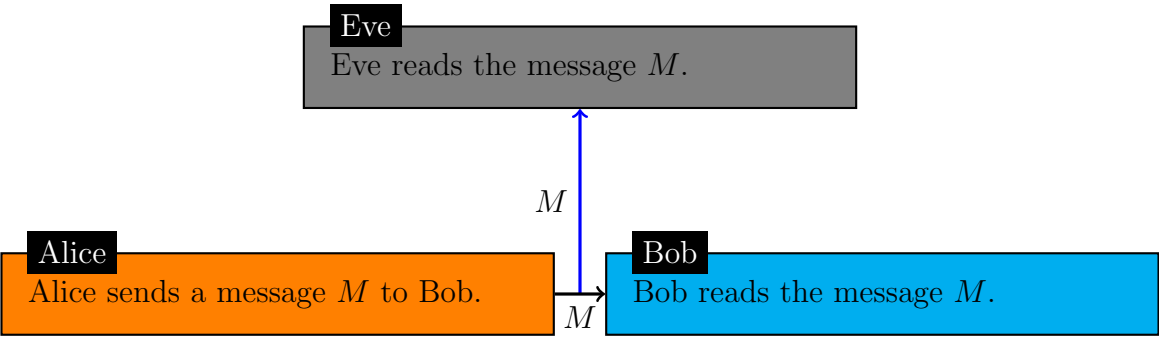
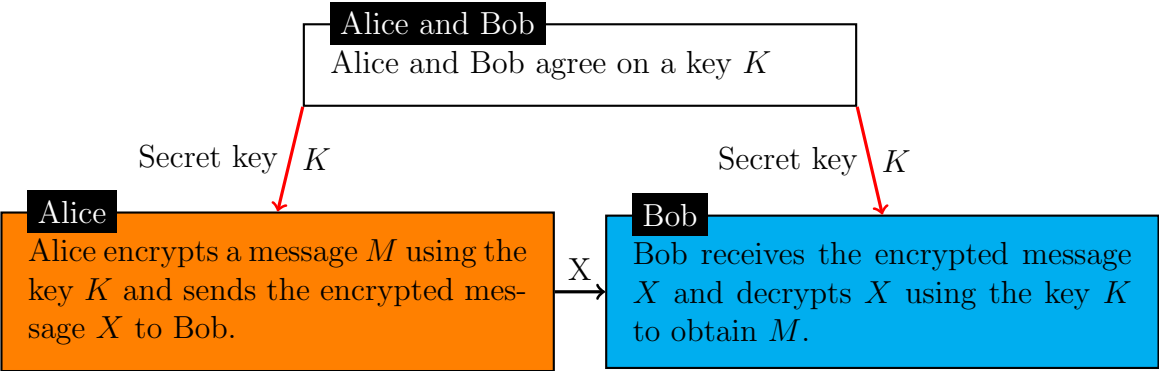


Figure 8.2.2: In a symmetric key encryption scheme, Alice and Bob share a common secret, namely the key K .



There are several ways of representing and evaluating the decryption and encryption functions of a Caesar cipher. In the following example we give the functions by table and a *decoder disc*. The decoder disc illustrates how the characters wrap around because of the cyclic shift, Figure 8.3.1.

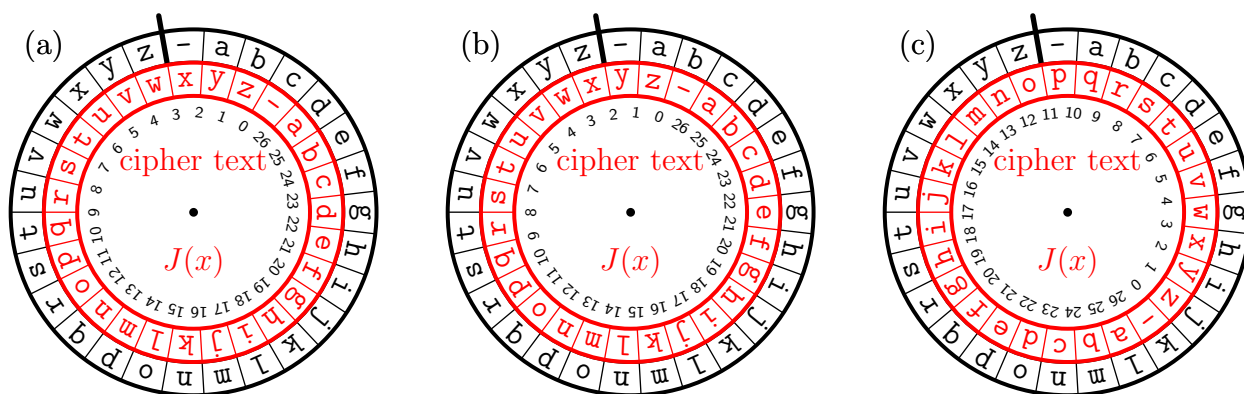
Example 8.3.1. The encryption function $J : \mathbb{A} \rightarrow \mathbb{A}$ for the Caesar cipher with $n = 3$ (Caesar’s original choice) is given in the following table:

x	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
$J(x)$	x	y	z	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w

The corresponding decryption function $J^{-1} : \mathbb{A} \rightarrow \mathbb{A}$ is given in the following table:

y	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
$J^{-1}(y)$	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	-	a	b

Figure 8.3.1: Decoder discs for Caesar ciphers (a) shifting by 3 characters, (b) shifting by 2 character, and (b) shifting by 11 characters. In the outer (black) ring are the characters in plain text and in the inner (red) ring the characters in cipher text. The number on the inner disc is aligned with the line between z and - is 3, 2, and 11 respectively.



Notice that the encryption function J the alphabet *backwards* by $n = 3$ character, and the decryption function J^{-1} shifted the alphabet *forwards* by $n = 3$ places.

The functions J and J^{-1} are also represented by the decoder disc in Figure 8.3.1 (a). To evaluate the encryption function J we read from the outer ring to the inner ring. To evaluate the decryption function we read from the inner ring to the outer ring.

Problem 8.3.2. *Encrypt gaius-julius using the Caesar cipher shifting by 3 characters.*

Solution. We apply the function J from Example 8.3.1 which is also given by the decoder disc in Figure 8.3.1 (a). Avoiding duplication we get:

$$J(g) = d, J(a) = y, J(i) = f, J(u) = r, J(s) = p, J(-) = x, J(j) = g, J(l) = i$$

Thus *gaius-julius* is encrypted as *dyfrpxgrifrp*.

If one does not have table for the encryption at hand one counts (in this example 3) characters backwards.

Decrypting text that was encrypted with a Caesar cipher is easier than encryption, since when decrypting we count forward in the alphabet and most of us are better at going forward in the alphabet than backwards.

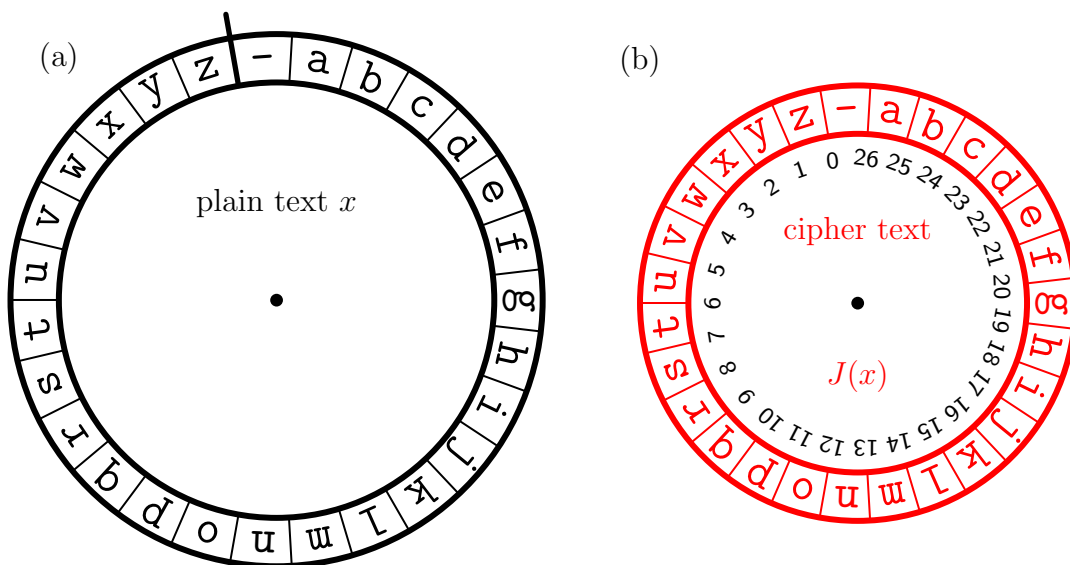
Problem 8.3.3. *Decrypt the cipher text zlbywmsy-psrsq that was encrypted with the Caesar cipher that shifts by 2 characters.*

Solution. To decrypt we count forward 2 characters, when we get to *z* we wrap around and continue with *-* which is followed by *a*. Thus for the decoding function J^{-1} we get

$$\begin{aligned} J^{-1}(z) &= a, J^{-1}(l) = n, J^{-1}(b) = d, J^{-1}(y) = -, J^{-1}(w) = y, J^{-1}(m) = o \\ J^{-1}(s) &= u, J^{-1}(-) = b, J^{-1}(p) = r, J^{-1}(r) = t, J^{-1}(q) = s \end{aligned}$$

So the decrypted text is *and-you-brutus*. Alternatively we can also use the decoder disc in Figure 8.3.1 (b).

Figure 8.3.2: Build your own decoder disc. Cut out the plain text disc (a) and the cipher text disc (b). Punch a hole into the center of each disc. Put a split pin through the center of both discs. To obtain the decoder disc for the Caesar cipher that shifts by a natural number n align the number n on the inner disc with the line between z and -.



In general, we may describe a Caesar cipher with an arbitrary $n \in \mathbb{Z}_{27}$ as a key using the following functions:

$C : \mathbb{A} \rightarrow \mathbb{Z}_{27}$	as defined in Figure 8.1.1
$E : \mathbb{Z}_{27} \rightarrow \mathbb{Z}_{27}$	given by $E(x) = (x - n) \bmod 27$
$E^{-1} : \mathbb{Z}_{27} \rightarrow \mathbb{Z}_{27}$	given by $E^{-1}(x) = (x + n) \bmod 27$
$C^{-1} : \mathbb{Z}_{27} \rightarrow \mathbb{A}$	as defined in Figure 8.1.1

The encryption function for a Caesar cipher is $J = C^{-1} \circ E \circ C$. It first encodes each character into an element of $\mathbb{Z}_{27}$ using the function C . Then, the function E does the shift for the encryption by subtracting n from each number and determining the result modulo n , where n is the key for the particular Caesar cipher. Finally, it converts each new number into a new character using the function C^{-1} .

The decryption function for a Caesar cipher is $J^{-1} = C^{-1} \circ E^{-1} \circ C$. It first encodes each character into an element of $\mathbb{Z}_{27}$ using the function C . Then, the function E^{-1} does the shift for the decryption by adding n from each number and determining the result modulo n , where n is the key for the particular Caesar cipher. Finally, it converts each new number into a new character using the function C^{-1} .

Example 8.3.4. Alice and Bob agree to encrypt their communication with the Caesar cipher using the key $n = 11$. Alice sends Bob the encrypted message:

ndjppgupaupqkycwpixupqbugysqcphusidg

To decrypt the message, Bob first encodes the cipher text with the encoding function C from Figure 8.1.1 and gets:

14, 4, 10, 16, 17, 7, 21, 16, 1, 21, 17, 11, 25, 3, 23, 16, 9, 24, 21, 16, 17, 2, 21, 7,
25, 19, 17, 3, 16, 8, 21, 19, 9, 4, 7

Next, he applies the function E^{-1} with $n = 11$ to each of the numbers by adding 11 and determining the result modulo 27 to obtain:

25, 15, 21, 0, 1, 18, 5, 0, 12, 5, 1, 22, 9, 14, 7, 0, 20, 8, 5, 0, 1, 13, 5, 18, 9, 3, 1,
14, 0, 19, 5, 3, 20, 15, 18

Finally, he decodes this with the decoding function C^{-1} from Figure 8.1.1 and gets the decrypted plain text message:

you-are-leaving-the-american-sector ³

Instead of formally applying the functions C , E^{-1} , and C^{-1} , Bob could have also created a table as in Example 8.3.1 or counted 11 letters forward (wrapping around to - after z) from the letters in the cipher text or used the decoder disc in Figure 8.3.1 (c).

8.4 Other Substitution Ciphers

The Caesar cipher is an example of a substitution cipher, where one character is replaced by another. Other substitution ciphers use more complicated rules or tables for the encoding of characters. We give an example for another substitution cipher given by an algebraic rule.

Example 8.4.1. Alice and Bob want to use the function

$$E : \mathbb{Z}_{27} \rightarrow \mathbb{Z}_{27}, E(c) = (7 \cdot c) \bmod 27$$

for the encryption. However, they need to also determine if E has an inverse function so that the encrypted messages can be decrypted. They notice that

$$(7 \cdot 4) \bmod 27 = 28 \bmod 27 = 1,$$

which leads them to hypothesize that the function

$$D : \mathbb{Z}_{27} \rightarrow \mathbb{Z}_{27}, D(b) = (4 \cdot b) \bmod 27$$

³from the signs along the border (that is, along the wall) between the American sector and the Soviet sector of Berlin, before the fall of the Berlin wall in 1989.

Figure 8.4.1: *Protocol* by R. Munroe (<https://xkcd.com/1323>).



Changing the names would be easier, but if you're not comfortable lying, try only making friends with people named Alice, Bob, Carol, etc

should be the inverse of the function E . To verify their hypothesis, they compute:

$$\begin{aligned} D(E(c)) &= D((7 \cdot c) \bmod 27) = (4 \cdot (7 \cdot c) \bmod 27) \bmod 27 = (4 \cdot 7 \cdot c) \bmod 27 \\ &= (28 \cdot c) \bmod 27 = ((28 \bmod 27) \cdot c) \bmod 27 = (1 \cdot c) \bmod 27 = c \bmod 27 = c \end{aligned}$$

They conclude that D is the inverse of E , So E is a useful encryption function and the corresponding decryption function is D . Alice decides to encrypt the following message and send it to Bob:

here-i-am-brain-the-size-of-a-planet⁴

She begins by encoding the message using the function C from Figure 8.1.1:

8, 5, 18, 5, 0, 9, 0, 1, 13, 0, 2, 18, 1, 9, 14, 0, 20, 8, 5, 0, 19, 9, 26, 5, 0, 15, 6, 0,
1, 0, 16, 12, 1, 14, 5, 20

Then she encrypts this sequence of numbers with the function E :

2, 8, 18, 8, 0, 9, 0, 7, 10, 0, 14, 18, 7, 9, 17, 0, 5, 2, 8, 0, 25, 9, 20, 8, 0, 24, 15, 0,
7, 0, 4, 3, 7, 17, 8, 5

For transmission, she applies the function C^{-1} from Figure 8.1.1 to obtain the cipher text:

bhrh-i-gj-nrgiq-ebh-yith-xo-g-dcgqhe

⁴from the *Hitchhikers Guide to the Galaxy* by Douglas Adams, 1978 (radio play), 1979 (novel)

Finally, Alice sends this encrypted message to Bob. After receiving the message, Bob needs to decrypt the message. So, he begins by applying the function C to change the cipher text to numbers:

2, 8, 18, 8, 0, 9, 0, 7, 10, 0, 14, 18, 7, 9, 17, 0, 5, 2, 8, 0, 25, 9, 20, 8, 0, 24, 15, 0,
7, 0, 4, 3, 7, 17, 8, 5

Then he decrypts this sequence of numbers with the function $D = E^{-1}$:

8, 5, 18, 5, 0, 9, 0, 1, 13, 0, 2, 18, 1, 9, 14, 0, 20, 8, 5, 0, 19, 9, 26, 5, 0, 15, 6, 0,
1, 0, 16, 12, 1, 14, 5, 20

Finally, he applies C^{-1} to change the numbers back to plain text:

`here-i-am-brain-the-size-of-a-planet`

Problem 8.4.2. *Encrypt the word **ball** with the encryption function $E : \mathbb{Z}_{27} \rightarrow \mathbb{Z}_{27}$ given by $E(x) = ((4 \cdot x) + 5) \bmod 27$. Give the cipher text as characters.*

Solution. With the encoding function C we get:

$$C(\mathbf{b}) = 2, C(\mathbf{a}) = 1, C(1) = 12$$

Applying the encryption function E we obtain:

$$E(2) = ((4 \cdot 2) + 5) \bmod 27 = 13$$

$$E(1) = ((4 \cdot 1) + 5) \bmod 27 = 9$$

$$E(12) = ((4 \cdot 12) + 5) \bmod 27 = 26$$

Now we convert these values back to characters with the decoding function C^{-1} :

$$C^{-1}(13) = \mathbf{m}, C^{-1}(9) = \mathbf{i}, C^{-1}(26) = \mathbf{z}$$

So the encrypted word is **mizz**.

Problem 8.4.3. *Decrypt the cipher text **bilt** with the decryption function $D : \mathbb{Z}_{27} \rightarrow \mathbb{Z}_{27}$ given by $D(x) = (7 \cdot x) \bmod 27$. Give the cipher text as characters.*

Solution. With the encoding function C we get:

$$C(\mathbf{b}) = 2, C(\mathbf{i}) = 9, C(1) = 12, C(\mathbf{t}) = 20$$

Applying the decryption function D we obtain:

$$D(2) = (7 \cdot 2) \bmod 27 = 14 \quad D(9) = (7 \cdot 9) \bmod 27 = 9$$

$$D(12) = (7 \cdot 12) \bmod 27 = 3 \quad D(20) = (7 \cdot 20) \bmod 27 = 5$$

Now we convert these values back to characters with the decoding function C^{-1} :

$$C^{-1}(14) = \mathbf{n}, C^{-1}(9) = \mathbf{i}, C^{-1}(3) = \mathbf{c}, C^{-1}(5) = \mathbf{e}$$

So the decrypted word is **nice**.

8.5 Frequency Analysis

Suppose that the eavesdropper Eve intercepts the cipher text from Alice to Bob. In order to decrypt the message, Eve would need to know the decryption function for the substitution cipher. A simple exhaustive attack on a Caesar cipher would be for Eve to try out all 27 possible decryption functions (the 27 possible shifts) until she obtains a readable message.

We describe another method, called *frequency analysis*, that enables Eve to decrypt messages encrypted with a substitution cipher. This attack is based on the observation that in an English text, not all letters occur with the same frequency. See Figure 8.5.1 for the frequency of letters and space in two classic novels. In a substitution cipher, if a letter or space is replaced by another symbol, the replacement symbol occurs with the same frequency as the original letter or space it is replacing. Counting the frequency of the symbols in the cipher text and comparing it with the expected frequency of the letters and space (as communicated in tables such as the one in Figure 8.5.1) gives an indication of which letter or space could have been replaced by which symbol.

For Caesar ciphers, this attack is particularly easy. We only have to find the plain text letter that corresponds to one cipher text letter. Doing so will yield the key n that provides the decryption function. The following problem demonstrates Eve's approach to deciphering the intercepted message that was encrypted using a Caesar cipher.

Problem 8.5.1. *The following encrypted message from Alice to Bob is intercepted by Eve.*

tddsztmdsakswanawxwsaflgsl-jxxshthlksgfxsgyso-av-sl-xsuxdztxsaf-
tualsl-xstimaltfastfgl-xjsl-gkxso-gsafsl-xajsgofsdftfzmtzxstjxsvt
ddxwsvxdlksafsgmjksztmdksl-xsl-ajwstddsl-xkxswayyxjsyjgesxtv-sgl
-xjsafsdftfzmtzxsvmklgekstfwsdtok

Eve knows that Alice and Bob are using a Caesar cipher. Decipher the message for Eve.

Solution. Counting the number of occurrences of the characters in the cipher text we get:

-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
15	15	0	0	13	2	12	11	1	1	10	10	16	7	1	4	0	0	0	40	19	2	5	7	22	4	7

According to the data in Figure 11.4.1 the character space which we represent by - is the most common character in English language texts. Since s is the character in the cipher text with the highest number of occurrences, Eve tries decrypting the cipher text with the Caesar cipher that replaces - with s.

x	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
$J(x)$	s	t	u	v	w	x	y	z	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r

When decrypting Eve reads the table from bottom to top. So Eve decrypts t to a, d to l, s to -, and so on. Eve obtains the following message.

all-gaul-is-divided-into-three-parts-one-of-which-the-belgae-inhabit-
the-aquitani-another-those-who-in-their-own-language-are-called-celts-
in-ours-gauls-the-third-all-these-differ-from-each-other-in-language-
customs-and-laws ⁵

Frequency analysis can also be effective for much shorter texts.

Problem 8.5.2. Decrypt the cipher text `qexldmwdedpsxdsjdjyr` that was encrypted using a Caesar cipher. By how many characters does the Caesar cipher shift?

Solution. We count the number of occurrences of each character in the cipher text. We get:

character	q	e	x	l	d	m	w	p	s	j	y	r
count	1	2	2	1	5	1	1	1	2	2	1	1

The character with the highest count in the cipher text is d. So we try decrypting with the Caesar cipher that encrypts - as d.

x	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
$J(x)$	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	-	a	b	c

Reading the table from bottom to top, that is, cipher text to plain text, we get:

$q \mapsto m, e \mapsto a, x \mapsto t, l \mapsto h, d \mapsto -, m \mapsto i, w \mapsto s, p \mapsto l, s \mapsto o, j \mapsto f, y \mapsto u, r \mapsto n,$

So the plain text was `math-is-a-lot-of-fun`. From the table for the encryption function J we get that the Caesar cipher shifted by 23 characters.

Frequency analysis can also be used to decrypt text that was encrypted with other substitution ciphers. In general this requires a more careful analysis of the number of occurrences of each character in the cipher text.

The symmetric key ciphers used today are block ciphers, that is, a larger block of characters is encrypted at a time. *AES* (*Advanced Encryption Standard*) is one such a cipher that is widely used.

⁵from the English translation of Julius Caesar's *De Bello Gallico* (The Gallic Wars) by W. A. McDevitte and W. S. Bohn.

Figure 8.5.1: Frequency of letters and space in *Alice in Wonderland* by Lewis Carroll (1865) and in *The Time Machine* by H. G. Wells (1898)

character	Alice in Wonderland		The Time Machine	
	count	frequency	count	frequency
- (space)	27305	20.22%	32679	18.85%
a	8791	6.51%	11704	6.75%
b	1475	1.09%	1897	1.09%
c	2399	1.78%	3424	1.98%
d	4931	3.65%	6337	3.66%
e	13576	10.05%	17838	10.29%
f	2001	1.48%	3354	1.94%
g	2531	1.87%	3075	1.77%
h	7375	5.46%	8257	4.76%
i	7515	5.57%	10138	5.85%
j	146	0.11%	97	0.06%
k	1158	0.86%	1087	0.63%
l	4716	3.49%	6146	3.55%
m	2107	1.56%	4043	2.33%
n	7016	5.20%	9917	5.72%
o	8145	6.03%	9758	5.63%
p	1524	1.13%	2427	1.40%
q	209	0.15%	95	0.05%
r	5438	4.03%	7674	4.43%
s	6501	4.81%	8486	4.90%
t	10689	7.92%	13515	7.80%
u	3468	2.57%	3805	2.20%
v	846	0.63%	1295	0.75%
w	2676	1.98%	3225	1.86%
x	148	0.11%	236	0.14%
y	2262	1.68%	2679	1.55%
z	78	0.06%	144	0.08%
sum	135026		173332	

Part III

Numbers and Counting

In this third part of the course we bring together some topics from the first two chapters. In Chapter 9 we use functions to define the cardinality of sets, talk about the cardinality of infinite sets. Chapter 10 is about special integers namely prime numbers, We introduce prime numbers using divisibility and talk about factorization. We show that there are infinitely many prime numbers and present the Twin Prime Conjecture. The Twin Prime Conjecture is a mathematical statement that is believed to be true, but has not been proven yet. In Chapter 11 we discuss different representations of integers. In Chapter 12 we apply the representation of integers in other bases in the encoding of colors, images, and text.

Chapter 9

Cardinality

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Compute the cardinality of a finite set.
- (2) Recognize when two sets have the same cardinality.
- (3) Recognize whether a set is infinite.
- (4) Compute the cardinality of Cartesian products.
- (5) Compute the number of subsets of a set.

In this section we define precisely the size (or cardinality) of a set. We will use functions to help determine cardinality, which becomes important when we deal with infinite sets. We also find formulas for the cardinality of Cartesian products and the number of subsets of a set.

9.1 Definition of Cardinality

We introduce the terminology for speaking about the number of elements in a set, called the cardinality of the set. Intuitively we can say what the cardinality of a set is.

In the example below we illustrate the properties that we would want a precise definition of cardinality to have.

Example 9.1.1. We discuss possible cardinalities of some sets.

- (i) The empty set $\{\}$ contains no elements, so its cardinality should be 0.
- (ii) The set $\{\mathbf{m}\}$ contains one element, namely the character $\mathbf{m}$, so its cardinality should be 1.
- (iii) The set $\{\mathbf{m}, \mathbf{a}\}$ contains two (distinct) elements, namely the characters $\mathbf{m}$ and $\mathbf{a}$, so its cardinality should be 2.

We start with a definition of the cardinality of the empty set.

Definition 9.1.2. The cardinality of the empty set $\{\}$ is 0.

We write $\#\{\} = 0$ which is read as “the cardinality of the empty set is zero” or “the number of elements in the empty set is zero.”

We have the idea that cardinality should be the number of elements in a set. This works for sets with finitely many elements, but fails for sets with infinitely many elements. We approach cardinality in a way that works for all sets. First we define when we consider two

sets to have the same cardinality. Certainly two sets A and B have the same number of elements if we can pair each element in A with an element in B such that each element of A is in exactly one pair and each element of B is in exactly one pair. These pairs define an invertible function from A to B . This observation yields our definition of equality of cardinality.

Definition 9.1.3. Let A and B be non-empty sets. The sets A and B have the same *cardinality* means that there is an invertible function $f : A \rightarrow B$.

This definition does not specify what we mean by the cardinality of a set and does not talk about the number of elements in a set. This will come in handy, when we consider the cardinality of infinite sets in the next section. If the set B can be chosen as one of the sets $\mathbb{Z}_n$, we use this to define the cardinality of the set A .

Definition 9.1.4. Let A be a set. If there is $n \in \mathbb{N}$ such that A and $\mathbb{Z}_n$ have the same cardinality, we say that the *cardinality* of A is n and write $\#A = n$.

We read $\#A = n$ as “the cardinality of A is n ” or “the number of elements of A is n .”

Example 9.1.5. Let $A = \{\mathfrak{m}, \mathfrak{a}, \mathfrak{t}, \mathfrak{h}\}$ and let $f : A \rightarrow \mathbb{Z}_4$ given by $f(\mathfrak{m}) := 0$, $f(\mathfrak{a}) := 1$, $f(\mathfrak{t}) := 2$, $f(\mathfrak{h}) := 3$. As the function f is invertible, the cardinality of A is 4.

Example 9.1.6. Consider the function $C : \mathbb{A} \rightarrow \mathbb{Z}_{27}$ given by

x	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
$C(x)$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

Clearly the function C is invertible, thus $\mathbb{A}$ has 27 elements.

We revisit the examples from the beginning of this section.

Example 9.1.7. We give the cardinalities of some sets. In all but the first example we list an invertible function whose existence gives the cardinality of the set. This function in most cases is not unique, we only need to know that such a function exists.

- (i) $\#\{\} = 0$ by Definition 9.1.2.
- (ii) $\#\{\mathfrak{m}\} = 1$, because the function $f : \{\mathfrak{m}\} \rightarrow \mathbb{Z}_1$ given by $f(\mathfrak{m}) = 0$ is invertible (recall that $\mathbb{Z}_1 = \{0\}$).
- (iii) $\#\{\mathfrak{m}, \mathfrak{a}\} = 2$, because the function $f : \{\mathfrak{m}, \mathfrak{a}\} \rightarrow \mathbb{Z}_2$ given by $f(\mathfrak{m}) = 0$ and $f(\mathfrak{a}) = 1$ is invertible.

In practice, for many sets, we do not need to find such an invertible function f to determine the cardinality of the set.

Example 9.1.8. We give the cardinality of some other sets.

- (i) $\#\{\} = 0$
- (ii) $\#\{1, 2, 3, 4\} = 4$
- (iii) $\#\{x \mid x \in \mathbb{N} \text{ and } x < 100\} = 99$

- (iv) $\#\{2, 4, 6, 8, 10\} = 5$
- (v) $\#\{1, 2, 3, \dots, 500\} = 500$
- (vi) $\#\mathbb{Z}_7 = \#\{0, 1, 2, 3, 4, 5, 6\} = 7$
- (vii) $\#\mathbb{Z}_7^\otimes = \#\{1, 2, 3, 4, 5, 6\} = 6$

Example 9.1.9. We give the cardinality of some of the sets from Definition 5.4.1. Let $n \in \mathbb{N}$. Then

- (i) $\#\mathbb{Z}_n = \#\{0, 1, 2, \dots, n-1\} = n$.
- (ii) $\#\mathbb{Z}_n^\otimes = \#\{1, 2, \dots, n-1\} = n-1$.

9.2 Infinite Sets

We have not addressed the cardinalities of the set of integers and the set of natural numbers. Before we address this issue, we define what we mean by finite and infinite sets.

Definition 9.2.1. Let A be a set. The set A is *finite* means that $A = \{\}$ or that there exists $n \in \mathbb{N}$ and an invertible function $f : A \rightarrow \mathbb{Z}_n$.

The set A is *infinite* means that it is not finite.

To show that a non-empty set A is finite we find an $n \in \mathbb{N}$ such that there is an invertible function from A to $\mathbb{Z}_n$.

To show that a non-empty set B is infinite, we need to show that there is no such n that will work. We do this by showing that whichever n we pick, we find that it is too small. That is if we choose any finite subset S of B with $\#S = n$ elements, there is an element of B that is not in S . Then B is infinite. In the formulation of the criterion, we do not need to mention the number n , it simply is the cardinality of S .

Theorem 9.2.2. Let B be a set. If for each finite subset S of B there is an element $x \in B$ with $x \notin S$, then B is infinite.

Example 9.2.3. We show that the set of natural numbers $\mathbb{N}$ is infinite.

Let S be a finite subset of $\mathbb{N}$. Let b be the greatest of the elements of S . Then $b+1$ is not an element of S but it is an element of $\mathbb{N}$. In this way we can find an element of $\mathbb{N}$ that is not in S for any finite subset of S of $\mathbb{N}$. Thus by Theorem 9.2.2, the set of natural numbers $\mathbb{N}$ is infinite.

Definition 9.1.3 is formulated for any two sets. So this also allows us to determine when two infinite sets have the same cardinality.

Definition 9.2.4. The set S is *countable* means that S has the same cardinality as $\mathbb{N}$.

This yields surprising results. We show that the set of natural numbers $\mathbb{N}$ and the set of integers $\mathbb{Z}$ have the same cardinality, which means that $\mathbb{Z}$ is countable.

Example 9.2.5. Consider the function $f : \mathbb{N} \rightarrow \mathbb{Z}$ given by

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	...
$f(x)$	0	1	-1	2	-2	3	-3	4	-4	5	-5	6	-6	7	...

It is not difficult to see that the function f is invertible. Thus $\mathbb{N}$ and $\mathbb{Z}$ have the same cardinality. This also means that $\mathbb{Z}$ is countable.

We end with remarking that not all infinite sets are countable. For example the real numbers are not countable. In the following theorem we give another example of a set that is not countable. The existence of such a set means that there are different kinds of infinity.

Theorem 9.2.6. *The set S of subsets of the set $\mathbb{N}$ of natural numbers is not countable.*

9.3 Cardinality of Cartesian Products

We continue our discussion of Cartesian products by providing a formula for the cardinality of a Cartesian product in terms of the cardinalities of the sets from which it is constructed.

Theorem 9.3.1. *Let A and B be finite sets. Then, $\#(A \times B) = \#A \cdot \#B$.*

Proof. Let $a \in A$. The number of pairs of the form (a, b) where $b \in B$ is $\#B$. Since there are $\#B$ choices for b for each of the $\#A$ choices for $a \in A$ the number of elements in $A \times B$ is $\#A \cdot \#B$. $\square$

Example 9.3.2. We give examples for the number of elements in Cartesian products.

- (i) For any finite set A , we have that $\#(A \times \{\}) = \#A \cdot \#\{\} = \#A \cdot 0 = 0$.
- (ii) Let $A = \{-4, -3, -2, -1, 0, 1, 2, 3, 4\}$. Then, $\#(A \times A) = \#A \cdot \#A = 9 \cdot 9 = 81$.
- (iii) Let $A = \{0, 1, 2\}$ and $B = \{0, 1, 2, 3, 4\}$. Then, $\#(A \times B) = \#A \cdot \#B = 3 \cdot 5 = 15$.

Knowing the cardinality of a Cartesian product helps us to verify that we have listed all of the elements of the Cartesian product. The following example demonstrates this by revisiting the Cartesian products introduced in Example 6.2.3.

Example 9.3.3. Let $A = \{0, 1\}$, and let $B = \{4, 5, 6\}$. Then, $\#A = 2$ and $\#B = 3$. By Theorem 9.3.1, $\#(A \times B) = \#A \cdot \#B = 2 \cdot 3 = 6$ and $\#(B \times A) = \#B \cdot \#A = 3 \cdot 2 = 6$. In Example 6.2.3, we explicitly gave that

$$A \times B = \{(0, 4), (0, 5), (0, 6), (1, 4), (1, 5), (1, 6)\},$$

and

$$B \times A = \{(4, 0), (4, 1), (5, 0), (5, 1), (6, 0), (6, 1)\}.$$

Notice that there are, in fact, 6 elements in $A \times B$ and in $B \times A$, so we may say with confidence that we listed all of the elements in those Cartesian products.

9.4 Number of Subsets

We want to develop a formula for the number of distinct subsets of a given set. We begin by considering a few examples to help us develop a pattern.

Example 9.4.1. We list all distinct subsets of certain sets and count these subsets to find the number of distinct subsets.

- (i) The only subset of $\{\}$ is $\{\}$. Thus $\{\}$ has one subset.
- (ii) The subsets of $\{1\}$ are $\{\}$ and $\{1\}$. Thus $\{1\}$ has two distinct subsets.
- (iii) The subsets of $\{1, 2\}$ are $\{\}$, $\{1\}$, $\{2\}$, and $\{1, 2\}$. Thus $\{1, 2\}$ has four distinct subsets.
- (iv) The subsets of $\{1, 2, 3\}$ are $\{\}$, $\{1\}$, $\{2\}$, $\{3\}$, $\{1, 2\}$, $\{1, 3\}$, $\{2, 3\}$, and $\{1, 2, 3\}$. Thus $\{1, 2, 3\}$ has eight distinct subsets.

In order to develop the desired formula for the number of distinct subsets of a given set, let us systematically consider how we can build up the lists of subsets for the sets in Example 9.4.1.

The set with no elements: We have already seen that $\{\}$ has the single subset $\{\}$.

Sets with one element: If we put in one element, say a_1 , to our set and consider the new set $\{a_1\}$, we still have the subset $\{\}$ from the previous set. However, we also have a new subset – the subset we get by including in that new element a_1 to the existing subset. The new subset is $\{a_1\}$, giving us the two distinct subsets $\{\}$ and $\{a_1\}$.

Sets with two elements: Now, we do the same for a set with two elements. We consider the new set $\{a_1, a_2\}$. Just as the set $\{a_1\}$ the set $\{a_1, a_2\}$ has the subsets $\{\}$ and $\{a_1\}$. However, we also have two new subsets, namely the subsets we get by putting the new element a_2 into the existing two subsets. The new subsets are $\{a_2\}$ and $\{a_1, a_2\}$, giving us the four distinct subsets $\{\}$, $\{a_1\}$, $\{a_2\}$, and $\{a_1, a_2\}$.

Sets with three elements: For clarity, we will do this one more time. If we put in the element a_3 and consider the new set $\{a_1, a_2, a_3\}$, we still have the four subsets $\{\}$, $\{a_1\}$, $\{a_2\}$, and $\{a_1, a_2\}$ from the previous set. However, we also have four new subsets – the subsets we get by putting the new element a_3 into the existing four subsets. The new subsets are $\{a_3\}$, $\{a_1, a_3\}$, $\{a_2, a_3\}$, and $\{a_1, a_2, a_3\}$, giving us the eight distinct subsets

$$\{\}, \{a_1\}, \{a_2\}, \{a_1, a_2\}, \{a_3\}, \{a_1, a_3\}, \{a_2, a_3\}, \text{ and } \{a_1, a_2, a_3\}.$$

Notice that each time we put in an extra element, we double the number of distinct subsets. We now show that this observation is true independent of the number of elements in the set without the extra element. We start with a set with n elements (for some $n \in \mathbb{N}$) and show that a set with $n + 1$ elements has twice as many distinct subsets.

Sets with n elements: Let A be a set that contains the n distinct elements $a_1, a_2, a_3, \dots, a_n$, that is, $A = \{a_1, a_2, a_3, \dots, a_n\}$. Assume we know that A has the m subsets $S_1, \dots, S_m$. Let a_{n+1} be an object not contained in A . From the list $S_1, \dots, S_m$ of subsets of A , we construct all subsets of the set $\{a_1, \dots, a_n, a_{n+1}\}$. The subsets of $\{a_1, \dots, a_{n+1}\}$ that do not contain a_{n+1} are $S_1, \dots, S_m$. So it remains to consider all subsets of $\{a_1, \dots, a_n, a_{n+1}\}$ that contain a_{n+1} . Let $T_1, \dots, T_m$ be the sets obtained by including the element a_{n+1} to $S_1, \dots, S_m$, respectively. Now, the $2 \cdot m$ sets $S_1, \dots, S_m, T_1, \dots, T_m$ are all subsets of A .

Since the empty set has one subset and each additional element doubles the number of subsets, a set with n elements has

$$1 \cdot \underbrace{2 \cdot 2 \cdot \dots \cdot 2}_{n \text{ times}} = 2^n$$

distinct subsets. We have proven the following result.

Theorem 9.4.2. *The number of distinct subsets of a set A is $2^{\#A}$.*

Problem 9.4.3. *Find the number of distinct subsets of $\mathbb{Z}_{11}^{\otimes}$.*

Solution. We have $\mathbb{Z}_{11}^{\otimes} = \{1, 2, 3, \dots, 10\}$. So $\#\mathbb{Z}_{11}^{\otimes} = 10$. By the formula in Theorem 9.4.2 the number of distinct subsets of $\mathbb{Z}_{11}^{\otimes}$ is

$$2^{\#\mathbb{Z}_{11}^{\otimes}} = 2^{10} = 1024.$$

The formulas for the number of distinct subsets also have more practical applications.

Problem 9.4.4. *Mario's Pizza offers the following toppings: onions, mushrooms, peppers, olives, and spinach. How many different types of pizza can be ordered using these toppings?*

Solution. Let $T = \{\text{onions, mushrooms, peppers, olives, spinach}\}$ be the set of toppings. Each pizza would be made using a subset of these toppings, so the number of different types of pizza that can be ordered would correspond to the number of distinct subsets of the set T of toppings. We have $\#T = 5$. So T has $2^{\#T} = 2^5 = 32$ subsets. Thus 32 different pizzas can be ordered using these toppings.

Problem 9.4.5. *A person can order a new car with some, all, or none of the following options: air conditioning, power windows, satellite radio, leather interior, bluetooth connectivity, and sun roof. How many different variations of the set of options are possible?*

Solution. Let O be the set of options. Since $\#O = 6$, there are $2^6 = 64$ distinct subsets of O . So, 64 different variations of the set of options are possible.

Chapter 10

Primes

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Produce a list of all prime numbers up to a given bound.
- (2) Compute the prime factorization of a number.
- (3) Show whether a number is prime.
- (4) Show that there are infinitely many prime numbers.
- (5) Show that twin primes exist.
- (6) Reproduce the twin prime conjecture.

Prime numbers are a cornerstone of mathematics. Their lack of divisors makes them valuable numbers to work with. In this section we introduce the fundamental concepts of primality and prime factorization. We show that there are infinitely many prime numbers and present an important conjecture about primes.

10.1 Definition of a Prime

Recall that for two integers a and b , b divides a means that b is a factor of a (see Definition 4.1.1). Every positive integer, n , has the property that both 1 and n are factors of n . Prime numbers have only these factors.

Definition 10.1.1. An integer $p > 1$ is *prime*¹ means that the only positive factors of p are 1 and p .

An integer greater than 1 that is not prime is called *composite*.

A number is composite if it is not a prime number. As a prime number is only divisible by 1 and itself, a composite number n has at least one other factor a (that is not 1 or n). Now as n is not prime, $a \neq n$ and $a \neq 1$ so also $n \operatorname{div} a \neq n$ and $n \operatorname{div} a \neq 1$. Let $b := n \operatorname{div} a$. So we can say that an integer $n > 1$ is composite if it can be written as $n = a \cdot b$, where a and b are integers greater than 1.

Example 10.1.2. We give examples of prime numbers and composite numbers.

- (i) The first 11 primes numbers are

2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31

¹For a history of the choice not to consider the number 1 a prime number see: Chris K. Caldwell and Yeng Xiong, *What is the Smallest Prime?*, Journal of Integer Sequences (2012), <https://cs.uwaterloo.ca/journals/JIS/VOL15/Caldwell11/cald5.pdf>

(ii) We list the first 8 composite numbers along with a representation as a product.

$$4 = 2 \cdot 2, 6 = 2 \cdot 3, 8 = 2 \cdot 4, 9 = 3 \cdot 3, 10 = 2 \cdot 5, 12 = 2 \cdot 6, 14 = 2 \cdot 7, 15 = 3 \cdot 5.$$

The representation of composite numbers as products are not always unique, for example we have $12 = 2 \cdot 6$ and also $12 = 3 \cdot 4$.

The *Sieve of Eratosthenes* is a method for finding all primes up to (and possibly including) a given integer bound $n > 1$. This method works well for relatively small bounds, allowing us to determine whether any natural number less than or equal to the bound is prime or composite. We provide the steps of the Sieve of Eratosthenes with integer bound $n > 1$ here: (recall that a is a multiple of b means that b divides a , see Definition 4.1.1)

- (1) List all integers from 2 to n .
- (2) The first integer on the list is 2, and it is prime. Mark out all multiples of 2 that are bigger than 2 because they are composite.
- (3) The next integer on the list that is not marked out is 3, and it is prime. Mark out all multiples of 3 that are bigger than 3 because they are composite. (Note that some of these, such as 6, will already be marked out).
- (4) The next integer on the list that is not marked out is 5, and it is prime. Mark out all multiples of 5 that are bigger than 5 because they are composite.
- (:) Continue in this way until there is no next integer on the list that is not marked out. Conclude that the integers that are not marked out are all of the primes up to (and possibly including) the integer bound n .

In Figures 10.1.1 and 10.1.2, we demonstrate the initial steps as well as the end result of the Sieve of Eratosthenes with integer bound 100. There are 25 primes up to 100.

10.2 Prime Factorization

Most of the results of unique prime factorization were already contained in Euclid's *Elements*². An early modern formulation of the result can be found in Gauss *Disquisitiones arithmeticae*³.

Theorem 10.2.1 (Fundamental Theorem of Arithmetic). *Every integer greater than 1 can be written uniquely as a prime number or a product of prime numbers.*

The unique representation of each integer greater than 1 that is guaranteed by the Fundamental Theorem of Arithmetic (Theorem 10.2.1) is called the *prime factorization* of the integer. For composites, the prime factorization may include multiple copies of the same prime. If so, exponents are typically used to condense the prime factorization. This is demonstrated in the following example.

Example 10.2.2. We provide the prime factorization for sample integers greater than 1.

²euclid.

³gauss.

Figure 10.1.1: Sieve of Eratosthenes up to 100 with (a) no composites removed and (b) multiples of 2 removed.

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

(a) Grid of all integers from 2 to 100

	2	3	4 Mult. of 2	5	6 Mult. of 2	7	8 Mult. of 2	9	10 Mult. of 2
11	12 Mult. of 2	13	14 Mult. of 2	15	16 Mult. of 2	17	18 Mult. of 2	19	20 Mult. of 2
21	22 Mult. of 2	23	24 Mult. of 2	25	26 Mult. of 2	27	28 Mult. of 2	29	30 Mult. of 2
31	32 Mult. of 2	33	34 Mult. of 2	35	36 Mult. of 2	37	38 Mult. of 2	39	40 Mult. of 2
41	42 Mult. of 2	43	44 Mult. of 2	45	46 Mult. of 2	47	48 Mult. of 2	49	50 Mult. of 2
51	52 Mult. of 2	53	54 Mult. of 2	55	56 Mult. of 2	57	58 Mult. of 2	59	60 Mult. of 2
61	62 Mult. of 2	63	64 Mult. of 2	65	66 Mult. of 2	67	68 Mult. of 2	69	70 Mult. of 2
71	72 Mult. of 2	73	74 Mult. of 2	75	76 Mult. of 2	77	78 Mult. of 2	79	80 Mult. of 2
81	82 Mult. of 2	83	84 Mult. of 2	85	86 Mult. of 2	87	88 Mult. of 2	89	90 Mult. of 2
91	92 Mult. of 2	93	94 Mult. of 2	95	96 Mult. of 2	97	98 Mult. of 2	99	100 Mult. of 2

(b) Multiples of 2 (except 2) removed

Figure 10.1.2: Sieve of Eratosthenes up to 100 with (c) multiples of 2 and 3 removed and (d) all composites removed, leaving only primes.

	2	3	4 Mult. of 2	5	6 Mult. of 2	7	8 Mult. of 2	9 Mult. of 3	10 Mult. of 2
11	12 Mult. of 2	13	14 Mult. of 2	15 Mult. of 3	16 Mult. of 2	17	18 Mult. of 2	19	20 Mult. of 2
21 Mult. of 3	22 Mult. of 2	23	24 Mult. of 2	25	26 Mult. of 2	27 Mult. of 3	28 Mult. of 2	29	30 Mult. of 2
31	32 Mult. of 2	33 Mult. of 3	34 Mult. of 2	35	36 Mult. of 2	37	38 Mult. of 2	39 Mult. of 3	40 Mult. of 2
41	42 Mult. of 2	43	44 Mult. of 2	45 Mult. of 3	46 Mult. of 2	47	48 Mult. of 2	49	50 Mult. of 2
51 Mult. of 3	52 Mult. of 2	53	54 Mult. of 2	55	56 Mult. of 2	57 Mult. of 3	58 Mult. of 2	59	60 Mult. of 2
61	62 Mult. of 2	63 Mult. of 3	64 Mult. of 2	65	66 Mult. of 2	67	68 Mult. of 2	69 Mult. of 3	70 Mult. of 2
71	72 Mult. of 2	73	74 Mult. of 2	75 Mult. of 3	76 Mult. of 2	77	78 Mult. of 2	79	80 Mult. of 2
81 Mult. of 3	82 Mult. of 2	83	84 Mult. of 2	85	86 Mult. of 2	87 Mult. of 3	88 Mult. of 2	89	90 Mult. of 2
91	92 Mult. of 2	93 Mult. of 3	94 Mult. of 2	95	96 Mult. of 2	97	98 Mult. of 2	99 Mult. of 3	100 Mult. of 2

(c) Multiples of 2 and 3 (except 2 and 3) removed

	2	3	4 Mult. of 2	5	6 Mult. of 2	7	8 Mult. of 2	9 Mult. of 3	10 Mult. of 2
11	12 Mult. of 2	13	14 Mult. of 2	15 Mult. of 3	16 Mult. of 2	17	18 Mult. of 2	19	20 Mult. of 2
21 Mult. of 3	22 Mult. of 2	23	24 Mult. of 2	25 Mult. of 5	26 Mult. of 2	27 Mult. of 3	28 Mult. of 2	29	30 Mult. of 2
31	32 Mult. of 2	33 Mult. of 3	34 Mult. of 2	35 Mult. of 5	36 Mult. of 2	37	38 Mult. of 2	39 Mult. of 3	40 Mult. of 2
41	42 Mult. of 2	43	44 Mult. of 2	45 Mult. of 3	46 Mult. of 2	47	48 Mult. of 2	49 Mult. of 7	50 Mult. of 2
51 Mult. of 3	52 Mult. of 2	53	54 Mult. of 2	55 Mult. of 5	56 Mult. of 2	57 Mult. of 3	58 Mult. of 2	59	60 Mult. of 2
61	62 Mult. of 2	63 Mult. of 3	64 Mult. of 2	65 Mult. of 5	66 Mult. of 2	67	68 Mult. of 2	69 Mult. of 3	70 Mult. of 2
71	72 Mult. of 2	73	74 Mult. of 2	75 Mult. of 3	76 Mult. of 2	77 Mult. of 7	78 Mult. of 2	79	80 Mult. of 2
81 Mult. of 3	82 Mult. of 2	83	84 Mult. of 2	85 Mult. of 5	86 Mult. of 2	87 Mult. of 3	88 Mult. of 2	89	90 Mult. of 2
91 Mult. of 7	92 Mult. of 2	93 Mult. of 3	94 Mult. of 2	95 Mult. of 5	96 Mult. of 2	97	98 Mult. of 2	99 Mult. of 3	100 Mult. of 2

(d) Completed sieve for integers up to 100

- (i) $100 = 2 \cdot 2 \cdot 5 \cdot 5 = 2^2 \cdot 5^2$
- (ii) $7007 = 7 \cdot 7 \cdot 11 \cdot 13 = 7^2 \cdot 11 \cdot 13$
- (iii) $23498357349 = 3 \cdot 53 \cdot 397 \cdot 372263$

In order to arrive at the prime factorization of a composite n , we can begin by simply breaking the composite n into the product of two integers a and b that are each greater than 1. We write $n = a \cdot b$ and then take a closer look at whether or not a and b are prime. If a and b are prime, we are essentially done. However, if either a or b is composite, we must continue the process by finding factors of the composite number(s). But, note that it will always be true that a and b are both less than n , so progress has been made.

Problem 10.2.3. *Give the prime factorization of 18810.*

Solution. We present a solution that explains some of the “figuring” that might be useful in determining the prime factorization. Since 18810 ends in a 0, we automatically know that 10 is a factor, and we write

$$18810 = 10 \cdot 1881.$$

Now, 10 is definitely composite, so we can further break it down as $10 = 2 \cdot 5$ to get

$$18810 = 2 \cdot 5 \cdot 1881.$$

It is less clear whether 1881 is prime or composite. Since 1881 is an odd number, it does not have a factor of 2. We move onto the next prime number, 3, and see if that is a factor of 1881. It turns out that it is.

A quick check to determine whether or not a number is divisible by 3 is to add up the digits of the number and determine whether or not that sum is divisible by 3. For the case of 1881, we have that $1 + 8 + 8 + 1 = 18$. It is easier to see that the sum, 18, is divisible by 3. But if that is not obvious, we can continue to sum the digits to get that $1 + 8 = 9$, which is clearly divisible by 3.

Since 1881 is divisible by 3, we can compute that $1881 = 3 \cdot 627$. We now have

$$18810 = 2 \cdot 5 \cdot 3 \cdot 627.$$

We can perform the same summing digits check mentioned above to determine that 627 is also divisible by 3. Since $627 = 3 \cdot 209$, we now have

$$18810 = 2 \cdot 5 \cdot 3 \cdot 3 \cdot 209.$$

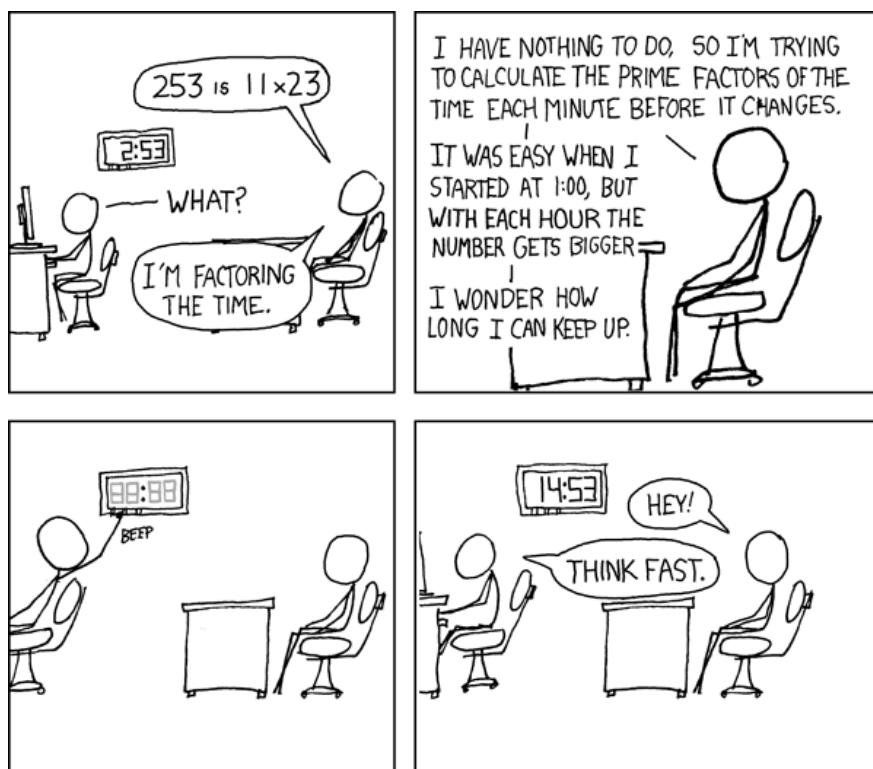
Summing the digits of 209 tells us that 209 is not divisible by 3 since $2 + 0 + 9 = 11$ is not divisible by 3. So, we move onto the next prime number, 5. Since 209 does not end in 0 or 5, we conclude that 5 is not a factor. We continue moving through the prime numbers and conclude that 7 is not a factor of 209 but that 11 is. Since $209 = 11 \cdot 19$, we now have

$$18810 = 2 \cdot 5 \cdot 3 \cdot 3 \cdot 11 \cdot 19.$$

Since 19 is also a prime, we have now found all of the prime factors of 18810 and how many times they each occur. However, we typically rearrange the prime factors into non-decreasing order and use exponents to condense the prime factorization. Our final answer is

$$18810 = 2 \cdot 3^2 \cdot 5 \cdot 11 \cdot 19.$$

Figure 10.2.1: *Factoring the Time* by R. Munroe (<https://xkcd.com/247>).



I occasionally do this with mile markers on the highway.

There are many possible paths to finding the prime factorization of a number. Because the Fundamental Theorem of Arithmetic (Theorem 10.2.1) guarantees the uniqueness of the prime factorization, the order in which we find the factors does not matter. Based on the solution given for Problem 10.2.3, it is apparent that finding prime factorizations of composite numbers can involve many steps. Generally speaking, the bigger the composite number is, the harder it is to find its prime factorization. It could be true that a big composite number has lots of different prime factors, lots of repeated prime factors, or even big prime factors. The next result provides us with a tool that will help us determine when we can say with confidence that we have completed a prime factorization of a composite.

Theorem 10.2.4. *If n is composite, then n has a prime factor less than or equal to $\sqrt{n}$.*

Proof. A composite n can be written as $n = a \cdot b$, where a and b are integers greater than 1. If a and b are both greater than $\sqrt{n}$, then $n = a \cdot b > \sqrt{n} \cdot \sqrt{n} = n$ would have to be true. However, $n > n$ is false, so either a or b must be less than or equal to $\sqrt{n}$. All prime factors of a are less than or equal to a , and these are also prime factors of n . If $a \leq \sqrt{n}$, then n has a prime factor less than or equal to $\sqrt{n}$. The analogous argument holds if $b \leq \sqrt{n}$. $\square$

A consequence of Theorem 10.2.4 is that if an integer $n > 1$ does not have a prime factor less than or equal to $\sqrt{n}$, then n is prime. This understanding of Theorem 10.2.4 gives us a tool to verify that an integer $n > 1$ is prime without exhaustively checking that each integer $2, 3, \dots, n - 1$ fails to be a factor.

To conclude that a natural number $n > 1$ is prime, we only need to know that $n \bmod p \neq 0$ for each prime $p \leq \sqrt{n}$. We do not actually need to know the exact value of each $n \bmod p$, we just need to know that the value is not zero. So, if we do not get an integer when we divide n by p , then we automatically know that $n \bmod p \neq 0$, and that is sufficient.

We demonstrate this use of Theorem 10.2.4 in the next example. This method of showing that a number is prime is called *trial division*.

Problem 10.2.5. *Prove that 523 is prime.*

Solution. By Theorem 10.2.4 it suffices to show that no prime less than or equal to $\sqrt{523} = 22.869\dots$ is a factor of 523. The primes less than or equal to $\sqrt{523}$ are 2, 3, 5, 7, 11, 13, 17, and 19. We compute

$$\begin{aligned} 523 \bmod 2 &= 1 \\ 523 \bmod 3 &= 1 \\ 523 \bmod 5 &= 3 \\ 523 \bmod 7 &= 5 \\ 523 \bmod 11 &= 6 \\ 523 \bmod 13 &= 3 \\ 523 \bmod 17 &= 13 \\ 523 \bmod 19 &= 10. \end{aligned}$$

Since none of the above values is 0, we conclude that none of the primes less than or equal to $\sqrt{523}$ is a factor of 523. Thus, 523 is prime.

When two numbers are given in factored form their greatest common divisor can be composed from the common factors of the two numbers.

Example 10.2.6. Let integers a and b be given in factored form:

$$a := 2^7 \cdot 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \text{ and } b = 3^7 \cdot 11^4 \cdot 13^5 \cdot 17$$

The greatest common divisor is the product of all common factors (including multiplicity). So $\gcd(a, b) = 3 \cdot 11$.

Example 10.2.7. Let integers a and b be given in factored form:

$$a := 2^3 \cdot 3^2 \cdot 7^5 \cdot 19^2 \text{ and } b = 2^5 \cdot 7^4 \cdot 19^5 \cdot 23$$

The greatest common divisor is the product of all common factors (including multiplicity). So $\gcd(a, b) = 2^3 \cdot 7^4 \cdot 19^2$.

10.3 Infinitude of Primes

In Example 9.2.3 we saw that there are infinitely many natural numbers. Certainly, not every natural number is prime because there are composites, too. However, an ancient number theory result⁴ asserts that there are still infinitely many primes.

⁴euclid.

Figure 10.3.1: All prime numbers less than 1660

2	3	5	7	11	13	17	19	23	29	31	37	41
43	47	53	59	61	67	71	73	79	83	89	97	101
103	107	109	113	127	131	137	139	149	151	157	163	167
173	179	181	191	193	197	199	211	223	227	229	233	239
241	251	257	263	269	271	277	281	283	293	307	311	313
317	331	337	347	349	353	359	367	373	379	383	389	397
401	409	419	421	431	433	439	443	449	457	461	463	467
479	487	491	499	503	509	521	523	541	547	557	563	569
571	577	587	593	599	601	607	613	617	619	631	641	643
647	653	659	661	673	677	683	691	701	709	719	727	733
739	743	751	757	761	769	773	787	797	809	811	821	823
827	829	839	853	857	859	863	877	881	883	887	907	911
919	929	937	941	947	953	967	971	977	983	991	997	1009
1013	1019	1021	1031	1033	1039	1049	1051	1061	1063	1069	1087	1091
1093	1097	1103	1109	1117	1123	1129	1151	1153	1163	1171	1181	1187
1193	1201	1213	1217	1223	1229	1231	1237	1249	1259	1277	1279	1283
1289	1291	1297	1301	1303	1307	1319	1321	1327	1361	1367	1373	1381
1399	1409	1423	1427	1429	1433	1439	1447	1451	1453	1459	1471	1481
1483	1487	1489	1493	1499	1511	1523	1531	1543	1549	1553	1559	1567
1571	1579	1583	1597	1601	1607	1609	1613	1619	1621	1627	1637	1657

Theorem 10.3.1. *There are infinitely many primes.*

Proof. Let $\mathbb{P}$ denote the set of all prime numbers. We show that for any finite subset Q of $\mathbb{P}$ there is an element in $\mathbb{P}$ that is not an element of the finite subset Q .

Let Q be a finite subset of the set $\mathbb{P}$. Denote the elements of Q by $p_1, p_2, \dots, p_n$ and let $q = p_1 \cdot p_2 \cdot \dots \cdot p_n$.

By Theorem 4.3.4 q and $q+1$ are coprime. So there is at least one prime number that divides $q+1$ but does not divide q . Call that prime number t . Then $t \notin Q$.

As we can find such a prime number t for every finite subset of $\mathbb{P}$, the set $\mathbb{P}$ is infinite by Theorem 9.2.2. $\square$

10.4 The Twin Prime Conjecture

It was relatively easy to prove that there are infinitely many primes (Theorem 10.3.1). In order to come up with a new mathematical result, a great deal of study, investigation, and insight is often required. Ideas arise, steps toward a proof are taken, and sometimes those ideas have to be tweaked. In this process, it is possible to develop a statement that is believed to be true but has not been formally proven. Such a statement is called a *conjecture* and

Figure 10.3.2: All prime numbers greater than 1660 and less than 3728

1663	1667	1669	1693	1697	1699	1709	1721	1723	1733	1741	1747	1753
1759	1777	1783	1787	1789	1801	1811	1823	1831	1847	1861	1867	1871
1873	1877	1879	1889	1901	1907	1913	1931	1933	1949	1951	1973	1979
1987	1993	1997	1999	2003	2011	2017	2027	2029	2039	2053	2063	2069
2081	2083	2087	2089	2099	2111	2113	2129	2131	2137	2141	2143	2153
2161	2179	2203	2207	2213	2221	2237	2239	2243	2251	2267	2269	2273
2281	2287	2293	2297	2309	2311	2333	2339	2341	2347	2351	2357	2371
2377	2381	2383	2389	2393	2399	2411	2417	2423	2437	2441	2447	2459
2467	2473	2477	2503	2521	2531	2539	2543	2549	2551	2557	2579	2591
2593	2609	2617	2621	2633	2647	2657	2659	2663	2671	2677	2683	2687
2689	2693	2699	2707	2711	2713	2719	2729	2731	2741	2749	2753	2767
2777	2789	2791	2797	2801	2803	2819	2833	2837	2843	2851	2857	2861
2879	2887	2897	2903	2909	2917	2927	2939	2953	2957	2963	2969	2971
2999	3001	3011	3019	3023	3037	3041	3049	3061	3067	3079	3083	3089
3109	3119	3121	3137	3163	3167	3169	3181	3187	3191	3203	3209	3217
3221	3229	3251	3253	3257	3259	3271	3299	3301	3307	3313	3319	3323
3329	3331	3343	3347	3359	3361	3371	3373	3389	3391	3407	3413	3433
3449	3457	3461	3463	3467	3469	3491	3499	3511	3517	3527	3529	3533
3539	3541	3547	3557	3559	3571	3581	3583	3593	3607	3613	3617	3623
3631	3637	3643	3659	3671	3673	3677	3691	3697	3701	3709	3719	3727

is often known in mathematics as an *open problem*. We conclude this section by presenting an important conjecture involving primes. While the statement of the conjecture is easy to understand and computer experiments have not come up with a counterexample, we do not know whether it is true.

Conjecture 10.4.1 (Twin Prime Conjecture). *There are infinitely many primes p such that $p + 2$ is also prime.*

If p and $p + 2$ are both prime, then $(p, p + 2)$ is called a *twin prime pair*. The Twin Prime Conjecture (Conjecture 10.4.1) is the claim that there are infinitely many twin prime pairs.

Example 10.4.2. The first few twin prime pairs are

$$(3, 5), (5, 7), (11, 13), (17, 19), \dots$$

Problem 10.4.3. *Determine whether or not each prime is a part of a twin prime pair.*

- (i) 89
- (ii) 137

Solution. For each given prime p , we must determine whether or not either $p - 2$ or $p + 2$ is prime to make our conclusion.

- (i) By the completed Sieve of Eratosthenes given in Figure 10.1.2 (d), we see that neither $89 - 2 = 87$ nor $89 + 2 = 91$ is prime. So, 89 is not a part of a twin prime pair.

- (ii) Since the prime 137 is beyond the bound for our completed Sieve of Eratosthenes, we have to work a bit harder on this problem. We begin by considering the integer $137 - 2 = 135$. Since 5 is a factor of 135, we conclude that 135 is not prime. Now, we consider the integer $137 + 2 = 139$. We use Theorem 10.2.4 to determine whether or not 139 is prime by checking whether or not each prime less than or equal to $\sqrt{139} = 11.7898\dots$ is a factor of 139. The primes less than or equal to $\sqrt{139}$ are 2, 3, 5, 7, and 11. We compute

$$139 \bmod 2 = 1$$

$$139 \bmod 3 = 1$$

$$139 \bmod 5 = 4$$

$$139 \bmod 7 = 6$$

$$139 \bmod 11 = 7.$$

Since none of the above values is 0, we conclude that none of the primes less than or equal to $\sqrt{139}$ is a factor of 139. Thus 139 is prime, and 137 is a part of the twin prime pair (137, 139).

Chapter 11

Other Bases

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Rewrite numbers in expanded form.
- (2) Convert numbers in other bases to decimal numbers.
- (3) Convert decimal numbers to other bases.

We begin this section by recalling how numbers are represented in the familiar decimal system (base 10). Next we consider binary (base 2) numbers and the conversion between binary and decimal numbers. Then, we generalize the representation of numbers to bases other than 2 and 10. Finally, we present an algorithm that converts a (base 10) natural number to a numeral in a different base.

11.1 Decimal Numbers

In the *decimal* system, every number is written with the 10 digits

0, 1, 2, 3, 4, 5, 6, 7, 8, and 9.

The value of each digit depends on its location. The right most digits are the ones, the second digit from the right are the 10s, the third digit from the right are the hundreds, the fourth digit from the right are the thousands and so on. When reading a number we multiply the right most digit by $1 = 10^0$, the the second digit from the right by $10 = 10^1$, the third digit by $100 = 10^2$, the fourth digit from the right by $1000 = 10^3$ and so on. We call 1, 10, 100, 1000, $\dots$ the *values of the places* of the digits. The value of the n -th digit from the right is 10^{n-1} (remember that the place value of the rightmost digit is $10^0 = 1$). Thus the values of the places of a number with n (decimal) digits are

$$10^{n-1}, 10^{n-2}, \dots, 10^5, 10^4, 10^3 = 1000, 10^2 = 100, 10^1 = 10, 10^0 = 1.$$

Each digit of a decimal number is an element of $\mathbb{Z}_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. In the following we denote the digits of a decimal number by $a_0, a_1, a_2, a_3, a_4, a_5, \dots, a_{n-1}$ arranged such that a_0 is the rightmost digit and the a_{n-1} is the leftmost digit. When we want to emphasize the value of the place of each digit we write a number in *base 10 expansion*:

$$a_{n-1}a_{n-2} \dots a_3a_2a_1a_0 = (a_{n-1} \cdot 10^{n-1}) + (a_{n-2} \cdot 10^{n-2}) + \dots + (a_3 \cdot 10^3) + (a_2 \cdot 10^2) + (a_1 \cdot 10^1) + (a_0 \cdot 10^0).$$

The digit a_0 is the first digit from the right and has the value 0, 1, 2, 3, 4, 5, 6, 7, 8, or 9, since it is in the “ones place”; the second digit from the right, which we called a_1 , has the

Figure 11.0.1: For selected numbers n , we give (a) the n in decimal (base 10) representation, (b) the digits of the decimal representation of n explicitly by place, and (c) the base 10 expansion of n . Recall that $10^1 = 10$ and $10^0 = 1$.

(a) n in base 10	(b) base 10 digits of n					(c) base 10 expansion of n
	10^4	10^3	10^2	10^1	10^0	
1					1	$1 \cdot 1$
10				1	0	$(1 \cdot 10) + (0 \cdot 1)$
100			1	0	0	$(1 \cdot 10^2) + (0 \cdot 10) + (0 \cdot 1)$
562			5	6	2	$(5 \cdot 10^2) + (6 \cdot 10) + (2 \cdot 1)$
2341		2	3	4	1	$(2 \cdot 10^3) + (3 \cdot 10^2) + (4 \cdot 10) + (1 \cdot 1)$
12004	1	2	0	0	4	$(1 \cdot 10^4) + (2 \cdot 10^3) + (0 \cdot 10^2) + (0 \cdot 10) + (4 \cdot 1)$
56784	5	6	7	8	4	$(5 \cdot 10^4) + (6 \cdot 10^3) + (7 \cdot 10^2) + (8 \cdot 10) + (4 \cdot 1)$

value 0, 10, 20, 30, 40, 50, 60, 70, 80, or 90, as it is in the “tens place”; the value of the third digit from the right (a_3) has the value 0, 100, 200, 300, 400, 500, 600, 700, 800, or 900, since it is in the “hundreds place”; and so on.

Example 11.1.1. We give the base 10 expansion of three numbers.

- (i) $562 = 5 \cdot 10^2 + 6 \cdot 10 + 2 \cdot 1 = 5 \cdot 100 + 6 \cdot 10 + 2 \cdot 1$
- (ii) $56200 = 5 \cdot 10^4 + 6 \cdot 10^3 + 2 \cdot 10^2 + 0 \cdot 10^1 + 0 \cdot 10^0 = 5 \cdot 10000 + 6 \cdot 1000 + 2 \cdot 100 + 0 \cdot 10 + 0 \cdot 1$
- (iii) $2001 = 2 \cdot 10^3 + 0 \cdot 10^2 + 0 \cdot 10 + 1 \cdot 1 = 2 \cdot 1000 + 0 \cdot 100 + 0 \cdot 10 + 1 \cdot 1$

See Figure 11.0.1 for further examples.

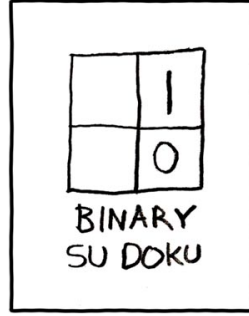
11.2 Binary Numbers

Before we move on to presenting numbers with arbitrary base b where b is a natural number greater than 2, we consider one more special case. One of the most common bases other than base 10 is base 2. While base 10 numbers are written with the ten symbols 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 numbers in base 2 are written using the two symbols 0 and 1. Base 2 numbers are of particular interest because digital devices (such as computers) work with two different states, for example off and on, which are represented by 0 and 1. To distinguish the binary numbers that only use two symbols from decimal numbers we add a little subscript 2.

The values of the places of base 10 numbers are the powers of 10, namely $10^0 = 1$, $10^1 = 10$, $10^2 = 100$, $10^3 = 1000$, $10^4 = 10000$, and so on. Similarly the place values of base 2 numbers are the powers of two, namely $2^0 = 1$, $2^1 = 2$, $2^2 = 4$, $2^3 = 8$, $2^4 = 16$ and so on. That means that given a binary number

$$a = (r_{n-1} \dots r_2 r_1 r_0)_2$$

Figure 11.2.1: *Su Doku* by R. Munroe (<https://xkcd.com/74>).



This one is from the Red Belt collection, of 'medium' difficulty

Figure 11.2.2: Binary (base 2) numbers, their base 2 digits, their base 2 expansion, and in base 10. The 2 digits used in binary numbers are 0 and 1.

n in base 2	base 2 digits of n				base 2 expansion of n	n in base 10
	2^3	2^2	2^1	2^0		
1_2				1	$1 \cdot 1$	1
10_2			1	0	$1 \cdot 2 + 0 \cdot 1$	2
100_2		1	0	0	$1 \cdot 2^2 + 0 \cdot 2 + 0 \cdot 1$	4
101_2		1	0	1	$1 \cdot 2^2 + 0 \cdot 2 + 1 \cdot 1$	5
1010_2	1	0	1	0	$1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2 + 0 \cdot 1$	10

where, for $i \in \{0, \dots, n-1\}$, we have $r_i \in \{0, 1\}$ its expanded binary (base 2) representation is

$$a = r_{n-1} \cdot 2^{n-1} + r_{n-2} \cdot 2^{n-2} + \dots + r_1 \cdot 2 + r_0.$$

As the place values of the n digits are

$$2^{n-1}, 2^{n-2}, \dots, 2^5 = 32, 2^4 = 16, 2^3 = 8, 2^2 = 4, 2^1 = 2, 2^0 = 1$$

we immediately obtain a method for converting base 2 numbers to base 10.

Example 11.2.1. We convert 100101_2 to base 10. We have

$$100101_2 = 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 32 + 4 + 1 = 37.$$

We have found that the decimal representation of the base 2 number 100101_2 is 37.

In Figure 11.2.2 we give more examples of numbers in base 2, their expanded base 2 representation and the number in decimal representation.

11.2.1 Counting in Base 2

When we start counting using only the two symbols 0 and 1. As in the case of decimal numbers we start with zero.

0_2 ,

Still with one digit we can also write the number one:

1_2

So with one digit we were able to count zero and one. As in the case of decimal numbers we add one more digit and obtain:

$10_2, 11_2$

With two digits we have counted to three. As we cannot go further with those two digits, we continue with:

$100_2, 101_2, 110_2, 111_2$

With three digits we have counted to seven. Adding one more digit we continue with:

$1000_2, 1001_2, 1010_2, 1011_2, 1100_2, 1101_2, 1110_2, 1111_2$

With 4 digits we have counted from zero to fifteen. By now the pattern is clear and we can keep counting like this indefinitely, adding one more digit when we have exhausted all combinations with the current number of digits.

Considering the numbers above we see that 10_2 is two, 100_2 is four, and 1000_2 is eight.

11.3 Conversion from Base 10 to Base 2

Now we investigate how we can convert a number in base 10 representation to base 2 representation. From the previous section it is evident that when we write a number in base 2 representation we write it as the sum of powers of 2.

In particular, for each power of two, we can compute whether or not that power is used in the sum by taking the exponent mod 2, then repeating the process with the exponent div 2.

Example 11.3.1. We compute 13 as a sum of powers of two without explicitly writing out and grouping. We will fill in the following table where the number in the n column is replaced by the number in the $n \div 2$ column for each following step. We stop when the number in the $n \div 2$ column becomes 0.

Step	Power of 2	n	$n \bmod 2$	$n \div 2$
0	2^0	13		

Filling in the last two columns we obtain:

Step	Power of 2	n	$n \bmod 2$	$n \div 2$
0	2^0	13	1	6

In the next step, $n = 13$ is replaced by $13 \div 2 = 6$.

Step	Power of 2	n	$n \bmod 2$	$n \operatorname{div} 2$
0	2^0	13	1	6
1	2^1	6	0	3

Continuing this process, we can fill out the rest of the table, and we stop when $n \operatorname{div} 2 = 0$.

Step	Power of 2	n	$n \bmod 2$	$n \operatorname{div} 2$
0	2^0	13	1	6
1	2^1	6	0	3
2	2^2	3	1	1
3	2^3	1	1	0

A one in the $n \bmod 2$ column tells us that the power of two in the corresponding row is used in writing our exponent as a sum of powers of two, and a zero in that column tells us that it is not. In other words, we have that $13 = (1 \cdot 2^0) + (0 \cdot 2^1) + (1 \cdot 2^2) + (1 \cdot 2^3)$.

Problem 11.3.2. Write 37 as a sum of powers of two.

Solution. We produce the table as in Example 11.3.1.

Step	Power of 2	n	$n \bmod 2$	$n \operatorname{div} 2$
0	2^0	37	1	18
1	2^1	18	0	9
2	2^2	9	1	4
3	2^3	4	0	2
4	2^4	2	0	1
5	2^5	1	1	0

So $37 = (1 \cdot 2^0) + (0 \cdot 2^1) + (1 \cdot 2^2) + (0 \cdot 2^3) + (0 \cdot 2^4) + (1 \cdot 2^5)$.

Another method for converting numbers to base 2 is used in the following algorithm.

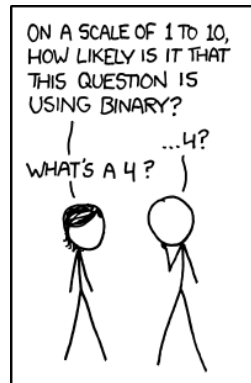
Algorithm 11.3.3 (*Conversion to Binary*).

Input: $a \in \mathbb{N}$

Output: The base 2 digits $r_0, \dots, r_m \in \mathbb{Z}_2$ of a such that $a = r_m \cdot 2^m + r_{m-1}2^{m-1} + \dots + r_3 \cdot 2^3 + r_2 \cdot 2^2 + r_1 \cdot 2 + r_0$

- (i) **let** $i := 0$
- (ii) **repeat**
 - (a) **let** $r_i := a \bmod 2$
 - (b) **let** $a := a \operatorname{div} 2$
 - (c) **let** $i := i + 1$
- (iii) **until** $a = 0$
- (iv) **return** $r_0, \dots, r_m$

Figure 11.3.1: *1 to 10* by R. Munroe (<https://xkcd.com/953>).



If you get an 11/100 on a CS test, but you claim it should be counted as a 'C', they'll probably decide you deserve the upgrade.

Example 11.3.4. We convert the base 10 number 13 to base 2 using Algorithm 11.5.1.

Input: $b = 2$, $a = 13$

$$\begin{array}{llll} i = 0 & r_0 = 13 \bmod 2 = 1 & a = 13 \operatorname{div} 2 = 6 \\ i = 1 & r_1 = 6 \bmod 2 = 0 & a = 6 \operatorname{div} 2 = 3 \\ i = 2 & r_2 = 3 \bmod 2 = 1 & a = 3 \operatorname{div} 2 = 1 \\ i = 3 & r_3 = 1 \bmod 2 = 1 & a = 1 \operatorname{div} 2 = 0 \end{array}$$

Output: $r_0 = 1$, $r_1 = 0$, $r_2 = 1$, $r_3 = 1$

The base 2 expansion of 13 is $13 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2 + 1 \cdot 1$. Thus the base 2 representation of 13 is 1101_2 .

11.4 Base b Numbers

Instead of using base 10, we can use any other natural number $b > 1$ as a base. To represent any number in base b , we must specify b unique symbols that represent the b values from 0 to $b - 1$. Those symbols are the first b symbols from the list

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, \dots$$

Note that if $b \leq 10$, we use the numbers $0, 1, 2, 3, \dots, b - 1$ as our b unique symbols. However, if $b > 10$, we use all of the numbers $0, 1, 2, \dots, 8, 9$ as well as enough capital letters to complete the list of b unique symbols. The value of A is the decimal number 10, the value of B is the decimal number 11, the value of C is the decimal number 12, and so on. We do not consider bases greater than 36, so we do not need further symbols. There are many applications of numbers in other bases. In particular, computer related fields frequently use base 2, 8, and 16.

Figure 11.4.1 provides various numbers written in base 2, 3, 8, 10, 12, and 16 as well as in English and French. When counting in some languages, there are some irregularities of

Figure 11.4.1: Selected numbers in English, French, and bases 2, 3, 8, 10, 12, 16

English	French	binary (base 2)	ternary (base 3)	octal (base 8)	decimal (base 10)	dozenal (base 12)	hexadecimal (base 16)
zero	zéro	0 ₂	0 ₃	0 ₈	0	0 ₁₂	0 ₁₆
one	un	1 ₂	1 ₃	1 ₈	1	1 ₁₂	1 ₁₆
two	deux	10 ₂	2 ₃	2 ₈	2	2 ₁₂	2 ₁₆
three	trois	11 ₂	10 ₃	3 ₈	3	3 ₁₂	3 ₁₆
four	quatre	100 ₂	11 ₃	4 ₈	4	4 ₁₂	4 ₁₆
five	cinq	101 ₂	12 ₃	5 ₈	5	5 ₁₂	5 ₁₆
six	six	110 ₂	20 ₃	6 ₈	6	6 ₁₂	6 ₁₆
seven	sept	111 ₂	21 ₃	7 ₈	7	7 ₁₂	7 ₁₆
eight	huit	1000 ₂	22 ₃	10 ₈	8	8 ₁₂	8 ₁₆
nine	neuf	1001 ₂	100 ₃	11 ₈	9	9 ₁₂	9 ₁₆
ten	dix	1010 ₂	101 ₃	12 ₈	10	A ₁₂	A ₁₆
eleven	onze	1011 ₂	102 ₃	13 ₈	11	B ₁₂	B ₁₆
twelve	douze	1100 ₂	110 ₃	14 ₈	12	10 ₁₂	C ₁₆
thirteen	treize	1101 ₂	111 ₃	15 ₈	13	11 ₁₂	D ₁₆
fourteen	quatorze	1110 ₂	112 ₃	16 ₈	14	12 ₁₂	E ₁₆
fifteen	quinze	1111 ₂	120 ₃	17 ₈	15	13 ₁₂	F ₁₆
sixteen	seize	10000 ₂	121 ₃	20 ₈	16	14 ₁₂	10 ₁₆
seventeen	dixsept	10001 ₂	122 ₃	21 ₈	17	15 ₁₂	11 ₁₆
twenty	vingt	10100 ₂	202 ₃	24 ₈	20	18 ₁₂	14 ₁₆
sixty	soixante	111100 ₂	2020 ₃	74 ₈	60	50 ₁₂	3C ₁₆
eighty	quatrevingt	1010000 ₂	2222 ₃	120 ₈	80	68 ₁₂	50 ₁₆
ninety	quatrevingt-dix	1011010 ₂	10100 ₃	132 ₈	90	76 ₁₂	5A ₁₆
hundred	cent	1100100 ₂	10201 ₃	144 ₈	100	84 ₁₂	64 ₁₆

words that represent numbers, and many of those irregularities originate in the traditional use of other number systems. In English, the numbers 11 and 12 do not follow the pattern of the other numbers between 10 and 20. In French, the numbers 11 to 16 follow a different pattern than the numbers 17 to 19, and the numbers 30 to 79 follow a different pattern than the numbers 80 to 99.

We generalize the expanded decimal (base 10) form to other bases in the following way. Let $b \in \mathbb{N}$ with $b > 1$. We can write any number $a \in \mathbb{N}$ with $a < b^n$ in the form

$$a = r_{n-1}b^{n-1} + r_{n-2}b^{n-2} + \cdots + r_1b + r_0,$$

where $0 \leq r_i < b$ for each $i \in \{0, \dots, n-1\}$. To write the number a in base b , we extract the coefficients r_0 to r_{n-1} from the expanded notation. To distinguish numbers in different bases, we add a subscript b to the number in base b if $b \neq 10$. So, the number a from above would be written as

$$a = (r_{n-1} \dots r_2 r_1 r_0)_b$$

in base b . In Figures 11.4.2 and 11.4.3 we give examples of numbers in base 7 and base 16 with their digits, expansions, and the numbers in base 10.

We compute the decimal representation of a base b number by evaluating its base b expansion.

Figure 11.4.2: Numbers in base 7, their base 7 digits, their base 7 expansion, and in base 10. The 7 digits used in base 7 numbers are 0, 1, 2, 3, 4, 5, and 6.

n in base 7	base 7 digits of n				base 7 expansion of n	n in base 10
	7^3	7^2	7^1	7^0		
1_7				1	$1 \cdot 1$	1
10_7			1	0	$1 \cdot 7 + 0 \cdot 1$	7
100_7		1	0	0	$1 \cdot 7^2 + 0 \cdot 7 + 0 \cdot 1$	49
200_7		2	0	0	$2 \cdot 7^2 + 0 \cdot 7 + 0 \cdot 1$	98
6200_7	6	2	0	0	$6 \cdot 7^3 + 2 \cdot 7^2 + 0 \cdot 7 + 0 \cdot 1$	341

Figure 11.4.3: Hexadecimal (base 16) numbers, their base 16 digits, their base 16 expansion, and in base 10. The 16 digits used in hexadecimal numbers are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F. We have $A_{16} = 10$, $B_{16} = 11$, $C_{16} = 12$, $E_{16} = 13$, $F_{16} = 14$, and $F_{16} = 15$.

n in base 16	base 16 digits of n				base 16 expansion of n	n in base 10
	16^3	16^2	16^1	16^0		
1_{16}				1	$1 \cdot 1$	1
C_{16}				C	$12 \cdot 1$	12
10_{16}			1	0	$1 \cdot 16 + 0 \cdot 1$	16
$A0_{16}$			A	0	$10 \cdot 16 + 0 \cdot 1$	160
FF_{16}			F	F	$15 \cdot 16 + 15 \cdot 1$	255
100_{16}		1	0	0	$1 \cdot 16^2 + 0 \cdot 16 + 0 \cdot 1$	256
200_{16}		2	0	0	$2 \cdot 16^2 + 0 \cdot 16 + 0 \cdot 1$	512
$6B00_{16}$	6	B	0	0	$6 \cdot 16^3 + 11 \cdot 16^2 + 0 \cdot 16 + 0 \cdot 1$	27392

Example 11.4.1. Given numbers in various bases b , we convert these numbers to their decimal representations by writing out their base b expansions and then evaluating them.

- (i) $1101_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2 + 1 \cdot 1 = 13$
- (ii) $1101_3 = 1 \cdot 3^3 + 1 \cdot 3^2 + 0 \cdot 3 + 1 \cdot 1 = 37$
- (iii) $201_3 = 2 \cdot 3^2 + 0 \cdot 3 + 1 \cdot 1 = 19$
- (iv) $201_5 = 2 \cdot 5^2 + 0 \cdot 5 + 1 \cdot 1 = 51$
- (v) $201_{16} = 2 \cdot 16^2 + 0 \cdot 16 + 1 \cdot 1 = 513$
- (vi) $A3B_{16} = 10 \cdot 16^2 + 3 \cdot 16 + 11 \cdot 1 = 2619$

Problem 11.4.2. Give the base 18 expansion of $99GD872_{18}$ and convert $99GD872_{18}$ to a decimal number.

Solution. In base 18 we use the characters 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F,G,H for the digits. The values of these are

$$\begin{array}{llllll}
0_{18} = 0 & 1_{18} = 1 & 2_{18} = 2 & 3_{18} = 3 & 4_{18} = 4 & 5_{18} = 5 \\
6_{18} = 6 & 7_{18} = 7 & 8_{18} = 8 & 9_{18} = 9 & A_{18} = 10 & B_{18} = 11 \\
C_{18} = 12 & D_{18} = 13 & E_{18} = 14 & F_{18} = 15 & G_{18} = 16 & H_{18} = 17
\end{array}$$

So as the base 18 expansion of $99GD87218$ we get

$$99GD87218 = 9 \cdot 18^6 + 9 \cdot 18^5 + 16 \cdot 18^4 + 13 \cdot 18^3 + 8 \cdot 18^2 + 7 \cdot 18 + 2 \cdot 1.$$

Evaluating the expression on the right yields the decimal representation of $99GD87218$:

$$99GD87218 = 9 \cdot 18^6 + 9 \cdot 18^5 + 16 \cdot 18^4 + 13 \cdot 18^3 + 8 \cdot 18^2 + 7 \cdot 18 + 2 \cdot 1 = 324874280$$

11.5 Conversion from Base 10 to Base b

Let $a \in \mathbb{N}$ and let

$$a = r_m \cdot b^m + r_{m-1}b^{m-1} + \cdots + r_3 \cdot b^3 + r_2 \cdot b^2 + r_1 \cdot b + r_0$$

be the base b expansion of a . Dividing a by b we obtain the rightmost base b digit of a :

$$\begin{aligned}
a \bmod b &= r_0 \\
a \operatorname{div} b &= r_m \cdot b^{m-1} + r_{m-1}b^{m-2} + \cdots + r_3 \cdot b^2 + r_2 \cdot b + r_1
\end{aligned}$$

Thus

$$a = r_0 + b \cdot (r_m \cdot b^{m-1} + r_{m-1}b^{m-2} + \cdots + r_3 \cdot b^2 + r_2 \cdot b + r_1).$$

Dividing $(r_m \cdot b^{m-1} + r_{m-1}b^{m-2} + \cdots + r_3 \cdot b^2 + r_2 \cdot b + r_1)$ by b we obtain the next base b digit of a :

$$\begin{aligned}
(r_m \cdot b^{m-1} + r_{m-1}b^{m-2} + \cdots + r_3 \cdot b^2 + r_2 \cdot b + r_1) \bmod b &= r_1 \\
(r_m \cdot b^{m-1} + r_{m-1}b^{m-2} + \cdots + r_3 \cdot b^2 + r_2 \cdot b + r_1) \operatorname{div} b &= r_m \cdot b^{m-2} + r_{m-1}b^{m-3} + \cdots + r_3 \cdot b + r_2
\end{aligned}$$

Thus

$$a = r_0 \cdot b^0 + r_1 \cdot b + (r_m \cdot b^{m-2} + r_{m-1}b^{m-3} + \cdots + r_3 \cdot b + r_2) \cdot b^2.$$

Continuing in this way, we successively compute the digits r_1 to r_m of the base b expansion of a using divisions with remainders. We formulate this method as an algorithm:

Algorithm 11.5.1 (*Base Conversion*).

Input: A base $b \in \mathbb{N}$ with $b \neq 1$ and $a \in \mathbb{N}$

Output: The base b digits $r_0, \dots, r_m \in \mathbb{Z}_b$ of a such that $a = r_m \cdot b^m + r_{m-1}b^{m-1} + \cdots + r_3 \cdot b^3 + r_2 \cdot b^2 + r_1 \cdot b + r_0$

- (i) **let** $i := 0$
- (ii) **repeat**
 - (a) **let** $r_i := a \bmod b$
 - (b) **let** $a := a \operatorname{div} b$

- (c) **let** $i := i + 1$
- (iii) **until** $a = 0$
- (iv) **return** $r_0, \dots, r_m$

Example 11.5.2. We convert the base 10 number 23 to base 3 using Algorithm 11.5.1.

Input: $b = 3, a = 23$

$$\begin{array}{lll} i = 0 & r_0 = 23 \bmod 3 = 2 & a = 23 \operatorname{div} 3 = 7 \\ i = 1 & r_1 = 7 \bmod 3 = 1 & a = 7 \operatorname{div} 3 = 2 \\ i = 2 & r_2 = 2 \bmod 3 = 2 & a = 2 \operatorname{div} 3 = 0 \end{array}$$

Output: $r_0 = 2, r_1 = 1, r_2 = 2$

The base 3 expansion of 25 is $23 = 2 \cdot 3^2 + 1 \cdot 3 + 2 \cdot 1$. Thus the base 3 representation of 23 is 212_3 .

Example 11.5.3. We convert the base 10 number 1709 to base 16 using Algorithm 11.5.1.

Input: $b = 16, a = 1709$

$$\begin{array}{lll} i = 0 & r_0 = 1709 \bmod 16 = 13 & a = 1709 \operatorname{div} 16 = 106 \\ i = 1 & r_1 = 106 \bmod 16 = 10 & a = 106 \operatorname{div} 16 = 6 \\ i = 2 & r_2 = 6 \bmod 16 = 6 & a = 6 \operatorname{div} 16 = 0 \end{array}$$

Output: $r_0 = 13, r_1 = 10, r_2 = 6$

The base 16 expansion of 1709 is $1709 = 6 \cdot 16^2 + 10 \cdot 16 + 13 \cdot 1$. Since $10 = A_{16}$ and $13 = D_{16}$, the base 16 representation of 1709 is $6AD_{16}$.

Example 11.5.4. Let $b = 16$ and $a = 2619$.

$$\begin{array}{lll} i = 0 & r_0 = 2619 \bmod 16 = 11 & a = 2619 \operatorname{div} 16 = 163 \\ i = 1 & r_1 = 163 \bmod 16 = 3 & a = 163 \operatorname{div} 16 = 10 \\ i = 2 & r_2 = 10 \bmod 16 = 10 & a = 10 \operatorname{div} 16 = 0 \end{array}$$

Since $10 = A_{16}$ and $11 = B_{16}$, the base 16 representation of 2619 is $A3B_{16}$.

Example 11.5.5. Let $b = 11$ and $a = 2619$.

$$\begin{array}{lll} i = 0 & r_0 = 2619 \bmod 11 = 1 & a = 2619 \operatorname{div} 11 = 238 \\ i = 1 & r_1 = 238 \bmod 11 = 7 & a = 238 \operatorname{div} 11 = 21 \\ i = 2 & r_2 = 21 \bmod 11 = 10 & a = 21 \operatorname{div} 11 = 1 \\ i = 3 & r_3 = 1 \bmod 11 = 1 & a = 1 \operatorname{div} 11 = 0 \end{array}$$

The base 11 expansion of 2619 is $2619 = 1 \cdot 11^3 + 10 \cdot 11^2 + 7 \cdot 11 + 1 \cdot 1$. Since $10 = A_{11}$, the base 11 representation of 2619 is $1A71_{11}$.

Example 11.5.6. Let $b = 3$ and $a = 2619$.

$i = 0$	$r_0 = 2619 \bmod 3 = 0$	$a = 2619 \operatorname{div} 3 = 873$
$i = 1$	$r_1 = 873 \bmod 3 = 0$	$a = 873 \operatorname{div} 3 = 291$
$i = 2$	$r_2 = 291 \bmod 3 = 0$	$a = 291 \operatorname{div} 3 = 97$
$i = 3$	$r_3 = 97 \bmod 3 = 1$	$a = 97 \operatorname{div} 3 = 32$
$i = 4$	$r_4 = 32 \bmod 3 = 2$	$a = 32 \operatorname{div} 3 = 10$
$i = 5$	$r_5 = 10 \bmod 3 = 1$	$a = 10 \operatorname{div} 3 = 3$
$i = 6$	$r_6 = 3 \bmod 3 = 0$	$a = 3 \operatorname{div} 3 = 1$
$i = 7$	$r_7 = 1 \bmod 3 = 1$	$a = 1 \operatorname{div} 3 = 0$

The base 3 expansion of 2619 is $2619 = 1 \cdot 3^7 + 0 \cdot 3^6 + 1 \cdot 3^5 + 2 \cdot 3^4 + 1 \cdot 3^3 + 0 \cdot 3^2 + 0 \cdot 3 + 0 \cdot 1$.
Thus the base 3 representation of 2619 is 10121000_3 .

Chapter 12

Applications of other Bases

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Convert black and white images to a sequence of numbers.
- (2) Convert a sequence of numbers into an image.
- (3) Identify colors given as hexadecimal triplets.
- (4) Compare shades of a gray given as hexadecimal triplets.
- (5) Convert a text into a number.
- (6) Convert a number into a text.

Numbers in bases other than the familiar base 10 have many real world applications. Base 12 used to be popular and is still used in packaging sizes. Computers internally represent everything as base 2 numbers. In this section, we give three real world examples of ways numbers in bases other than base 10 are used – the encoding of images by numbers description of colors as triples of hexadecimal numbers, and the encoding of texts by numbers.

12.1 Images

We describe how an image that consists of *pixels* (the little rectangles that are the points in a raster image) can be encoded into numbers.

Example 12.1.1. In the steps illustrated by an example in the columns of Figure 12.1.1 we do the following:

- (a) We start with the initial image from Figure 6.3.1.
- (b) We represent white pixels by 0s and black pixels by 1s and fill in the cells of the raster accordingly. We call this version of the raster a *bitmap*.
- (c) We consider the digits in each line of the raster as the digits of a binary number.
- (d) Since leading zeros do not change the value of numbers in any base, we can remove the leading zeros of the binary numbers.
- (e) We convert the binary numbers into decimal numbers.

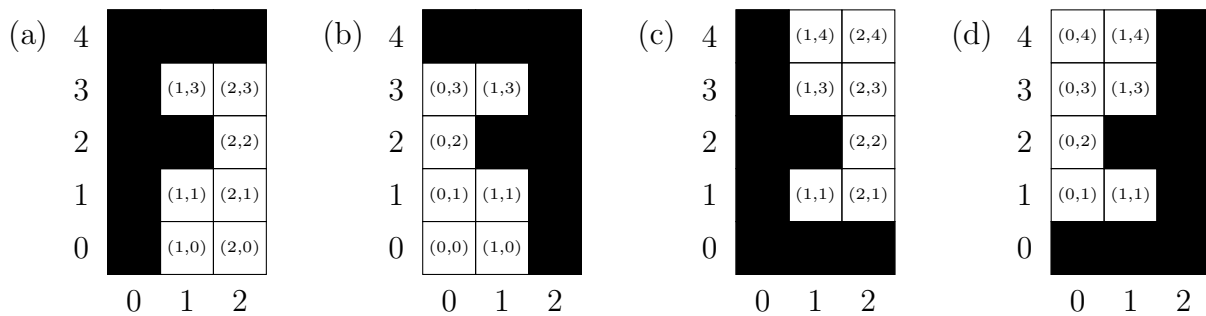
We have obtained an encoding of the image as 0, 8, 56, 84, 108, 56, 108, 0, 0. This is a more compact representation of the image than the representation as a subset of a Cartesian product.

Problem 12.1.2. Represent the image from Figure 12.1.2 (a) by a decimal number for each row. In the bitmap, use 1s to represent black pixels.

Figure 12.1.1: Encoding the image on the left, as a bitmap (black pixels are 1s, white pixels are 0s), in binary, and in decimal numbers. Each number represents one row in the image.

(a) image	(b) bitmap	(c) binary (with leading 0s)	(d) binary	(e) decimal																																																																																																																																		
	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	1	1	0	1	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	<table><tr><td>000000000₂</td></tr><tr><td>000001000₂</td></tr><tr><td>000111000₂</td></tr><tr><td>001010100₂</td></tr><tr><td>001101100₂</td></tr><tr><td>000111000₂</td></tr><tr><td>001101100₂</td></tr><tr><td>000000000₂</td></tr><tr><td>000000000₂</td></tr><tr><td>000000000₂</td></tr></table>	000000000 ₂	000001000 ₂	000111000 ₂	001010100 ₂	001101100 ₂	000111000 ₂	001101100 ₂	000000000 ₂	000000000 ₂	000000000 ₂	<table><tr><td>0₂</td></tr><tr><td>1000₂</td></tr><tr><td>111000₂</td></tr><tr><td>1010100₂</td></tr><tr><td>1101100₂</td></tr><tr><td>111000₂</td></tr><tr><td>1101100₂</td></tr><tr><td>0₂</td></tr><tr><td>0₂</td></tr><tr><td>0₂</td></tr></table>	0 ₂	1000 ₂	111000 ₂	1010100 ₂	1101100 ₂	111000 ₂	1101100 ₂	0 ₂	0 ₂	0 ₂	<table><tr><td>0</td></tr><tr><td>8</td></tr><tr><td>56</td></tr><tr><td>84</td></tr><tr><td>108</td></tr><tr><td>56</td></tr><tr><td>108</td></tr><tr><td>0</td></tr><tr><td>0</td></tr><tr><td>0</td></tr></table>	0	8	56	84	108	56	108	0	0	0
0	0	0	0	0	0	0	0	0	0																																																																																																																													
0	0	0	0	0	1	0	0	0	0																																																																																																																													
0	0	0	1	1	1	0	0	0	0																																																																																																																													
0	0	1	0	1	0	1	0	0	0																																																																																																																													
0	0	1	1	0	1	1	0	0	0																																																																																																																													
0	0	0	1	1	1	0	0	0	0																																																																																																																													
0	0	1	1	0	1	1	0	0	0																																																																																																																													
0	0	0	0	0	0	0	0	0	0																																																																																																																													
0	0	0	0	0	0	0	0	0	0																																																																																																																													
0	0	0	0	0	0	0	0	0	0																																																																																																																													
000000000 ₂																																																																																																																																						
000001000 ₂																																																																																																																																						
000111000 ₂																																																																																																																																						
001010100 ₂																																																																																																																																						
001101100 ₂																																																																																																																																						
000111000 ₂																																																																																																																																						
001101100 ₂																																																																																																																																						
000000000 ₂																																																																																																																																						
000000000 ₂																																																																																																																																						
000000000 ₂																																																																																																																																						
0 ₂																																																																																																																																						
1000 ₂																																																																																																																																						
111000 ₂																																																																																																																																						
1010100 ₂																																																																																																																																						
1101100 ₂																																																																																																																																						
111000 ₂																																																																																																																																						
1101100 ₂																																																																																																																																						
0 ₂																																																																																																																																						
0 ₂																																																																																																																																						
0 ₂																																																																																																																																						
0																																																																																																																																						
8																																																																																																																																						
56																																																																																																																																						
84																																																																																																																																						
108																																																																																																																																						
56																																																																																																																																						
108																																																																																																																																						
0																																																																																																																																						
0																																																																																																																																						
0																																																																																																																																						

Figure 12.1.2: Images for Problems 12.1.2 and 12.1.3



Solution. As the binary representations for the rows of the raster we obtain:

$$111_2, 100_2, 110_2, 100_2, 100_2$$

Now, we must change these base 2 numbers into base 10 numbers by writing out their base 2 expansions and evaluating them

$$111_2 = 1 \cdot 2^2 + 1 \cdot 2 + 1 \cdot 1 = 7$$

$$100_2 = 1 \cdot 2^2 + 0 \cdot 2 + 0 \cdot 1 = 4$$

$$110_2 = 1 \cdot 2^2 + 1 \cdot 2 + 0 \cdot 1 = 6$$

$$100_2 = 1 \cdot 2^2 + 0 \cdot 2 + 0 \cdot 1 = 4$$

$$100_2 = 1 \cdot 2^2 + 0 \cdot 2 + 0 \cdot 1 = 4$$

So, the representation of the image by decimal numbers is 7, 4, 6, 4, 4.

Problem 12.1.3. Which of the images in Figure 12.1.2 can be encoded as 1,1,3,1,7, when 1s represent black pixels in the bitmap?

Solution. We must work in the opposite direction from the previous problem to answer this question. The decimal numbers are given. First, we change those decimal numbers into binary numbers:

$$\begin{aligned}1_{10} &= 1_2 \\1_{10} &= 1_2 \\3_{10} &= 11_2 \\1_{10} &= 1_2 \\7_{10} &= 111_2\end{aligned}$$

To help us more completely visualize the image, we insert leading 0s:

$$\begin{aligned}1_{10} &= 001_2 \\1_{10} &= 001_2 \\3_{10} &= 011_2 \\1_{10} &= 001_2 \\7_{10} &= 111_2\end{aligned}$$

Now, we see that the decimal numbers 1,1,3,1,7 encode Figure 12.1.2 (d).

12.2 Colors

There are different models for describing color. In the *RGB* (*Red Green Blue*) color model, colors are additively mixed from the colors red, green, and blue (thus the name), see Figure 12.2.1. By varying the intensity of the three colors all colors can be mixed this way. On the *World Wide Web* (WWW) the intensity of each of the three colors is represented two digit hexadecimal number which yields $16 \cdot 16 = 256$ different intensities.

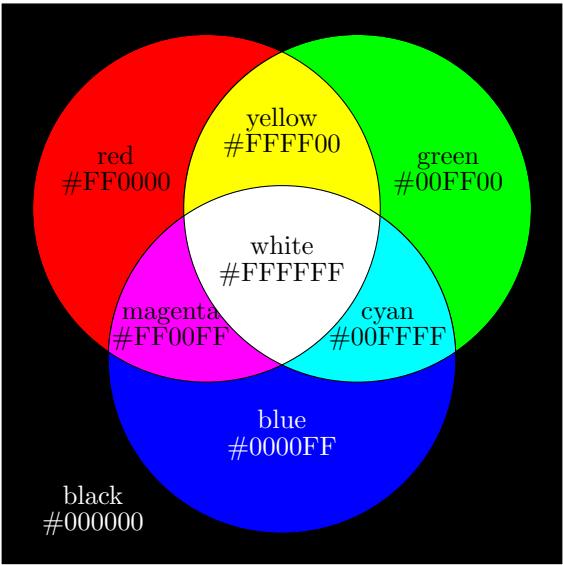
In Figure 12.2.2 the intensities of each of the three colors red, green, and blue are represented by numbers between 0 and 255. Where 0 stands for no contribution of a color and 255 the strongest possible contribution. We use the hexadecimal color representation that is also used for colors on the World Wide Web. Each color is represented by a three two digit hexadecimal numbers, called an *RGB hex triplet*. Each of the two digit hexadecimal numbers represents the intensity of one of the colors red, green, and blue (in this order). To indicate that the six digit hexadecimal number should be interpreted as a RGB hex triplet it is prefixed by a hash mark. So we get

$$\# \underbrace{r_1 r_2}_{\text{red}} \underbrace{g_1 g_2}_{\text{green}} \underbrace{b_1 b_2}_{\text{blue}},$$

where the two digit hexadecimal number $(r_1 r_2)_{16}$ represents the intensity of red, the two digit hexadecimal number $(g_1 g_2)_{16}$ represents the intensity of green, and the two digit hexadecimal number $(b_1 b_2)_{16}$ represents the intensity of blue.

Example 12.2.1. We give examples of colors represented by hex triplets, compare Figure 12.2.1.

Figure 12.2.1: Primary and secondary RGB colors. Mixing blue and green yields cyan. Mixing blue and red yields magenta. Mixing green and red yields yellow. Mixing blue, green, and red yields white.



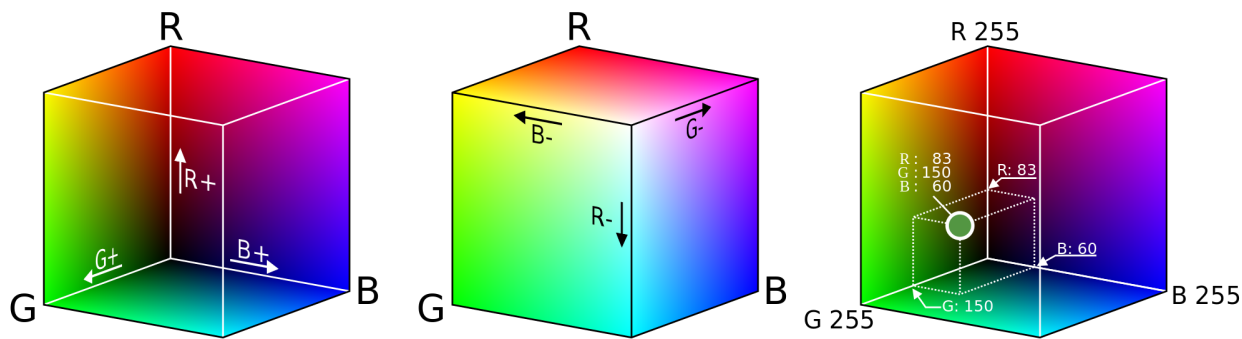
hex triplet		color
#000000		black
#FF0000		red
#00FF00		green
#0000FF		blue
#FFFF00		yellow
#FF00FF		magenta
#00FFFF		cyan
#FFFFFF		white
#53964C		a muddy green (Figure 12.2.2)

12.2.1 Shades of Grey

We obtain grays by setting red, green, and blue to the same intensity. We already have seen that #000000 yields black and that #FFFFFF yields white. The hex triplets of the form # $a_1a_2a_1a_2a_1a_2$ where $(a_1a_2)_{16}$ is a two digit hexadecimal number yield 256 different levels of gray, one for each two digit hexadecimal number between $00_{16} = 0$ and $FF_{16} = 255$.

Example 12.2.2. We give six different shades of gray.

Figure 12.2.2: RGB (Red Green Blue) color cube (by Maklaan licensed under the Creative Commons Attribution-Share Alike 3.0 Unported). The 256 shades of gray #000000 (white) to #FFFFFF (black) are on the line from the bottom back corner to the top front corner. The hexadecimal representation of the color in the circle in the third cube is #53964C, since $53_{16} = 83$, $96_{16} = 150$, and $4C_{16} = 60$.



hex triplet		color
#000000		black
#404040		a gray
#4D4D4D		a slightly lighter gray
#808080		a lighter gray
#E1E1E1		an even lighter gray
#FFFFFF		white

The differences in brightness are evident from $0_{16} < 40_{16} < 4D_{16} < 80_{16} < E1_{16} < FF_{16}$.

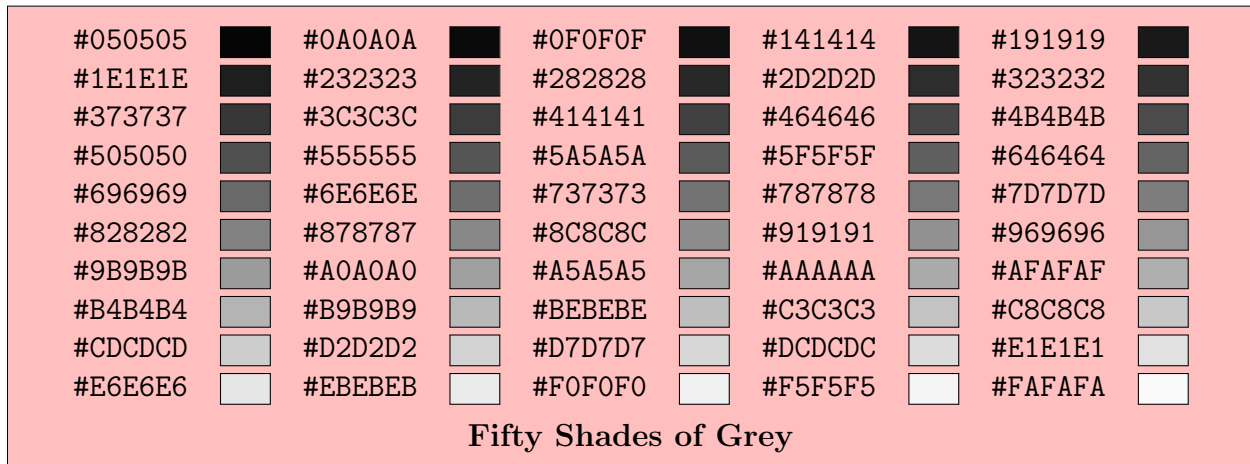
Problem 12.2.3. *Is the shade of gray given by #A2A2A2 darker than or lighter than the gray given by #474747 ?*

Solution. To compare the two grays we only need to compare one of the three two digit hexadecimal colors. So we compare $A2_{16}$ and 47_{16} . In this example the sixteens (A and 4) differ so determining which of these is larger yields the solution. We have $A_{16} > 4_{16}$. So $A2_{16} > 47_{16}$ which means that the gray given by #A2A2A2 is lighter than the gray given by #474747.

Problem 12.2.4. *Is the shade of gray given by #ABABAB darker than or lighter than the gray given by #AEAEAE ?*

Solution. To compare the two Grey's we only need to compare one of the three two digit hexadecimal colors. So we compare AB_{16} and AE_{16} . In this example the sixteens (A for both colors) are the same, we need to consider the ones to determine which is the lighter shade of gray. We have $B_{16} < E_{16}$. So $AB_{16} < AE_{16}$ which means that the gray given by #ABABAB is darker than the gray given by #AEAEAE.

Figure 12.2.3: Two examples of hex triplet color humor.



Not everything is #000000 and #FFFFFF.

12.2.2 Darker and Lighter Colors

We now describe how darker and lighter versions of the principal colors red green blue are obtained. If we set two of the three colors red, green, and blue to zero and decrease the third color we obtain darker versions of the that third color. Fixing one of the three colors red, green, and blue and setting the two remaining intensities to the same level yield brighter versions of the first color.

Example 12.2.5. We demonstrate how darker blues are formulated as RGB hex triplets.

hex triplet		color
#0000FF		blue
#0000AB		a darker blue
#00009C		an even darker blue
#000006		a very dark blue, almost black
#000000		black

The differences in brightness are evident from $0_{16} < 6_{16} < 9C_{16} < AB_{16} < FF_{16}$.

Example 12.2.6. We demonstrate how lighter blues are formulated as RGB hex triplets.

hex triplet		color
#0000FF		blue
#3434FF		a lighter blue
#8080FF		an even lighter blue
#E1E1FF		a very light blue
#FFFFFF		white

Figure 12.3.1: Tables that specify the encoding function $C : \mathbb{A} \rightarrow \mathbb{Z}_{27}$ and its inverse the decoding function $C^{-1} : \mathbb{Z}_{27} \rightarrow \mathbb{A}$

x	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
$C(x)$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

y	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
$C^{-1}(y)$	-	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z

The differences in brightness are evident from $0_{16} < 34_{16} < 80_{16} < E1_{16} < FF_{16}$.

Problem 12.2.7. Which color best describes #A1FFA1 ?

(i) a gray, (ii) light green, (iii) dark red, (iv) red, (v) cyan, (vi) blue

Solution. In #A1FFA1 the strongest of the three colors is green. The other two are at the same medium level. This makes for a lighter color than green. Thus (ii) light green is the correct solution.

12.3 Text

We describe how a string of several characters (like a word) can be encoded into a single decimal number. This is often the first step in many cryptographic protocols, after which the characters within the number are then encrypted with an encryption function.

Strategy 12.3.1. To compute the decimal representation of a word proceed as follows.

- (1) Encode the letters of the given word into numbers in $\mathbb{Z}_{27}$ using the function $C : \mathbb{A} \rightarrow \mathbb{Z}_{27}$ from Figure 12.3.1.
- (2) Consider these numbers as the digits of a base 27 number. Convert this base 27 number into a single base 10 number by writing out the base 27 expansion and evaluating it to complete the decimal representation of the given word.

Example 12.3.2. We demonstrate how to compute a decimal representation of the word **wombat** by following the steps given in the strategy above.

- (1) The function C defined in Figure 12.3.1 encodes the letters in **wombat** as the numbers 23, 15, 13, 2, 1, 20. We now consider these as the values of the digits of a base 27 number. We obtain

$$23 \cdot 27^5 + 15 \cdot 27^4 + 13 \cdot 27^3 + 2 \cdot 27^2 + 1 \cdot 27 + 20 \cdot 1 = 338253860$$

So, the decimal representation of the word **wombat** is the decimal number 338253860.

Notice that actually writing out the base 27 number using the capital letters that represent values bigger than 9 in the second step is not necessary. Once we have the numbers from the

first step, we could simply jump down to the base 27 expansion in the third step. We will demonstrate the abbreviated strategy in the solution of the problem that follows. The second step was included in the strategy above to fully and correctly communicate the mathematics that is being used.

Problem 12.3.3. *Compute a decimal representation of the word **dog**.*

Solution. Encoding the letters in **dog** by the function C from Figure 12.3.1 we obtain

$$C(\mathbf{d}) = 4, \quad C(\mathbf{o}) = 15, \quad C(\mathbf{g}) = 7.$$

Considering as the values of the digits of a base 27 representation, we write out the base 27 expansion and evaluate it:

$$4 \cdot 27^2 + 15 \cdot 27 + 7 \cdot 1 = 3328$$

So, the decimal representation of the word **dog** is the decimal number 3328.

We can also work backwards to find the word that is encoded in a given decimal number.

Strategy 12.3.4. To convert a decimal representation of a word to the word proceed as follows.

- (1) Find the base 27 expansion of the word.
- (2) Decode each digit of the base 27 expansion using the decoding function $C^{-1} : \mathbb{Z}_{27} \rightarrow \mathbb{A}$ given by from Figure 12.3.1 to obtain the characters of the word.

Problem 12.3.5. *Find the word encoded as the number 2234.*

Solution. We start by converting the decimal number 2234 to a base 27 number:

$$\begin{array}{ll} 2234 \operatorname{div} 27 = 82 & 2234 \operatorname{mod} 27 = 20 \\ 82 \operatorname{div} 27 = 3 & 82 \operatorname{mod} 27 = 1 \\ 3 \operatorname{div} 27 = 0 & 3 \operatorname{mod} 27 = 3 \end{array}$$

Thus the base 27 expansion of 2234 is $2234 = 3 \cdot 27^2 + 1 \cdot 27 + 20 \cdot 1$. We decode the digits of the base 27 expansion number into letters using the function C^{-1} from Figure 12.3.1:

$$C^{-1}(3) = \mathbf{c}, \quad C^{-1}(1) = \mathbf{a}, \quad C^{-1}(20) = \mathbf{t}$$

So, the word encoded as the number 2234 is **cat**.

Figure 12.3.2: *Code Talkers* by R. Munroe (<https://xkcd.com/257>).



As far as I can tell, Navajo doesn't have a common word for 'zero'. do-neh-lini means 'neutral'.

Part IV

Groups and Cryptography

In this fourth part of the course, we consider commutative groups and one of their most important applications in every day life. At the end of this chapter we present the Diffie-Hellman key exchange that is, for example, used when your web browser establishes a secure (https) connection with a web server.

We bring together together many of the topics from chapters 1, 2, and 3. The sets $\mathbb{Z}_n$ and $\mathbb{Z}_n^\otimes$ show up again, in particular the sets $\mathbb{Z}_p^\otimes$ where p is a prime number will be of interest. We introduce new operations on these sets and revisit exponentiation. Finally, we apply these in real world encryption algorithms.

One of the most familiar examples of a commutative group is the set of integers with the addition operation. There is a wide variety of groups that find applications in a multitude of fields. In addition to their application in cryptography, groups are used to describe symmetries of objects in physics and chemistry.

In Chapter 13, we introduce binary operations and properties of binary operations. We give the definition of a commutative group and some examples of commutative groups in Chapter 14. As mentioned before, the mod operation will become important to us. We give some more applications of mod and then show how the sets $\mathbb{Z}_n$ and $\mathbb{Z}_n^\otimes$ together with operations based on mod and addition or multiplication, respectively, give us infinitely many groups. We present two families of groups whose operations are modular addition and modular multiplication, respectively. Within these groups, we examine groups that are generated by one element in Chapter 15 and show how they are used in public key cryptosystems in Chapter 16.

Chapter 13

Binary Operations

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Show that there is an identity with respect to a binary operation.
- (2) Show that an element has an inverse with respect to a binary operation.
- (3) Recognize whether a binary operation is associative.
- (4) Show that a binary operation is commutative.

A binary operation is a function on a set that combines two elements of the set to form a third element of the set.

Examples of binary operations on the integers are addition, subtraction, multiplication, div , and mod . In this section we introduce four properties of some of the binary operations that we have already encountered. These properties are:

- (i) Existence of an identity element,
- (ii) Existence of inverses,
- (iii) Associativity, and
- (iv) Commutativity

13.1 Definition

A binary operation can be considered as a function whose input is two elements of the same set S and whose output also is an element of S . Two elements a and b of S can be written as a pair (a, b) of elements in S . As (a, b) is an element of the Cartesian product $S \times S$ we specify a binary operation as a function from $S \times S$ to S . We use symbols to represent functions that are binary operations instead of using variables or function names, just as we do with addition and multiplication of integers. Addition uses the symbol $+$ and multiplication uses the symbol $\cdot$. We will use symbols such as $\star$ and $\bullet$ to represent arbitrary (non-specific) binary operations, and we will also define new binary operations using the symbols $\oplus$ and $\otimes$.

Definition 13.1.1. A *binary operation* $\bullet$ on a set S is a function $\bullet : S \times S \rightarrow S$. For the image of $(a, b) \in S \times S$ under the function $\bullet$ we write $a \bullet b$ (read ‘ a dot b ’).

Example 13.1.2. We give examples for binary operations that we have encountered before.

- (i) The addition of integers $+: \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$ is a binary operation on $\mathbb{Z}$. We denote the image of $(a, b) \in \mathbb{Z} \times \mathbb{Z}$ by $a + b$.
- (ii) The multiplication of natural numbers $\cdot: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ is a binary operation on $\mathbb{N}$. We denote the image of $(a, b) \in \mathbb{N} \times \mathbb{N}$ by $a \cdot b$.

- (iii) The subtraction of integers $- : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$ is a binary operation on $\mathbb{Z}$. We denote the image of $(a, b) \in \mathbb{Z} \times \mathbb{Z}$ by $a - b$.

As is the case for other functions, there are several ways of specifying a binary operation. If the set is small, we sometimes specify the binary operation by a table.

Example 13.1.3. Let $T := \{x, y, z\}$. The binary operation $\star : T \times T \rightarrow T$ is given by the operation table:

		b		
		x	y	z
a	$\star$	x	y	z
	x	z	x	y
	y	x	y	z
	z	y	z	x

From the table, we can obtain $a \star b$ (read “ a star b ”) for each $a, b \in T$:

$$\begin{array}{lll} x \star x = z & x \star y = x & x \star z = y \\ y \star x = x & y \star y = y & y \star z = z \\ z \star x = y & z \star y = z & z \star z = x \end{array}$$

Sometimes it can be useful to generate the operation table from a binary operation given by an algebraic rule.

Example 13.1.4. The operation table for the binary operation $\oplus : \mathbb{Z}_5 \times \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ given by $a \oplus b = (a + b) \bmod 5$ is:

$\oplus$	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

We read $a \oplus b$ as “ a mod plus b .”

13.2 Associativity

If a binary operation is Associative, the order in which we evaluate expressions that only involve that one binary operation does not matter.

Definition 13.2.1. Let S be a set and $\bullet : S \times S \rightarrow S$ be a binary operation on S . Then, $\bullet$ is *associative* if $a \bullet (b \bullet c) = (a \bullet b) \bullet c$ for all $a \in S$, $b \in S$, and $c \in S$.

Example 13.2.2. We continue to consider the binary operations from Example 13.1.2:

- (i) The addition of integers $+: \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$ is associative.
- (ii) The multiplication of natural numbers $\cdot: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ is associative.
- (iii) For the difference of integers $-: \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$ we have

$$3 - (2 - 1) = 3 - 1 = 2 \text{ and } (3 - 2) - 1 = 1 - 1 = 0.$$

As $2 \neq 0$ the binary operation $-$ (minus) is not associative.

It is often labor-intensive to verify that a binary operation is associative. We demonstrate the verification process for a binary operation on a (small) finite set in the following example.

Example 13.2.3. Let $T = \{x, y, z\}$, and let the binary operation $\star: T \times T \rightarrow T$ be given by the table in Example 13.1.3. To prove that $\star$ is associative, we exhaust all possibilities. We verify that for all $a \in T$, $b \in T$, and $c \in T$,

$$a \star (b \star c) \text{ is equal to } (a \star b) \star c$$

by separately computing $a \star (b \star c)$ in the left column and $(a \star b) \star c$ in the right column and noticing that the two computations in each row match.

In the case where one of the general elements is the identity element, there is a shortcut. We can handle several cases at the same time by setting one of the three general elements equal to the identity element and using variables for the other two general elements. Notice that for all $a \in T$ we have $y \star a = a$ and $a \star y = a$. Then, for all $a \in T$ and all $b \in T$ we have:

$$\begin{array}{ll} y \star (a \star b) = a \star b & (y \star a) \star b = a \star b \\ a \star (y \star b) = a \star b & (a \star y) \star b = a \star b \\ a \star (b \star y) = a \star b & (a \star b) \star y = a \star b \end{array}$$

We explicitly cover the remaining cases:

$$\begin{array}{ll} x \star (x \star x) = x \star z = y & (x \star x) \star x = z \star x = y \\ x \star (x \star z) = x \star y = x & (x \star x) \star z = z \star z = x \\ x \star (z \star x) = x \star y = x & (x \star z) \star x = y \star x = x \\ x \star (z \star z) = x \star x = z & (x \star z) \star z = y \star z = z \\ \\ z \star (x \star x) = z \star z = x & (z \star x) \star x = y \star x = x \\ z \star (x \star z) = z \star y = z & (z \star x) \star z = y \star z = z \\ z \star (z \star x) = z \star y = z & (z \star z) \star x = x \star x = z \\ z \star (z \star z) = z \star x = y & (z \star z) \star z = x \star z = y \end{array}$$

We have shown that $a \star (b \star c) = (a \star b) \star c$ for all $a \in T$, $b \in T$, and $c \in T$. Thus, the binary operation $\star: T \times T \rightarrow T$ is associative.

Example 13.2.4. Consider the binary operation $\oplus: \mathbb{Z}_5 \times \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ defined by $a \oplus b = (a + b) \bmod 5$. We use the associativity of $+$ to show that $\oplus$ is associative. For all $a \in \mathbb{Z}_5$ and $b \in \mathbb{Z}_5$ and $c \in \mathbb{Z}_5$ we have by the definition of $\oplus$ that

$$a \oplus (b \oplus c) = a \oplus ((b + c) \bmod 5) = (a + ((b + c) \bmod 5)) \bmod 5.$$

With Theorem 3.4.5 we get

$$(a + ((b + c) \bmod 5)) \bmod 5 = (a + (b + c)) \bmod 5.$$

As the addition $+$ of integers is associative we have

$$(a + (b + c)) \bmod 5 = ((a + b) + c) \bmod 5.$$

Again applying Theorem 3.4.5 and the definition of $\oplus$ we obtain

$$((a + b) + c) \bmod 5 = (((a + b) \bmod 5) + c) \bmod 5 = ((a \oplus b) + c) \bmod 5 = (a \oplus b) \oplus c.$$

We have shown that $a \oplus (b \oplus c) = (a \oplus b) \oplus c$ for all $a \in \mathbb{Z}_5$ and $b \in \mathbb{Z}_5$ and $c \in \mathbb{Z}_5$, so $\oplus$ is associative.

Problem 13.2.5. Consider $\mathbb{Z}_4 = \{0, 1, 2, 3\}$ with the binary operation $\star : \mathbb{Z}_4 \rightarrow \mathbb{Z}_4$ given by $a \star b = (a^b) \bmod 4$. Decide whether $\star$ is associative.

Solution. As it often is easier to find a counterexample than to find a proof, we try finding a counterexample first. We have

$$2 \star (3 \star 2) = 2 \star (3^2 \bmod 4) = 2 \star (9 \bmod 4) = 2 \star 1 = (2^1) \bmod 4 = 2$$

$$(2 \star 3) \star 2 = (2^3 \bmod 4) \star 2 = (8 \bmod 4) \star 2 = 0 \star 2 = (0^2) \bmod 4 = 0 \bmod 4 = 0$$

As $2 \star (3 \star 2) \neq (2 \star 3) \star 2$ the binary operation $\star$ is not associative.

13.3 Identity

An identity element with respect to a binary operation is an element such that when a binary operation is performed on it and any other given element, the result is the given element.

Definition 13.3.1. Let S be a set and $\bullet : S \times S \rightarrow S$ be a binary operation on S . An element $e \in S$ is an *identity element* of the set S with respect to the operation $\bullet$ if $s \bullet e = s$ and $e \bullet s = s$ for all $s \in S$.

Example 13.3.2. We revisit the binary operations from Example 13.1.2:

- (i) Consider the binary operation $+$: $\mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$. The sum of 0 and any given integer is the given integer. In other words, for all integers s , we have that $s + 0 = 0 + s = s$. So, we call the number 0 the additive identity element for the set of integers.
- (ii) Consider the binary operation $\cdot$: $\mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$. The product of 1 and any given natural number is the given natural number. In other words, for all natural numbers s , we have that $s \cdot 1 = 1 \cdot s = s$. So, we call the number 1 the multiplicative identity element for the set of natural numbers.
- (iii) Consider the binary operation $-$: $\mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$. The only integer e such that $a - e = a$ for all integers a is $e := 0$. For 0 to be an identity with respect to the binary operation $-$ we also need $0 - a = a$ for all $a \in \mathbb{Z}$. We have $0 - 1 = (-1) \neq 1$. So 0 is not an identity with respect to the binary operation $-$. As 0 was the only candidate for an identity, there is no identity with respect to the binary operation $-$.

Theorem 13.3.3. Let S be a set and let $\bullet : S \times S \rightarrow S$ be a binary operation on S . Then, there is at most one element $e \in S$ such that $s \bullet e = s$ and $e \bullet s = s$ for all $s \in S$, implying that if there is an identity element of the set S with respect to the operation $\bullet$, then it is unique.

Proof. Suppose that there are two identity elements e and f of the set S with respect to the operation $\bullet$. Since e is an identity element, $s \bullet e = s$ for all $s \in S$. So, in particular, $f \bullet e = f$. However, since f is an identity element, $f \bullet s = s$ for all $s \in S$. Because $e \in S$, then this implies that $f \bullet e = e$. So, $f \bullet e$ is equal to both f and to e , implying that $f = e$. Therefore f and e must be the same element. Thus, there is at most one identity element of S with respect to $\bullet$. $\square$

Since there can be at most one identity element of a set with respect to a binary operation, we call it *the* identity element, if it exists.

Example 13.3.4. Let $T = \{x, y, z\}$, and let the binary operation $\star : T \times T \rightarrow T$ be given by the table in Example 13.1.3. Notice that $x \star y = y \star x = x$, $y \star y = y$, and $y \star z = z \star y = z$. Since $t \star y = y \star t = t$ for all $t \in T$, we have that $y \in T$ is the identity element of the set T with respect to the operation $\star$.

When we have an operation on a set given by an operation table, we can determine the identity element (if there is one) by locating the element corresponding to a special row and special column within the table. That special row within the table would need to match the header row at the top of the table and that special column within the table would need to match the header column on the left side of the table.

Example 13.3.5. With the above comment in mind, we revisit Example 13.3.4. Notice that the row corresponding to y matches the header row at the top of the table and the column corresponding to y matches the header column on the left side of the table. So, the element y is the identity element.

$\star$	x	y	z
x	z	x	y
y	x	y	z
z	y	z	x

$\star$	x	y	z
x	z	x	y
y	x	y	z
z	y	z	x

Example 13.3.6. The identity element with respect to the operation $\oplus : \mathbb{Z}_5 \times \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$, $a \oplus b = (a+b) \bmod 5$ is 0. To see this we can either use that $a+0 = 0+a = a$ for all integers a or use the method from the previous example and the operation table from Example 13.1.4.

Problem 13.3.7. Consider the binary operation $\oplus : \mathbb{Z}_3 \times \mathbb{Z}_3 \rightarrow \mathbb{Z}_3$ given by $a \oplus b = (a+b) \bmod 3$. Find the identity with respect to $\oplus$ in $\mathbb{Z}_3$.

Solution. Recall that $\mathbb{Z}_3 = \{0, 1, 2\}$.

The binary operation $\oplus$ is based on the addition of integers and the identity with respect to the addition of integers is 0. So we check whether 0 is also the identity with respect to $\oplus$.

For all $a \in \mathbb{Z}_3$ we have

$$a \oplus 0 = (a + 0) \bmod 3 = a \bmod 3 = a$$

where the last equality holds because $a \in \mathbb{Z}_3$. Also for all $a \in \mathbb{Z}_3$ we have

$$0 \oplus a = (0 + a) \bmod 3 = a \bmod 3 = a.$$

Thus 0 is the identity with respect to $\oplus$ in $\mathbb{Z}_3$.

13.4 Inverses

When a binary operation is performed on two elements in a set and the result is the identity element of the set, with respect to the binary operation, the elements are said to be inverses of each other.

Definition 13.4.1. Let S be a set and $\bullet : S \times S \rightarrow S$ be a binary operation on S . Suppose that e is the identity element of S with respect to $\bullet$, and let $s \in S$. An element $t \in S$ is an *inverse* of s with respect to the operation $\bullet$ if $s \bullet t = e$ and $t \bullet s = e$. If $s \in S$ has exactly one inverse, we denote the inverse of s by $s^{-1\bullet}$.

It follows directly from the definition that inverses with respect to a binary operation $\bullet : S \times S \rightarrow S$ can only exist if the set S contains an identity element with respect to $\bullet$.

Theorem 13.4.2. Let S be a set and $\bullet : S \times S \rightarrow S$ be a binary operation on S . If $\bullet$ is associative and $t \in S$ is the inverse of $s \in S$ with respect to $\bullet$, then t is the only inverse of s with respect to $\bullet$.

With this theorem, we have that if $\bullet$ is associative, then inverses are unique.

Definition 13.4.3. Let S be a set and $\bullet : S \times S \rightarrow S$ be a binary operation on S . If $s \in S$ has exactly one inverse with respect to $\bullet$, we denote the inverse of s by $s^{-1\bullet}$.

The notation for inverses uses notation similar to what we used for function inverses. The symbol used for the binary operation is shown with the $^{-1}$ to remind you with respect to which binary operation it is the inverse.

Theorem 13.4.4. Let e be the identity with respect to an associative binary operation $\bullet : S \times S \rightarrow S$ on a set S . Then $e^{-1\bullet} = e$.

Proof. Since e is the identity we have $e \bullet e = e$. So e satisfies all properties of the inverse of e . □

Furthermore, from the condition $s \bullet t = e$ and $t \bullet s = e$ in the definition, we know that if t is the inverse of s , then s is the inverse t .

Theorem 13.4.5. Let S be a set and $\bullet : S \times S \rightarrow S$ be a binary operation on S . If s has an inverse $s^{-1\bullet}$ then

$$(s^{-1\bullet})^{-1\bullet} = s.$$

Example 13.4.6. We continue to consider the binary operations from Example 13.1.2:

- (i) We consider the addition of integers $+: \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$. Recall that the identity element of $\mathbb{Z}$ with respect to addition is 0. Let $s \in \mathbb{Z}$, and note that its negative $-s$ is also in $\mathbb{Z}$. Since $s + (-s) = 0$ and $(-s) + s = 0$, we may conclude that $-s$ is an inverse of s in $\mathbb{Z}$ with respect to $+$. In fact, it is the only such inverse, and we call $-s$ the additive inverse of s .
- (ii) We consider the multiplication of natural numbers $\cdot : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$. Recall that the identity element of $\mathbb{N}$ with respect to multiplication is 1. For $2 \in \mathbb{N}$, we are looking for an element t such that $2 \cdot t = 1$ and $t \cdot 2 = 1$. The only choice would be $t = \frac{1}{2}$; however, $\frac{1}{2}$ is not a natural number. So, 2 does not have a multiplicative inverse in the set of natural numbers. In fact, for each natural number $n > 1$, we have that $\frac{1}{n} \notin \mathbb{N}$, implying that each natural number $n > 1$ does not have a multiplicative inverse in $\mathbb{N}$.
- (iii) As there is no identity with respect to subtraction of integers, there cannot be any inverses.

Problem 13.4.7. Find the inverse of 3 with respect to the addition of integers $+: \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$.

Solution. We have $3 + (-3) = 0$ and $(-3) + 3 = 0$, so (-3) is the inverse of 3 with respect to addition of integers.

Example 13.4.8. Let $T = \{x, y, z\}$, and let the binary operation $\star : T \times T \rightarrow T$ be given by the table in Example 13.1.3. Recall from Example 13.3.4 that y is the identity element of T with respect to $\star$. As is always the case, the inverse of the identity element is itself, so the unique inverse of y is $y^{-1\star} = y$. Also, since $x \star z = y$ and $z \star x = y$, x and z are inverses of each other. Since there are no other elements in T that satisfy the requirements to be an inverse of either x or of z , we may communicate the uniqueness by writing $x^{-1\star} = z$ and $z^{-1\star} = x$. Thus every element in T has a unique inverse with respect to $\star$.

When we have an operation on a set given by an operation table, we can determine which elements are inverses of each other by first determining the identity element (if there is one). Then, we locate the identity element within the table and trace back to the header column on the left side of the table and the header row on the top of the table to find elements that are inverses of each other.

Example 13.4.9. With the above comment in mind, we revisit Example 13.4.8. Recall that the identity element is y . First, we trace back to the header column on the left side of the table and the header row on the top of the table from the following shaded y within the table. We find that the corresponding element in the header column is y and in the header row is y .

$\star$	x	y	z
x	z	x	y
y	x	y	z
z	y	z	x

So, we see that $y \star y = y$ and conclude that $y^{-1\star} = y$.

Now, we trace back to the header column on the left side of the table and the header row on the top of the table from each of the following two shaded y 's within the table. We find that for the first shaded y , the corresponding element in the header column is z and in the header row is x . Furthermore, we find that for the second shaded y , the corresponding element in the header column is x and in the header row is z .

$\star$	x	y	z
x	z	x	y
y	x	y	z
z	y	z	x

$\star$	x	y	z
x	z	x	y
y	x	y	z
z	y	z	x

From the first highlighted table, we see that $z \star x = y$, and from the second highlighted table, we see that $x \star z = y$. Since $z \star x = y$ and $x \star z = y$, we simultaneously conclude that $x^{-1\star} = z$ and that $z^{-1\star} = x$.

Example 13.4.10. Consider the binary operation $\oplus : \mathbb{Z}_5 \times \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ given by $a \oplus b = (a + b) \bmod 5$. The identity element with respect to $\oplus$ is 0 (compare Example 13.3.6). We explicitly give the inverse of each element in $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$.

- (i) As $0 \oplus 0 = (0 + 0) \bmod 5 = 0 \bmod 5 = 0$ the inverse of 0 with respect to $\oplus$ is 0. This illustrates our earlier observation that the inverse of the identity element is the identity element.
- (ii) As $1 \oplus 4 = (1 + 4) \bmod 5 = 5 \bmod 5 = 0$ and $4 \oplus 1 = (4 + 1) \bmod 5 = 5 \bmod 5 = 0$ the inverse of 1 with respect to $\oplus$ is 4. This also shows that the inverse of 4 with respect to $\oplus$ is 1.
- (iii) As $2 \oplus 3 = (2 + 3) \bmod 5 = 5 \bmod 5 = 0$ and $3 \oplus 2 = (3 + 2) \bmod 5 = 5 \bmod 5 = 0$ the inverse of 2 with respect to $\oplus$ is 3. This also shows that the inverse of 3 with respect to $\oplus$ is 2.

13.5 Commutativity

Definition 13.5.1. Let S be a set and $\bullet : S \times S \rightarrow S$ be a binary operation on S . Then, $\bullet$ is *commutative* if $a \bullet b = b \bullet a$ for all $a \in S$ and $b \in S$.

Example 13.5.2. We continue to consider the binary operations from Example 13.1.2:

- (i) The addition of integers $+ : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$ is commutative.
- (ii) The multiplication of natural numbers $\cdot : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ is commutative.

Example 13.5.3. Let $T = \{x, y, z\}$, and let the binary operation $\star : T \times T \rightarrow T$ be given by the table in Example 13.1.3. To prove that $\star$ is commutative, we exhaust all possibilities. We verify that for all $a \in T$ and $b \in T$,

$$a \star b \text{ is equal to } b \star a$$

by separately computing $a \star b$ in the left column and $b \star a$ in the right column and noticing that the two computations in each row match.

In the case where one of the general elements is the identity element, there is a shortcut. We can handle several cases at the same time by setting one of the two general elements equal to the identity element and using a variable for the other general element. Recall that the identity element is y for T with respect to $\star$. Then, for all $a \in T$ we have:

$$a \star y = a \quad y \star a = a$$

Now, note that if the two general elements are the same, there is nothing to check. For all $a \in T$, we trivially have that $a \star a = a \star a$. So, the only remaining case to check is covered here:

$$x \star z = y \quad z \star x = y$$

We have shown that $a \star b = b \star a$ for all $a \in T$ and $b \in T$. Thus, the binary operation $\star : T \times T \rightarrow T$ is commutative.

When we have an operation on a set given by an operation table, we can determine whether or not the operation is commutative by observing whether or not the operation table possesses a particular symmetry. We locate the diagonal of the table from the operation symbol in the top left corner of the table to the bottom right corner of the table. Then, we determine whether or not that diagonal acts as a mirror for the other entries in the table. If so, the operation is commutative.

Example 13.5.4. With the above comment in mind, we revisit Example 13.5.3. We shade the diagonal that must act as a mirror for the other entries in the table if the operation is commutative. Then, we individually verify the symmetry by pointing out the pairs of entries that need to match and noting that they do, in fact, match.

$\star$	x	y	z
x	z	x	y
y	x	y	z
z	y	z	x

$\star$	x	y	z
x	z	x	y
y	x	y	z
z	y	z	x

$\star$	x	y	z
x	z	x	y
y	x	y	z
z	y	z	x

Example 13.5.5. Consider the binary operation $\oplus : \mathbb{Z}_5 \times \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ defined by $a \oplus b = (a + b) \bmod 5$. We follow an approach that is similar to that from Example 13.2.4 to show that $\oplus$ is commutative. Let $a \in \mathbb{Z}_5$ and $b \in \mathbb{Z}_5$. By the definition of $\oplus$ and the commutativity of addition of integers we have

$$a \oplus b = (a + b) \bmod 5 = (b + a) \bmod 5 = b \oplus a.$$

Thus $\oplus$ is commutative

Problem 13.5.6. Let $A = \{g, h, c, d\}$ and let $\diamond : A \times A \rightarrow A$ be defined by the table:

$\diamond$	g	h	c	d
g	g	h	c	d
h	h	g	d	c
c	c	d	g	
d	d	c	h	g

Which element in the box makes the operation $\diamond$ commutative?

Solution. The operation $\diamond$ is commutative if for all a and b in A we have $a \diamond b = b \diamond a$. In particular we must have $d \diamond c = c \diamond d$. Since $d \diamond c = h$ we must also have $c \diamond d = h$. Hence the element h in the box makes $\diamond$ commutative.

Problem 13.5.7. Give an example of a binary operation that is not commutative.

Solution. Consider the binary operation subtraction $- : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$. Since $3 - 2 = 1$ and $2 - 3 = -1$, and $1 \neq -1$, the binary operation $-$ is not commutative.

Problem 13.5.8. Decide which of these binary operations are commutative.

- (i) $\otimes : \mathbb{Z}_{11}^{\otimes} \times \mathbb{Z}_{11}^{\otimes} \rightarrow \mathbb{Z}_{11}^{\otimes}$ given by $a \otimes b = (a \cdot b) \bmod 11$
- (ii) $\oplus : \mathbb{Z}_{11} \times \mathbb{Z}_{11} \rightarrow \mathbb{Z}_{11}$ given by $a \oplus b = (a + b) \bmod 11$
- (iii) $\ominus : \mathbb{Z}_{11} \times \mathbb{Z}_{11} \rightarrow \mathbb{Z}_{11}$ given by $a \ominus b = (a - b) \bmod 11$

Solution. (i) We know multiplication of integers is commutative. That is, for all integers a and b we have $(a \cdot b) = (b \cdot a)$. Thus

$$a \otimes b = (a \cdot b) \bmod 11 = (b \cdot a) \bmod 11 = b \otimes a,$$

which means that the binary operation $\otimes$ is commutative.

- (ii) We know addition of integers is commutative. We can proceed as in (i), and use this fact inside of the mod operator to prove that $\oplus : \mathbb{Z}_{11} \times \mathbb{Z}_{11} \rightarrow \mathbb{Z}_{11}$ given by $a \oplus b = (a + b) \bmod 11$ is also a commutative binary operation.
- (iii) We know that subtraction of integers is not commutative. So we suspect that the binary operation $\ominus$ that is based on subtraction is not commutative. We find a counterexample. Let $a := 1$ and $b := 0$. Then

$$a \ominus b = 1 \ominus 0 = (1 - 0) \bmod 11 = 1 \bmod 11 = 1$$

and

$$b \ominus a = 0 \ominus 1 = (0 - 1) \bmod 11 = (-1) \bmod 11 = 10.$$

We have found a and b such that $a \ominus b$ is not equal to $b \ominus a$. So the binary operation $\ominus$ is not commutative.

Chapter 14

Groups

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Reproduce the definition of a group.
- (2) Recognize whether a set with a given binary operation is a group.
- (3) Compute with elements in a group.
- (4) Find the identity element of a group.
- (5) Find the inverse of an element in a group.
- (6) Recognize whether a set with modular multiplication is a group.

Groups are simple mathematical structures that only consist of a set and a binary operation on that set with certain properties, namely those that we investigated in the previous section. Groups can be found in many areas of mathematics and are used to describe symmetries in other fields such as chemistry and physics. We define groups, give examples of groups, and introduce two collections of infinitely many groups.

14.1 Definition of a Group

We now have the terminology needed to formally define a commutative group.

Definition 14.1.1. A pair $(G, \bullet)$ consisting of a set G and a binary operation $\bullet : G \times G \rightarrow G$ is a *commutative group* if the following properties hold:

- (i) **Identity:** There is an element $e \in G$ such that for all $a \in G$ we have $a \bullet e = e \bullet a = a$.
The element e is called the *identity element*.
- (ii) **Inverses:** For each $a \in G$ there is $b \in G$ such that $a \bullet b = b \bullet a = e$, where e is the identity element in G with respect to $\bullet$.
The element b is called an *inverse* of a .
- (iii) **Associativity:** The operation $\bullet$ is associative.
So, $a \bullet (b \bullet c) = (a \bullet b) \bullet c$ for all $a \in G$, $b \in G$, and $c \in G$.
- (iv) **Commutativity:** The operation $\bullet$ is commutative.
So, $a \bullet b = b \bullet a$ for all $a \in G$ and $b \in G$.

Commutative groups are also called *abelian* groups after the Norwegian mathematician Niels Abel (1802 – 1829). A group that does not satisfy property (iv) is simply referred to as a *group*, or more specifically, a *non-commutative* group or *non-abelian* group. As we only consider commutative groups in this course, when we say group, we are referring to a commutative group. We call the operation $\bullet$ of a group $(G, \bullet)$ the *group operation* of the group.

Recall that in Theorem 13.3.3, we showed that a set with a binary operation has at most one identity element. So the identity element in a group is unique. Similarly we can show that each element of a group has exactly one inverse with respect to the group operation $\bullet$, allowing us to speak of *the* inverse of an element. Recall that we denote the inverse of an element a with respect to the operation $\bullet$ by $a^{-1\bullet}$.

Theorem 14.1.2. *Let $(G, \bullet)$ be a group with identity element $e \in G$. Then, for each element $a \in G$, there is exactly one element $b \in G$ such that $a \bullet b = e$ and $b \bullet a = e$, implying that the inverse of each element $a \in G$ is the unique element $b = a^{-1\bullet}$.*

Proof. Let $(G, \bullet)$ be a group with identity element $e \in G$. Suppose that $b \in G$ and $c \in G$ are both inverses of the element a in $(G, \bullet)$. Then:

$$\begin{array}{ll}
 b = b \bullet e & \text{since } e \text{ is the identity element of } (G, \bullet) \\
 = b \bullet (a \bullet c) & \text{since } a \text{ and } c \text{ are inverses in } (G, \bullet) \\
 = (b \bullet a) \bullet c & \text{since } (G, \bullet) \text{ is associative} \\
 = e \bullet c & \text{since } a \text{ and } b \text{ are inverses in } (G, \bullet) \\
 = c & \text{since } e \text{ is the identity element of } (G, \bullet)
 \end{array}$$

Since $b = c$, there is exactly one inverse of a in $(G, \bullet)$, and we write the inverse of a as $a^{-1\bullet}$. □

14.2 Examples of Groups

In order to determine whether or not a set with a binary operation defined on the set forms a group, we must investigate whether or not each of the properties (i) to (iv) from Definition 14.1.1 are met. If all of the properties are met, we conclude that the set with the operation defined on it forms a group. If even one of the properties is not met, we conclude that the set with the operation defined on it does not form a group.

We begin by piecing together information developed in the examples of Section 13.

Problem 14.2.1. *Is the set $\mathbb{Z}$ of integers with addition a commutative group?*

Solution. As the sum of two integers is an integer, addition is a binary operation on the set of integers $\mathbb{Z}$. We check whether the set of integers $\mathbb{Z}$ with the operation addition (+) fulfills the properties (i) to (iv) from Definition 14.1.1:

- (i) **Identity:** For all $a \in \mathbb{Z}$ we have $a + 0 = a$ and $0 + a = a$, hence the integer 0 is the identity element with respect to addition. (Compare Example 13.3.2 (a).)
- (ii) **Inverses:** For all $a \in \mathbb{Z}$ we have $a + (-a) = 0$ and $(-a) + a = 0$, hence the inverse of a with respect to addition is the integer $-a$. So every integer has an inverse. (Compare Example 13.4.6 (a).)
- (iii) **Associativity:** Addition of integers is associative. (See Example 13.2.2 (a).)

(iv) **Commutativity:** Addition of integers is commutative. (See Example 13.5.2 (a).)

Thus $(\mathbb{Z}, +)$ is a commutative group.

Problem 14.2.2. *Is the set $\mathbb{N}$ of natural numbers with multiplication a commutative group?*

Solution. By Example 13.4.6 (b), there is a natural number that does not have a multiplicative inverse. So, property (ii) is not met, and we conclude that the set of natural numbers with multiplication is not a commutative group. The properties (i), (iii), and (iv) are met by part (b) of Examples 13.3.2, 13.2.2, and 13.5.2, respectively.

Problem 14.2.3. *Is the set $T = \{x, y, z\}$ with the binary operation $\star : T \rightarrow T$ that is given by the table in Example 13.1.3 a commutative group?*

Solution. We consider the properties (i) to (iv) from Definition 14.1.1:

- (i) **Identity:** The identity element is y . (See Examples 13.3.4 and 13.3.5.)
- (ii) **Inverses:** The inverse of x is $x^{-1\star} = z$, the inverse of y is $y^{-1\star} = y$, and the inverse of z is $z^{-1\star} = x$, so every element of T has an inverse. (See Examples 13.4.8 and 13.4.9.)
- (iii) **Associativity:** $\star$ is associative. (See Example 13.2.3.)
- (iv) **Commutativity:** $\star$ is commutative. (See Examples 13.5.3 and 13.5.4.)

Thus $(T, \star)$ is a commutative group.

Similarly it follows from Examples 13.3.6, 13.4.10, 13.2.4, and 13.5.5 that $(\mathbb{Z}_5, \oplus)$ where $\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$ and $\oplus : \mathbb{Z}_5 \times \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$ is defined by $a \oplus b = (a + b) \bmod 5$ is a group.

Now that we have considered all of the sets and operations that were given in Section 13, we will provide some additional problems for variety. A commutative group must contain at least one element, namely the identity. The following problem demonstrates that there are commutative groups with just one element.

Problem 14.2.4. *Is $(\{1\}, \cdot)$, where $\cdot$ is multiplication, a commutative group?*

Solution. We consider the properties (i) to (iv) from Definition 14.1.1. Since $1 \cdot 1 = 1 \in \{1\}$ we have that $\cdot$ is a binary operation on $\{1\}$.

- (i) **Identity:** The identity element is 1.
- (ii) **Inverses:** The inverse of 1 is $1^{-1\cdot} = 1$, so every element of $\{1\}$ has an inverse.
- (iii) **Associativity:** $1 \cdot (1 \cdot 1) = 1 \cdot 1 = 1$ and $(1 \cdot 1) \cdot 1 = 1 \cdot 1 = 1$. Since $1 = 1$, we have that $\cdot$ is associative.
- (iv) **Commutativity:** $1 \cdot 1 = 1 \cdot 1$, so $\cdot$ is commutative.

Thus $(\{1\}, \cdot)$ is a commutative group.

We conclude by taking one more look at a set with an operation defined by a table.

Example 14.2.5. Let $S = \{f, g, h, j\}$, and let the operation $\diamond : S \diamond S \rightarrow S$ be given by the operation table:

$\diamond$	f	g	h	j
f	f	g	h	j
g	g	f	j	h
h	h	j	g	f
j	j	h	f	g

Each entry in the table is an element in S , so $\diamond$ is a binary operation on S . We show that $(S, \diamond)$ is a commutative group by verifying that the properties (i) to (iv) from Definition 14.1.1 hold.

- (i) **Identity:** The row corresponding to the element $\mathbf{f}$ matches the header row at the top of the table and the column corresponding to the element $\mathbf{f}$ matches the header column on the left side of the table. Thus the identity element is $\mathbf{f}$.
- (ii) **Inverses:** Since $\mathbf{f}$ is the identity element, we begin by locating all of the places $\mathbf{f}$ appears in the table. For each table entry that is $\mathbf{f}$, we trace back to the header column on the left side of the table and the header row on the top of the table. Since $\mathbf{f} \diamond \mathbf{f} = \mathbf{f}$, we conclude that $\mathbf{f}$ is its own inverse, and since $(\mathbf{g}) \diamond (\mathbf{g}) = \mathbf{f}$, we conclude that $\mathbf{g}$ is also its own inverse. Finally, since $\mathbf{h} \diamond (\mathbf{j}) = \mathbf{f}$ and $(\mathbf{j}) \diamond \mathbf{h} = \mathbf{f}$, we conclude that $\mathbf{h}$ and $\mathbf{j}$ are inverses of each other. Thus every element of S has an inverse.
- (iii) **Associativity:** Exhausting all 64 possibilities, we would be able to see that the operation $\diamond$ is associative.
- (iv) **Commutativity:** The table is symmetric about the diagonal from the operation symbol in the top left corner of the table to the bottom right corner of the table. Thus $\diamond$ is commutative.

Thus $(S, \diamond)$ is a commutative group.

14.3 Modular Addition and Multiplication

In section 3.4 we have encountered the addition of hours, weekdays, and months as an example for modular arithmetic. We now introduce binary operations on the sets $\mathbb{Z}_n = \{0, 1, 2, \dots, n-1\}$ where $n \in \mathbb{N}$ based on the addition and multiplication of integers. For a and b in $\mathbb{Z}_n$ we consider $(a + b) \bmod n$ and $(a \cdot b) \bmod n$. Because the remainder of division by n is always an element of $\mathbb{Z}_n = \{0, 1, 2, \dots, n-1\}$ these yield binary operations on $\mathbb{Z}_n$.

Definition 14.3.1. Let $n \in \mathbb{N}$. We define two binary operations on the set

$$\mathbb{Z}_n = \{0, 1, 2, \dots, n-1\}.$$

- (i) We call $\oplus : \mathbb{Z}_n \times \mathbb{Z}_n \rightarrow \mathbb{Z}_n$, $a \oplus b := (a + b) \bmod n$ *addition modulo n* .
- (ii) We call $\otimes : \mathbb{Z}_n \times \mathbb{Z}_n \rightarrow \mathbb{Z}_n$, $a \otimes b := (a \cdot b) \bmod n$ *multiplication modulo n* .

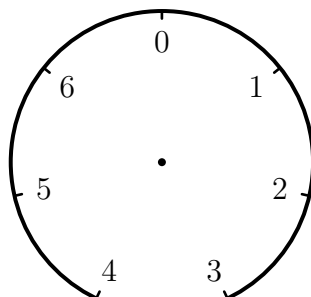
Example 14.3.2. We present examples for addition and multiplication modulo 7. Let $a \oplus b := (a + b) \bmod 7$ and $a \otimes b := (a \cdot b) \bmod 7$. Tables for the binary operations $\oplus$ and $\otimes$ are given in Figure 14.3.1.

- (i) $5 \otimes 4 = (5 \cdot 4) \bmod 7 = 20 \bmod 7 = 6$
- (ii) $3 \oplus 4 = (3 + 4) \bmod 7 = 7 \bmod 7 = 0$
- (iii) $2 \otimes (3 \oplus 6) = 2 \otimes ((3+6) \bmod 7) = (2 \otimes (9 \bmod 7)) = 2 \otimes 2 = (2 \cdot 2) \bmod 7 = 4 \bmod 7 = 4$

We apply modular addition and multiplication in the definition of certain groups. We show that for any $n \in \mathbb{N}$, the set $\mathbb{Z}_n$ with addition modulo n is a group and that for any prime number p the set $\mathbb{Z}_p^\otimes$ with multiplication modulo p is a group.

Figure 14.3.1: Addition and multiplication tables for arithmetic modulo 7, that is, for the operations given by $a \oplus b = (a + b) \bmod 7$ and $a \otimes b = (a \cdot b) \bmod 7$.

$\oplus$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5



$\otimes$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

14.4 The additive groups $(\mathbb{Z}_n, \oplus)$

Before we prove that $(\mathbb{Z}_n, \oplus)$ where $a \oplus b = (a + b) \bmod n$ is a group for all $n \in \mathbb{N}$, we examine an example.

Problem 14.4.1. Show that $(\mathbb{Z}_7, \oplus)$ where $\mathbb{Z}_7 = \{0, 1, 2, 3, 4, 5, 6\}$ and $a \oplus b = (a + b) \bmod 7$ is a group.

Solution. We show that $\mathbb{Z}_7$ with $\oplus$ satisfies the properties of a group from Definition 14.1.1. As the remainder of division by 7 is always in $\mathbb{Z}_7$ we have that $\oplus$ is indeed a binary operation on $\mathbb{Z}_7$.

- (i) **Identity:** Because $a \oplus 0 = (a + 0) \bmod 7 = a$ and $0 \oplus a = (0 + a) \bmod 7 = a$ for all $a \in \mathbb{Z}_7$, 0 is the identity element of $\oplus$.
- (ii) **Inverse:** We have $0 \oplus 0 = (0 + 0) \bmod 7 = 0$. So 0 is the identity element in $\mathbb{Z}_7$. Let $a \in \mathbb{Z}_7$ with $a \neq 0$ and let $b = 7 - a$. Then,

$$a \oplus b = a \oplus (7 - a) = (a + 7 - a) \bmod 7 = (a - a + 7) \bmod 7 = 7 \bmod 7 = 0.$$

Thus b is the inverse of a with respect to $\oplus$.

- (iii) **Associativity:** Let $a \in \mathbb{Z}_7$, $b \in \mathbb{Z}_7$, and $c \in \mathbb{Z}_7$. By Theorem 3.4.5 we only need to show that $(a + (b + c)) \bmod 7 = ((a + b) + c) \bmod 7$. This holds since $a + (b + c) = (a + b) + c$ for all integers a , b , and c by the associative property of the integers. Hence $\oplus$ is associative.
- (iv) **Commutativity:** By the commutative property of the integers we have $a + b = b + a$ for all integers a and b . Thus also for all $a \in \mathbb{Z}_7$ and $b \in \mathbb{Z}_7$, we have $a + b = b + a$ and $a \oplus b = (a + b) \bmod 7 = (b + a) \bmod 7 = b \oplus a$. We can also deduce the commutativity of $\oplus$ from the symmetry of the addition table in Figure 14.3.1.

In general we have:

Theorem 14.4.2. Let $n \in \mathbb{N}$. The set $\mathbb{Z}_n = \{0, 1, 2, \dots, n - 1\}$ with the operation $\oplus : \mathbb{Z}_n \times \mathbb{Z}_n \rightarrow \mathbb{Z}_n$, $a \oplus b = (a + b) \bmod n$ is a group.

Proof. We show that $(\mathbb{Z}_n, \oplus)$ satisfies properties (i) to (iv) from Definition 14.1.1.

- (i) **Identity:** Let $a \in \mathbb{Z}_n$. We have $a \oplus 0 = (a + 0) \bmod n = a \bmod n = a$ and similarly $0 \oplus a = (0 + a) \bmod n = a \bmod n = a$. Hence 0 is an identity element with respect to $\oplus$.
- (ii) **Inverses:** We have $0 \oplus 0 = (0 + 0) \bmod n = 0$. Thus 0 is the inverse of 0 in $\mathbb{Z}_n$ with respect to $\oplus$. Now consider $a \in \mathbb{Z}_n$ and $a \neq 0$. Let $b = n - a$. So $b \in \mathbb{Z}_n$. Then

$$a \oplus b = a \oplus (n - a) = (a + (n - a)) \bmod n = (a - a + n) \bmod n = n \bmod 0.$$

Thus $n - a = b$ is the inverse of a .

- (iii) **Associativity:** The associativity of $\oplus$ follows from the associativity of $+$. Let $a \in \mathbb{Z}_n$, $b \in \mathbb{Z}_n$, and $c \in \mathbb{Z}_n$. By Theorem 3.4.5 we only need to show that $(a + (b + c)) \bmod n = ((a + b) + c) \bmod n$. This holds since $a + (b + c) = (a + b) + c$ for all integers a , b , and c by the associative property of the integers. Hence $\oplus$ is associative.
- (iv) **Commutativity:** By the commutative property of the integers we have $a + b = b + a$ for all integers a and b . Thus also for all $a \in \mathbb{Z}_n$ and $b \in \mathbb{Z}_n$ we have $a + b = b + a$ and $a \oplus b = (a + b) \bmod n = (b + a) \bmod n = b \oplus a$.

□

Directly from the proof of Theorem 14.4.2(ii) we obtain a method for finding inverses in $(\mathbb{Z}_n, \oplus)$. Namely if $a \in \mathbb{Z}_n$ and $a \neq 0$ then $b = n - a \in \mathbb{Z}_n$ and $a \oplus b = 0$.

Problem 14.4.3. Find the inverse of 5 in the group $(\mathbb{Z}_{12}, \oplus)$ where $a \oplus b = (a + b) \bmod 12$.

Solution. We have $5 \oplus 7 = (5 + 7) \bmod 12 = 12 \bmod 12 = 0$. As the group $(\mathbb{Z}_{12}, \oplus)$ is commutative this shows that 7 is the inverse of 5.

14.5 The multiplicative groups $(\mathbb{Z}_p^\otimes, \otimes)$

In Section 14.4 we had seen that for all natural numbers m the set $\mathbb{Z}_m = \{0, 1, 2, \dots, m - 1\}$ with addition modulo m is a group. In this section, we form a group using the operation $\otimes$ as well.

We first consider $\mathbb{Z}_7$ with the binary operation $\otimes : \mathbb{Z}_7 \times \mathbb{Z}_7 \rightarrow \mathbb{Z}_7$ given by $a \otimes b = (a \cdot b) \bmod 7$. From the multiplication table in Figure 14.3.1 we see that 1 is the only possibility of an identity element with respect to $\otimes$. We also see that 0 does not have an inverse with respect to $\otimes$. Thus $\mathbb{Z}_7$ with $\otimes$ does not satisfy Definition 14.1.1(ii). We remedy this situation by excluding 0 from the set and show that $(\mathbb{Z}_7^\otimes, \otimes)$ is a group. We will use the properties of prime numbers to do this. Recall that $p \in \mathbb{N}$ is a prime number if its only divisors are 1 and p . Natural numbers that are not prime numbers are called composite.

Problem 14.5.1. Show that the set $\mathbb{Z}_7^\otimes = \{1, 2, 3, 4, 5, 6\}$ with the operation $a \otimes b = (a \cdot b) \bmod 7$ is a group.

Solution. We show that $\mathbb{Z}_7^\otimes$ with $\otimes$ satisfies the properties of a groups from Definition 14.1.1. Because 7 is a prime number, the product of two numbers that are not divisible by 7 is also not divisible by 7. Again because 7 is prime, none of the elements in $\mathbb{Z}_7^\otimes$ are divisible by

seven. Thus the product of two elements of $\mathbb{Z}_7^\otimes$ is not divisible by 7, and its remainder is not 0. Thus $\otimes$ is a binary operation on $\mathbb{Z}_7^\otimes$.

- (i) **Identity:** Since $a \cdot 1 = a$ and $1 \cdot a = a$ for all integers a we have $a \otimes 1 = (a \cdot 1) \bmod 7 = a$ and $1 \otimes a = (1 \cdot a) \bmod 7 = a \bmod 7 = a$ for all $a \in \mathbb{Z}_7^\otimes$. Hence 1 is the identity with respect to $\otimes$.
- (ii) **Inverses:** From the multiplication table in Figure 14.3.1 we get $2 \otimes 4 = 1$, $3 \otimes 5 = 1$, $4 \otimes 2 = 1$, $5 \otimes 3 = 1$, and $6 \otimes 6 = 1$. Thus each element in $\mathbb{Z}_7^\otimes$ has an inverse.
- (iii) **Associativity:** Let $a \in \mathbb{Z}_7^\otimes$, $b \in \mathbb{Z}_7^\otimes$, and $c \in \mathbb{Z}_7^\otimes$. By Theorem 3.4.8 we only need to show that $(a \cdot (b \cdot c)) \bmod 7 = ((a \cdot b) \cdot c) \bmod 7$. This holds since $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ for all integers a , b , and c by the associative property of the integers. Hence $\otimes$ is associative.
- (iv) **Commutativity:** By the commutative property of multiplication of integers we have $a \cdot b = b \cdot a$ for all integers a and b . Thus also for all $a \in \mathbb{Z}_7^\otimes$ and $b \in \mathbb{Z}_7^\otimes$ we have $a \cdot b = b \cdot a$ and $a \otimes b = (a \cdot b) \bmod 7 = (b \cdot a) \bmod 7 = b \otimes a$. We can also deduce the commutativity of $\otimes$ from the symmetry of the multiplication table in Figure 14.3.1.

We have seen from the example above that $\mathbb{Z}_m$ with the binary operation $\otimes : \mathbb{Z}_m \times \mathbb{Z}_m \rightarrow \mathbb{Z}_m$ is not a group because $0 \in \mathbb{Z}_m$ does not have an inverse with respect to $\otimes$.

Now, we will investigate for which natural numbers m we have an operation $\otimes : \mathbb{Z}_m^\otimes \times \mathbb{Z}_m^\otimes \rightarrow \mathbb{Z}_m^\otimes$ on the set $\mathbb{Z}_m^\otimes$ to form a group. Namely, we will show that this is not possible when m is a composite number.

Theorem 14.5.2. *If $m \in \mathbb{N}$ is a composite number, the set $\mathbb{Z}_m^\otimes = \{1, \dots, m-1\}$ with the operation $a \otimes b = (a \cdot b) \bmod m$ is not a group.*

Proof. As m is composite, there are natural numbers $k \neq 1$ and $l \neq 1$ such that $m = k \cdot l$, and $k < m$ and $l < m$ so $k \in \mathbb{Z}_m^\otimes$ and $l \in \mathbb{Z}_m^\otimes$. We get $k \otimes l = (k \cdot l) \bmod m = m \bmod m = 0$ which is not an element of $\mathbb{Z}_m^\otimes$. So if m is composite, $\otimes$ is not a binary operation on the set $\mathbb{Z}_m^\otimes$, and we cannot form a group. $\square$

Example 14.5.3. We consider the set $\mathbb{Z}_6$ with the binary operation $\otimes : \mathbb{Z}_6 \times \mathbb{Z}_6 \rightarrow \mathbb{Z}_6$ given by $a \otimes b = (a \cdot b) \bmod 6$. Note that $\otimes$ is not a binary operation on $\mathbb{Z}_6^\otimes$ as $2 \otimes 4 = 0 \notin \mathbb{Z}_6^\otimes$. The operation table for $\otimes$ is:

$\otimes$	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	1	2	3	4	5
2	0	2	4	0	2	4
3	0	3	0	3	0	3
4	0	4	3	0	4	2
5	0	5	4	3	2	1

From the table we see that 1 is the identity element with respect to $\otimes$. The only elements that have a 1 in their row (or column) are 1 and 5. So 0, 2, 3, and 4 do not have inverses with respect to $\otimes$. Hence $\mathbb{Z}_6$ with the operation $\otimes$ is not a group.

Thus $\mathbb{Z}_m^\otimes$ with modular multiplication can only be a group if m is not composite, that is when m is a prime number.

If $p \in \mathbb{N}$ is prime, we still need to check that $\otimes$ is a binary operation on $\mathbb{Z}_p^\otimes$ and that every element in $\mathbb{Z}_p^\otimes$ has an inverse. In the following problem, we compute the inverse of an element in $\mathbb{Z}_7^\otimes$ with respect to modular multiplication.

Problem 14.5.4. Find $b \in \mathbb{Z}_7^\otimes$ such that $5 \cdot b \bmod 7 = 1$.

Solution. As there are only 6 elements in $\mathbb{Z}_7^\otimes$, we decide to try them all until we find the solution. We have $(5 \cdot 1) \bmod 7 = 5 \neq 1$, $(5 \cdot 2) \bmod 7 = 10 \bmod 7 = 3 \neq 1$, $(5 \cdot 3) \bmod 7 = 15 \bmod 7 = 1$ so we have found the solution $b = 3$ and do not need to continue our search.

We show that $\mathbb{Z}_p^\otimes = \{1, 2, \dots, p-1\}$ with the operation given by $a \otimes b = (a \cdot b) \bmod p$ is a group. In particular, when p is a prime number any element in $\mathbb{Z}_p^\otimes$ has a multiplicative inverse in $\mathbb{Z}_p^\otimes$ with respect to $\otimes$.

Theorem 14.5.5. If $p \in \mathbb{N}$ is a prime number and $a \in \mathbb{Z}_p^\otimes$, then there is $b \in \mathbb{Z}_p^\otimes$ such that $a \otimes b = (a \cdot b) \bmod p = 1$, that is, b is the multiplicative inverse of a with respect to multiplication modulo p .

Proof. As $1 \leq a \leq p-1$ and p is prime, we have $\gcd(a, p) = 1$. By Bézout's theorem (Theorem 4.4.1) there are $s \in \mathbb{Z}$ and $t \in \mathbb{Z}$ such that $(s \cdot a) + (t \cdot p) = 1$, hence $(s \cdot a) = 1 - (t \cdot p)$ and $(s \cdot a) \bmod p = 1$. Thus $s \bmod p$ is the inverse of a with respect to $\otimes$. $\square$

The Euclidean algorithm (Algorithm 4.3.1) along with the computation of the quotients is everything that is needed to find the values of s and t in Bézout's identity, so it is possible to develop a method of finding modular multiplicative inverses. In particular if for a prime p and $1 \leq a \leq (p-1)$ the s from Bézout's identity for $\gcd(a, p)$ is known, we can easily find the inverse of a in $(\mathbb{Z}_p^\otimes, \otimes)$.

Strategy 14.5.6. Let p be a prime number and $1 \leq a \leq p-1$. Let s and t be such that

$$(s \cdot a) + (t \cdot p) = \gcd(a, p) = 1.$$

Then the inverse $a^{-1 \otimes}$ of a in the group $(\mathbb{Z}_p^\otimes, \otimes)$ is $s \bmod p$. That is, $a^{-1 \otimes} = s \bmod p$.

Example 14.5.7. We have that $\gcd(19, 7) = 1$. By Bézout's Identity (Theorem 4.4.1) there are $s \in \mathbb{Z}$ and $t \in \mathbb{Z}$ such that $(s \cdot 19) + (t \cdot 7) = \gcd(19, 7)$. Possible solutions for s and t are $s = 3$ and $t = -8$. We get

$$(3 \cdot 19) + ((-8) \cdot 7) = \gcd(19, 7) = 1.$$

So $(3 \cdot 19) + ((-8) \cdot 7) = 1$. Using modular arithmetic, $((-3 \cdot 19) + ((-8) \cdot 7)) \bmod 19 = 1 \bmod 19$. Recalling that the order in which we perform the mod and the arithmetic does not change the outcome, observe that $(-3 \cdot 19) \bmod 19 = 0$. So $((-8 \cdot 7) \bmod 19 = 1$, and $(-8) \bmod 19 = 11$. Hence $(7 \cdot 11) \bmod 19 = 1$, and 11 is the inverse of 7 in $(\mathbb{Z}_{19}^\otimes, \otimes)$.

Problem 14.5.8. Knowing that $\gcd(113, 80) = 1 = (17 \cdot 113) + ((-24) \cdot 80)$, find the inverse of 80 in the group $(\mathbb{Z}_{113}^\otimes, \otimes)$.

Solution. We have $((-24) \cdot 80) + (17 \cdot 113) \bmod 113 = 1$. Hence $((-24) \cdot 80) \bmod 113 = 1$. Thus the inverse of 80 is $(-24) \bmod 113 = 89$.

Theorem 14.5.9. *Let $p \in \mathbb{N}$ be a prime number. The set $\mathbb{Z}_p^\otimes = \{1, \dots, p-1\}$ with the operation $a \otimes b = (a \cdot b) \bmod p$ is a group.*

Proof. We show that $\mathbb{Z}_p^\otimes$ with $\otimes$ satisfies the properties of a group from Definition 14.1.1.

- (i) **Identity:** Since $a \cdot 1 = a$ and $1 \cdot a = a$ for all integers a we have $a \otimes 1 = (a \cdot 1) \bmod p = a$ and $1 \otimes a = (1 \cdot a) \bmod p = a \bmod p = a$ for all $a \in \mathbb{Z}_p^\otimes$. Hence 1 is the identity with respect to $\otimes$.
- (ii) **Inverse:** Since p is prime every $a \in \mathbb{Z}_p^\otimes$ has an inverse with respect to $\otimes$ by Theorem 14.5.5.
- (iii) **Associativity:** Let $a \in \mathbb{Z}_p^\otimes$, $b \in \mathbb{Z}_p^\otimes$, and $c \in \mathbb{Z}_p^\otimes$. By Theorem 3.4.8 we only need to show that $(a \cdot (b \cdot c)) \bmod p = ((a \cdot b) \cdot c) \bmod p$. This holds since $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ for all integers a , b , and c by the associative property of the integers. Hence $\otimes$ is associative.
- (iv) **Commutativity:** By the commutative property of multiplication of integers we have $a \cdot b = b \cdot a$ for all integers a and b . Thus also for all $a \in \mathbb{Z}_p^\otimes$ and $b \in \mathbb{Z}_p^\otimes$ we have $a \cdot b = b \cdot a$ and $a \otimes b = (a \cdot b) \bmod p = (b \cdot a) \bmod p = b \otimes a$.

□

Problem 14.5.10. *In the group $(\mathbb{Z}_5^\otimes, \otimes)$ where $a \otimes b := (a \cdot b) \bmod 5$ find the inverses of all elements of $\mathbb{Z}_5^\otimes$.*

Solution. The numbers in this problem are small enough for trial and error works well enough. Recall that $\mathbb{Z}_5^\otimes = 1, 2, 3, 4$.

The inverse of 1 We try the values in 1,2,3,4 until we succeed.

$1 \otimes 1 = 1$, so the inverse of 1 is 1.

The inverse of 2 : We try the values in 1,2,3,4 until we succeed.

$1 \otimes 2 = 2$, so 1 is not the inverse of 2.

$2 \otimes 2 = 4$, so 2 is not the inverse of 2.

$3 \otimes 2 = 1$, so 3 could be the inverse of 2. As $2 \otimes 3 = 1$, the inverse of 2 is 3.

As we have found the inverse we do not have to keep trying.

The inverse of 3 : As 3 is the inverse of 2, we have that 2 is the inverse of 3.

The inverse of 4 : We try the values in 1,2,3,4 until we succeed.

$1 \otimes 4 = 4$, so 1 is not the inverse of 4.

$2 \otimes 4 = 3$, so 2 is not the inverse of 4.

$3 \otimes 4 = 2$, so 3 is not the inverse of 4.

$4 \otimes 4 = 1$, so 4 is the inverse of itself.

Chapter 15

Powers and Logarithms

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Compute powers of group elements.
- (2) Compute powers whose exponent is a power of 2 using repeated squaring.
- (3) Compute powers of group elements using fast exponentiation.
- (4) Compute discrete logarithms of group elements.

15.1 Exponentiation

Recall that $\cdot : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{N}$ (multiplication) is a binary operation on the set $\mathbb{Z}$ of integers. We defined exponentiation as repeated multiplication. For $a \in \mathbb{Z}$ and $n \in \mathbb{N}$ introduced the notation

$$a^n := \underbrace{a \cdot a \cdot \dots \cdot a}_{n \text{ copies of } a}$$

and also defined $a^0 := 1$. Following the definition of powers of integers in Definition 1.4.1, we introduce exponentiation notation for group elements as repeated application of the group operation. To be able to distinguish exponentiation with respect to different binary operations in our notation of powers of group elements we always give the binary operation next to the exponent.

Definition 15.1.1. Let $(G, \star)$ be a group and $b \in G$.

- (i) We set $b^{0\star} = e$ where $e \in G$ is the identity of the group $(G, \star)$.
- (ii) For $n \in \mathbb{N}$ we set $b^{n\star} = \underbrace{b \star b \star \dots \star b}_{n \text{ copies of } b}$.

We read $b^{n\star}$ as “ b to the n by $\star$ ” and call b the *base* and n the *exponent*.

It follows from the definition that in a group $(G, \star)$ we have $b^{1\star} = b$ for all $b \in G$. For the identity e of any group we have $e^{x\star} = e$ for all $x \in \mathbb{W}$.

The properties of powers of integers from Theorem 1.4.5) also hold for powers of group elements:

Theorem 15.1.2. Let $(G, \star)$ be a group and $b \in G$. Let $m \in \mathbb{N}$ and $n \in \mathbb{N}$. In $(G, \star)$ we have

- (i) $b^{m\star} \star b^{n\star} = b^{(m+n)\star}$
- (ii) $(b^{m\star})^{n\star} = b^{(m \cdot n)\star}$

Proof. Both statements are proven by counting the number of copies of the base b when rewriting the powers according to Definition 15.1.1.

$$\begin{aligned}
 \text{(i)} \quad b^{m\star} \star b^{n\star} &= \underbrace{b \star \cdots \star b}_{m \text{ copies of } b} \star \underbrace{b \star \cdots \star b}_{n \text{ copies of } b} = \underbrace{b \star \cdots \star b}_{m+n \text{ copies of } b} = b^{(m+n)\star} \\
 \text{(ii)} \quad (b^{m\star})^{n\star} &= \underbrace{b^{m\star} \star \cdots \star b^{m\star}}_{n \text{ copies of } b^{m\star}} = \underbrace{\underbrace{b \star \cdots \star b}_{m \text{ copies of } b} \star \cdots \star \underbrace{b \star \cdots \star b}_{m \text{ copies of } b}}_{n \text{ copies of } \underbrace{(b \star \cdots \star b)}_{m \text{ copies of } b}} = \underbrace{b \star \cdots \star b}_{m \cdot n \text{ copies of } b} = b^{(m \cdot n)\star}
 \end{aligned}$$

□

Note that the notation of inverses with respect to binary operations in Definition 13.4.3 is chosen such that the properties proven above also work for negative exponents. In a group $(G, \star)$ with identity e we have for $a \in G$ that

$$a^{0\star} = e = a \star a^{-1\star} = a^{1\star} \star a^{-1\star} = a^{(1+(-1))\star}.$$

Although our definition of exponentiation works in every group we restrict our examples to the groups $(\mathbb{Z}_p^\otimes, \otimes)$ where p is a prime number the operation $\otimes : \mathbb{Z}_p^\otimes \times \mathbb{Z}_p^\otimes \rightarrow \mathbb{Z}_p^\otimes$ is given by $a \otimes b = (a \cdot b) \bmod p$. Specializing Definition 15.1.1 to the group $(\mathbb{Z}_p^\otimes, \otimes)$ we have for all $b \in \mathbb{Z}_p^\otimes$ that

$$b^{0\otimes} = 1$$

and for all $b \in \mathbb{Z}_p^\otimes$ and all $n \in \mathbb{N}$ that

$$b^{n\otimes} = \underbrace{b \otimes b \otimes \cdots \otimes b}_{n \text{ copies of } b} = \underbrace{(b \cdot b \cdot \cdots \cdot b)}_{n \text{ copies of } b} \bmod p = (b^n) \bmod p.$$

The second equality above holds because of Theorem 3.4.8.

Example 15.1.3. In $(\mathbb{Z}_{11}^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 11$ we have

- (i) $1^{0\otimes} = 1$ by Definition 15.1.1 (i)
- (ii) $2^{0\otimes} = 1$ by Definition 15.1.1 (i)
- (iii) $2^{1\otimes} = 2$ by Definition 15.1.1 (ii)
- (iv) $2^{2\otimes} = 2 \otimes 2 = (2 \cdot 2) \bmod 11 = 4 \bmod 11 = 4$ by Definition 15.1.1 (ii)
- (v) $2^{3\otimes} = 2 \otimes 2 \otimes 2 = (2 \cdot 2 \cdot 2) \bmod 11 = 8 \bmod 11 = 8$ by Definition 15.1.1 (ii)
- (vi) $2^{4\otimes} = 2 \otimes 2 \otimes 2 \otimes 2 = (2 \cdot 2 \cdot 2 \cdot 2) \bmod 11 = 16 \bmod 11 = 5$ by Definition 15.1.1 (ii)

When numbers become bigger the computations become easier when we compute mod after each multiplication.

Example 15.1.4. In the group $(\mathbb{Z}_{11}^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 11$ we compute

$$6^{8\otimes} = 6 \otimes 6 \otimes 6 \otimes 6 \otimes 6 \otimes 6 \otimes 6 \otimes 6.$$

We use two approaches.

- (i) We directly follow the definition, that is, we repeatedly apply $a \otimes b = (a \cdot b) \bmod 11$. We first compute

$$6^{2\otimes} = 6 \otimes 6 = (6 \cdot 6) \bmod 11 = 36 \bmod 11 = 3$$

We now compute the other powers up to $6^{8\otimes}$ making use of the previous result. In every step we apply Theorem 15.1.2(i).

$$6^{3\otimes} = 6^{2\otimes} \otimes 6 = 3 \otimes 6 = (3 \cdot 6) \bmod 11 = 18 \bmod 11 = 7$$

$$6^{4\otimes} = 6^{3\otimes} \otimes 6 = 7 \otimes 6 = (7 \cdot 6) \bmod 11 = 42 \bmod 11 = 9$$

$$6^{5\otimes} = 6^{4\otimes} \otimes 6 = 9 \otimes 6 = (9 \cdot 6) \bmod 11 = 54 \bmod 11 = 10$$

$$6^{6\otimes} = 6^{5\otimes} \otimes 6 = 10 \otimes 6 = (10 \cdot 6) \bmod 11 = 60 \bmod 11 = 5$$

$$6^{7\otimes} = 6^{6\otimes} \otimes 6 = 5 \otimes 6 = (5 \cdot 6) \bmod 11 = 30 \bmod 11 = 8$$

$$6^{8\otimes} = 6^{7\otimes} \otimes 6 = 8 \otimes 6 = (8 \cdot 6) \bmod 11 = 48 \bmod 11 = 4$$

We have computed $6^{8\otimes} = 4$.

- (ii) We compute 6^8 in the integers and then compute the result mod 11.

$$6^8 \bmod 11 = 1679616 \bmod 11 = 4$$

Note that we can easily conduct the computations in (i) by hand, but we would not want to compute 6^8 without the help of a calculator. When bases and exponents are larger, the second approach is not feasible anymore as the numbers become too large for most calculators.

Computing powers as in Example 15.1.4 (i), where we essentially follow Definition 15.1.1, is called *naive exponentiation*. This is the strategy that we already had used in Algorithm 2.6.1 for computing powers of integers. Replacing the multiplication of integers by the group operation we obtain a naive exponentiation algorithm for group elements.

Algorithm 15.1.5 (*Naive Exponentiation*).

Input: A group $(G, \star)$, $b \in G$, and a non-negative integer n

Output: $b^{n\star}$

- (1) **if** $n = 0$ **then return** the identity of $(G, \star)$
- (2) **if** $n = 1$ **then return** b
- (3) **let** $c := b$
- (4) **let** $i := 1$
- (5) **repeat**
 - (a) **let** $c := c \star b$
 - (b) **let** $i := i + 1$
- (6) **until** $i = n$
- (7) **return** c

15.2 Repeated Squaring

Because time is a valuable resource, we often look for ways of completing a given task as quickly as possible. In order to decide which way of completing the task is faster we compare

the time needed.

In this course the tasks are computations and we formulate ways of completing them as strategies or algorithms. Depending on who follows the instruction (say a human being or a computer) the time needed to perform a computation differs. So instead of measuring time, we count the number of operations needed to compare how fast a strategy or an algorithm is. This count of operations usually depends on the numbers involved in the computation. For algorithms we give this count depending on the input. This process is called *complexity analysis*. To simplify the analysis of our algorithms we only count the number of the most involved operations, which in this section are multiplications or group operations.

Theorem 15.2.1. *Let $(G, \star)$ be a group, $b \in G$, and $n \in \mathbb{N}$. The naive exponentiation algorithm (Algorithm 15.1.5) computes $b^{n\star}$ with $n - 1$ operations $\star$.*

Proof. In Algorithm 15.1.5 the operation $\star$ only occurs in step (4) (a) in the **repeat__until**-loop. Assuming that $n \neq 0, 1$, in step (4) the variable i is set to 1. From there, we enter into the **repeat__until**-loop.

After each operation $\star$ in step (4) (a) we add 1 to i in step (4) (b). Because the **repeat__until**-loop ends when i is equal to n , we follow the instructions in steps (4) (a) and (4) (b) exactly $n - 1$ times. Thus to compute $b^{n\star}$ with Algorithm 15.1.5 we need $n - 1$ operations $\star$. $\square$

In section 2, we gave two algorithms for computing c^4 for some integer c , Algorithm 2.2.3 and Algorithm 2.4.2. Although the output of both algorithms was the same, the number of multiplications to compute the output differed. Algorithm 2.2.3 computes $c^4 = c \cdot c \cdot c \cdot c$ which needs three multiplications. Algorithm 2.4.2 first computes $d := c \cdot c$ and then $c^4 = d \cdot d$ which need two multiplications.

In this section we employ the idea behind the latter algorithm to compute powers of group elements in the case when the exponent is a power of 2. This strategy is called *repeated squaring*. We first demonstrate it with an example.

Example 15.2.2. In the group $(\mathbb{Z}_{11}^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 11$ we compute $6^{8\otimes}$. Instead of the naive exponentiation method that we employed in Example 15.1.4 we use repeated squaring. We first compute

$$6^{2\otimes} = 6 \otimes 6 = (6 \cdot 6) \bmod 11 = 36 \bmod 11 = 3$$

By Theorem 15.1.2(ii) we have

$$6^{4\otimes} = 6^{(2 \cdot 2)\otimes} = (6^{2\otimes})^{2\otimes} = 6^{2\otimes} \otimes 6^{2\otimes} = 3 \otimes 3 = (3 \cdot 3) \bmod 11 = 9 \bmod 11 = 9.$$

Instead of the 3 operations $\otimes$ in which we computed $6^{4\otimes}$ in Example 15.1.4 we have computed $6^{4\otimes}$ in 2 operations $\otimes$. By Theorem 15.1.2(ii) we have

$$6^{8\otimes} = 6^{(4 \cdot 2)\otimes} = (6^{4\otimes})^{2\otimes} = 6^{4\otimes} \otimes 6^{4\otimes} = 9 \otimes 9 = (9 \cdot 9) \bmod 11 = 81 \bmod 11 = 4.$$

Instead of the 7 operations $\otimes$ in which we computed $6^{8\otimes}$ in Example 15.1.4 we have computed $6^{8\otimes}$ in 3 operations $\otimes$.

When a power of a group element is given we can easily find its square.

Problem 15.2.3. In the group $(\mathbb{Z}_{19843}^{\otimes}, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 19843$ we have $19^{1024\otimes} = 2327$. Find $19^{2048\otimes}$.

Solution. We notice that $2048 = 2 \cdot 1024$. So we compute

$$19^{2048\otimes} = (19^{1024\otimes})^{2\otimes} = 2327^{2\otimes} = (2327 \cdot 2327) \bmod 19843 = 5414929 \bmod 19843 = 17633.$$

When using the repeated squaring strategy to compute a power in a group $(G, \star)$ we start with squaring the base b to obtain $b^{2\star}$. Squaring $b^{2\star}$ yields $(b^{2\star})^{2\star} = b^{4\star}$. Squaring $b^{4\star}$ yields $(b^{4\star})^{2\star} = b^{8\star}$ and so on. Each time we square the exponent doubles. That means that the exponents after squaring s times is the product of s copies of 2 which is equal to 2^s . Thus we can use repeated squaring to compute powers of the form

$$b^{(2^s)\star}.$$

That is, the repeated squaring strategy works for any power, whose exponent is a power of 2.

Example 15.2.4. In the group $(\mathbb{Z}_{101}^{\otimes}, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 101$ we compute $3^{32\otimes}$ with repeated squaring. Note that $32 = 2^5$. We start with computing

$$3^{2\otimes} = 3 \otimes 3 = 9.$$

Now we use that $3^{4\otimes} = 3^{2 \cdot 2\otimes} = (3^{2\otimes})^{2\otimes} = 3^{2\otimes} \otimes 3^{2\otimes}$. Replacing $3^{2\otimes}$ by 9 we get

$$3^{4\otimes} = 3^{2\otimes} \otimes 3^{2\otimes} = 9 \otimes 9 = 81.$$

Now we use that $3^{8\otimes} = 3^{4 \cdot 2\otimes} = (3^{4\otimes})^{2\otimes} = 3^{4\otimes} \otimes 3^{4\otimes}$. Replacing $3^{4\otimes}$ by 81 we get

$$3^{8\otimes} = 3^{4\otimes} \otimes 3^{4\otimes} = 81 \otimes 81 = (81 \cdot 81) \bmod 101 = 6561 \bmod 101 = 97.$$

Now we use that $3^{16\otimes} = 3^{8 \cdot 2\otimes} = (3^{8\otimes})^{2\otimes} = 3^{8\otimes} \otimes 3^{8\otimes}$. Replacing $3^{8\otimes}$ by 97 we get

$$3^{16\otimes} = 3^{8\otimes} \otimes 3^{8\otimes} = 97 \otimes 97 = (97 \cdot 97) \bmod 101 = 9401 \bmod 101 = 16.$$

Now we use that $3^{32\otimes} = 3^{16 \cdot 2\otimes} = (3^{16\otimes})^{2\otimes} = 3^{16\otimes} \otimes 3^{16\otimes}$. Replacing $3^{16\otimes}$ by 16 we get

$$3^{32\otimes} = 3^{16\otimes} \otimes 3^{16\otimes} = 16 \otimes 16 = (16 \cdot 16) \bmod 101 = 256 \bmod 101 = 54.$$

We have found that $3^{32\otimes} = 54$. While the above process may seem awkward, we only needed to evaluate the binary operation $\otimes$ five times to compute the result. With the method from the previous section we would have needed 31 operations $\otimes$.

We formulate the repeated squaring strategy as an algorithm.

Algorithm 15.2.5 (*Repeated Squaring*).

Input: A group $(G, \star)$, $b \in G$, and a non-negative integer s

Output: $b^{(2^s)\star}$

```

(1) if  $s = 0$  then return  $b$ 
(2) let  $i := 1$ 
(3) let  $c := b \star b$ 
(4) repeat
    (a) let  $c := c \star c$ 
    (b) let  $i := i + 1$ 
(5) until  $i = s$ 
(6) return  $c$ 

```

To compute $b^{(2^s)\star}$ we compute a square s times. As each squaring needs on group operation $\star$ we can compute $b^{(2^s)\star}$ with s operations $\star$.

Theorem 15.2.6. *Let $\star$ be a binary operation on a set G and $a \in G$ and $n \in \mathbb{N}$. Then, using repeated squaring, $a^{(2^n)\star}$ can be computed with n operations $\star$.*

By Theorem 15.2.1 computing $a^{2^n\star}$ using the naive exponentiation algorithm (Algorithm 15.1.5)) needs $2^n - 1$ operations $\star$. So for $n > 1$ the repeated squaring strategy is faster.

Problem 15.2.7. *Let $\otimes : \mathbb{Z}_{19843}^\otimes \times \mathbb{Z}_{19843}^\otimes \rightarrow \mathbb{Z}_{19843}^\otimes$ be the binary operation given by $a \otimes b = (a \cdot b) \bmod 19843$.*

- (i) *How many operations $\otimes$ are needed to compute $19^{128\otimes}$ with the naive exponentiation method (repeated multiplication by 19) ?*
- (ii) *How many operations $\otimes$ are needed to compute $19^{128\otimes}$ with the repeated squaring method ?*

Solution. (i) By Theorem 15.2.1 $128 - 1 = 127$ operations $\otimes$ to compute $19^{128\otimes}$ with the naive exponentiation algorithm.

- (ii) As $128 = 2^7$ by Theorem 15.2.6 we need 7 operations $\otimes$ to compute $19^{128\otimes}$ using repeated squaring.

15.3 Fast Exponentiation

In the preceding Section 15.2 we saw that powers whose exponents are powers of two can be computed very efficiently. In the fast exponentiation strategy developed in this section we write any powers such that it can be computed as a product of powers obtained with repeated squaring.

In Section 11.2 on binary numbers, we saw that every natural number can be written as a sum of powers of 2. By writing the exponent as a sum of powers of two, we can compute the value as a product of other values whose exponent is a power of 2. These are the powers we can compute efficiently with repeated squaring.

Problem 15.3.1. *Let b be an integer. How can we compute b^{13} if we know b , b^2 , b^4 and b^8 ?*

Solution. We can write b^{13} as a product of the powers of b that we know.

$$b^{13} = \underbrace{b \cdot b}_{b^2} \cdot \underbrace{b \cdot b}_{b^2} \cdot \underbrace{b \cdot b}_{b^2} \cdot \underbrace{b \cdot b}_{b^2} \cdot \underbrace{b \cdot b}_{b^2} \cdot \underbrace{b \cdot b}_{b^2} \cdot b$$

$$\underbrace{\hspace{1.5cm}}_{b^4} \quad \underbrace{\hspace{1.5cm}}_{b^4} \quad \underbrace{\hspace{1.5cm}}_{b^4}$$

$$\underbrace{\hspace{3.5cm}}_{b^8}$$

So we can write b^{13} as $b^8 \cdot b^4 \cdot b$. This can also be seen by writing the exponent 13 is equal to $8 + 4 + 1$.

Example 15.3.2. We compute 3^{13} using only 5 multiplications.

- (i) The first multiplication gives us $3^2 = 3 \cdot 3 = 9$.
- (ii) With the second multiplication we compute $3^4 = 3^2 \cdot 3^2 = 9 \cdot 9 = 81$.
- (iii) With the third multiplication we compute $3^8 = 3^4 \cdot 3^4 = 81 \cdot 81 = 6561$.
- (iv) The fourth and fifth multiplication yield the desired result

$$3^{13} = 3 \cdot 3^4 \cdot 3^8 = 3 \cdot 81 \cdot 6561 = 243 \cdot 6561 = 1594323.$$

By squaring the result each time, we can efficiently compute the result when the exponents that are powers of two (2, 4, 8, 16, ...). These numbers are exactly the place values of the base 2 (or binary) representation of integers. So writing an exponent as a sum of powers of two is the same as writing a number in base 2. To minimize the number of multiplications, we will always use the highest powers of two possible.

Now we look at example of computing powers in a group first using repeated multiplication and then using the method where we first write our exponent as a sum of powers of two, as in Example 15.3.2.

Example 15.3.3. In the group $(\mathbb{Z}_{29}^{\otimes}, \otimes)$ we compute $3^{18\otimes}$ in two different ways.

- (i) We use the naive exponentiation algorithm (Algorithm 15.1.5). The computations will be easier than in the case of integers, because we compute modulo 29. We obtain:

$3^{1\otimes} = 3$	$3^{2\otimes} = 3 \otimes 3 = 9$
$3^{3\otimes} = 9 \otimes 3 = 27$	$3^{4\otimes} = 27 \otimes 3 = 81 \bmod 29 = 23$
$3^{5\otimes} = 23 \otimes 3 = 69 \bmod 29 = 11$	$3^{6\otimes} = 11 \otimes 3 = 33 \bmod 29 = 4$
$3^{7\otimes} = 4 \otimes 3 = 12$	$3^{8\otimes} = 12 \otimes 3 = 36 \bmod 29 = 7$
$3^{9\otimes} = 7 \otimes 3 = 21$	$3^{10\otimes} = 21 \otimes 3 = 63 \bmod 29 = 5$
$3^{11\otimes} = 5 \otimes 3 = 15$	$3^{12\otimes} = 15 \otimes 3 = 45 \bmod 29 = 16$
$3^{13\otimes} = 16 \otimes 3 = 48 \bmod 29 = 19$	$3^{14\otimes} = 19 \otimes 3 = 57 \bmod 29 = 28$
$3^{15\otimes} = 28 \otimes 3 = 84 \bmod 29 = 26$	$3^{16\otimes} = 26 \otimes 3 = 78 \bmod 29 = 20$
$3^{17\otimes} = 20 \otimes 3 = 60 \bmod 29 = 2$	$3^{18\otimes} = 2 \otimes 3 = 6$

Thus in $(\mathbb{Z}_{29}^{\otimes}, \otimes)$ we have $3^{18\star} = 6$.

- (ii) We present a faster method. The exponent 18 written as a sum of powers of 2 is $18 = 16 + 2 = 2^4 + 2$. We obtain this either by educated guesses or by considering the base 2 representation

$$18 = 10010_2 = 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2 + 0 \cdot 1.$$

With the rules of exponentiation we get

$$3^{18\otimes} = 3^{(2+16)\otimes} = 3^{2\otimes} \otimes 3^{16\otimes}.$$

So it is sufficient to find $3^{2\otimes}$ and $3^{16\otimes}$ and multiply them to compute $3^{18\otimes}$. We compute the highest power of these, namely $3^{16\otimes}$, by repeated squaring. The power $3^{2\otimes}$ is also computed in this process.

$$\begin{aligned} 3^{2\otimes} &= 3 \otimes 3 = 9 \\ 3^{4\otimes} &= (3^{2\otimes})^{2\otimes} = 9 \otimes 9 = 81 \bmod 29 = 23 \\ 3^{8\otimes} &= (3^{4\otimes})^{2\otimes} = 23 \otimes 23 = 529 \bmod 29 = 7 \\ 3^{16\otimes} &= (3^{8\otimes})^{2\otimes} = 7 \otimes 7 = 49 \bmod 29 = 20. \end{aligned}$$

Now we compute $3^{18\otimes} = 3^{2+16\otimes} = 3^{2\otimes} \otimes 3^{16\otimes} = 9 \otimes 20 = 180 \bmod 29 = 6$.

In (i) we computed $3^{18\otimes}$ with 17 group operations $\otimes$, while in (ii) we needed 5 group operations $\otimes$. So the method in (ii) is considerably faster than the method we used in (i).

We call the method used in 15.3.3(ii) *fast exponentiation*.

Problem 15.3.4. In the group $(\mathbb{Z}_{101}^{\otimes}, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 11$ compute $7^{66\otimes}$.

Solution. First compute the base 2 expansion of 66 and obtain

$$66 = (1 \cdot 2^6) + (0 \cdot 2^5) + (0 \cdot 2^4) + (0 \cdot 2^3) + (0 \cdot 2^2) + (1 \cdot 2^1) + (0 \cdot 2^0).$$

So the powers of 7 that we need are $7^{2^6\otimes} = 7^{64\otimes}$ and $7^{2^1\otimes} = 7^{2\otimes}$. Repeated squaring yields these powers of 7:

$$\begin{aligned} 7^{2\otimes} &= 7 \otimes 7 = 49 \bmod 101 = 49 \\ 7^{4\otimes} &= 7^{2\otimes} \otimes 7^{2\otimes} = 49 \otimes 49 = 2401 \bmod 101 = 78 \\ 7^{8\otimes} &= 7^{4\otimes} \otimes 7^{4\otimes} = 78 \otimes 78 = 6084 \bmod 101 = 24 \\ 7^{16\otimes} &= 7^{8\otimes} \otimes 7^{8\otimes} = 24 \otimes 24 = 576 \bmod 101 = 71 \\ 7^{32\otimes} &= 7^{16\otimes} \otimes 7^{16\otimes} = 71 \otimes 71 = 5041 \bmod 101 = 92 \\ 7^{64\otimes} &= 7^{32\otimes} \otimes 7^{32\otimes} = 92 \otimes 92 = 8464 \bmod 101 = 81 \end{aligned}$$

Multiplying the powers of 7 whose exponents occur in the base 2 expansion of $66 = 64 + 2$ we obtain

$$7^{66\otimes} = 7^{64\otimes} \otimes 7^{2\otimes} = 81 \otimes 49 = 3969 \bmod 101 = 30.$$

Problem 15.3.5. In the group $(\mathbb{Z}_{47}^{\otimes}, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 47$ we have

$$3^{1\otimes} = 3, 3^{2\otimes} = 9, 3^{4\otimes} = 34, 3^{8\otimes} = 28, 3^{16\otimes} = 32, 3^{32\otimes} = 37$$

With this information compute $3^{40\otimes}$ by fast exponentiation.

Solution. First we determine which powers of 3 we need to compute $3^{40\otimes}$. The base 2 expansion of 40 is

$$30 = (1 \cdot 2^5) + (0 \cdot 2^4) + (1 \cdot 2^3) + (0 \cdot 2^2) + (0 \cdot 2^1) + (0 \cdot 2^0).$$

Thus the powers of 3 that we need are

$$3^{2^5\otimes} = 3^{32\otimes} \text{ and } 3^{2^3\otimes} = 3^{8\otimes}$$

With the powers of 3 given in the problem we obtain

$$3^{40\otimes} = 3^{32\otimes} \otimes 3^{8\otimes} = 37 \otimes 28 = 1036 \bmod 47 = 2.$$

Thus in $(\mathbb{Z}_{47}^\otimes, \otimes)$ we have $3^{40\otimes} = 8$.

We formulate the fast exponentiation strategy as an algorithm. Instead of first going through the repeated squaring and then multiplying the needed powers we combine the two steps in one loop. In this loop we square and at the same time compute whether or not that power of two is used in the exponent as a sum of powers of two.

Algorithm 15.3.6 (*Fast Exponentiation*).

Input: A group $(G, \star)$, $b \in G$, and $n \in \mathbb{N}$

Output: $a = b^{n\star}$

- (1) **let** $a := 1$
- (2) **let** $c := b$
- (3) **repeat**
 - (a) **let** $r := n \bmod 2$
 - (b) **if** $r = 1$ **then let** $a := a \star c$
 - (c) **let** $n := n \operatorname{div} 2$
 - (d) **let** $c := c \star c$
- (4) **until** $n = 0$
- (5) **return** a

Example 15.3.7. With Algorithm 15.3.6 we compute $4^{25\otimes}$ in the group $(\mathbb{Z}_{53}^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 53$.

Initially we have $b = 5$ and $n = 25$ and set $a := 1$ and $c := 4$. In the iterations of the loop the variables have the following values. In each row of the table we give the values of r , a , c , and n at the end of step (b).

r	a	c	n
	1	4	25
let $r := n \bmod 2$	if $r = 1$ then let $a := a \otimes c$	let $c := c \otimes c$	let $n := n \operatorname{div} 2$
$25 \bmod 2 = 1$	As $r = 1$ we set a to $1 \otimes 4 = 4$	$4 \otimes 4 = 16$	$25 \operatorname{div} 2 = 12$
$12 \bmod 2 = 0$	As $r = 0$ the value of a stays 4	$16 \otimes 16 = 44$	$12 \operatorname{div} 2 = 6$
$6 \bmod 2 = 0$	As $r = 0$ the value of a stays 4	$44 \otimes 44 = 28$	$6 \operatorname{div} 2 = 3$
$3 \bmod 2 = 1$	As $r = 1$ we set a to $4 \otimes 28 = 6$	$28 \otimes 28 = 42$	$3 \operatorname{div} 2 = 1$
$1 \bmod 2 = 1$	As $r = 1$ we set a to $6 \otimes 42 = 40$		$1 \operatorname{div} 2 = 0$

So we get $4^{25\otimes} = 4^{25} \bmod 53 = 40$.

Problem 15.3.8. In the group $(\mathbb{Z}_{101}^\otimes, \otimes)$ where $\otimes : \mathbb{Z}_{101}^\otimes \times \mathbb{Z}_{101}^\otimes \rightarrow \mathbb{Z}_{101}^\otimes$ is defined by $a \otimes b = (a \cdot b) \bmod 101$ find $2^{24\otimes}$ using the fast exponentiation method. Count the number of group operations needed.

Solution. We apply the fast exponentiation method. As $24 = 8 + 16 = 2^3 + 2^4$ the highest power of the group element 2 that we need is $2^{16\otimes}$. We compute:

$$\begin{aligned} 2^{2\otimes} &= 2 \otimes 2 = 4 & 2^{4\otimes} &= 2^{2\otimes} \otimes 2^{2\otimes} = 4 \otimes 4 = 16 \\ 2^{8\otimes} &= 2^{4\otimes} \otimes 2^{4\otimes} = 16 \otimes 16 = 54 & 2^{16\otimes} &= 2^{8\otimes} \otimes 2^{8\otimes} = 54 \otimes 54 = 88 \end{aligned}$$

Thus $2^{24\otimes} = 2^{(8+16)\otimes} = 2^{8\otimes} \otimes 2^{16\otimes} = 54 \otimes 88 = 5$. We found the solution with 5 group operations $\otimes$.

15.4 Discrete Logarithm

We now consider the discrete logarithm to the base b which is the inverse of exponentiation with base b .

We first investigate in a concrete example which elements of a group are powers of the group elements.

Example 15.4.1. In $(\mathbb{Z}_7^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 7$ we investigate the powers of all elements. Recall that $\mathbb{Z}_7^\otimes = \{1, 2, 3, 4, 5, 6\}$ and $\mathbb{W} = \{0, 1, 2, \dots\}$.

powers of 1: $1^{0\otimes} = 1, 1^{1\otimes} = 1, 1^{2\otimes} = 1$; as we obtain the n -th power of 1 multiplying n copies of 1 we have for all $n \in \mathbb{W}$ that $1^{n\otimes} = 1$.

powers of 2: $2^{0\otimes} = 1, 2^{1\otimes} = 2, 2^{2\otimes} = 4, 2^{3\otimes} = 1, 2^{4\otimes} = 2$; when we continue multiplying by 2 we cycle through 1, 2, and 4, see Figure 15.4.1 (b).

powers of 3: $3^{0\otimes} = 1, 3^{1\otimes} = 3, 3^{2\otimes} = 2, 3^{3\otimes} = 6, 3^{4\otimes} = 4, 3^{5\otimes} = 5, 3^{6\otimes} = 1$; so all elements of $\mathbb{Z}_7^\otimes$ are powers of 3, see Figure 15.4.1 (c).

powers of 4: $4^{0\otimes} = 1, 4^{1\otimes} = 4, 4^{2\otimes} = 2, 4^{3\otimes} = 1, 4^{4\otimes} = 4$; when we continue multiplying by 4 we cycle through 1, 4, and 2.

powers of 5: $5^{0\otimes} = 1, 5^{1\otimes} = 5, 5^{2\otimes} = 4, 5^{3\otimes} = 6, 5^{4\otimes} = 2, 5^{5\otimes} = 3, 5^{6\otimes} = 1$; so all elements of $\mathbb{Z}_7^\otimes$ are powers of 5.

powers of 6: $6^{0\otimes} = 1$ and $6^{1\otimes} = 6$; all other powers of 6 are 1 or 6, see Figure 15.4.1 (a).

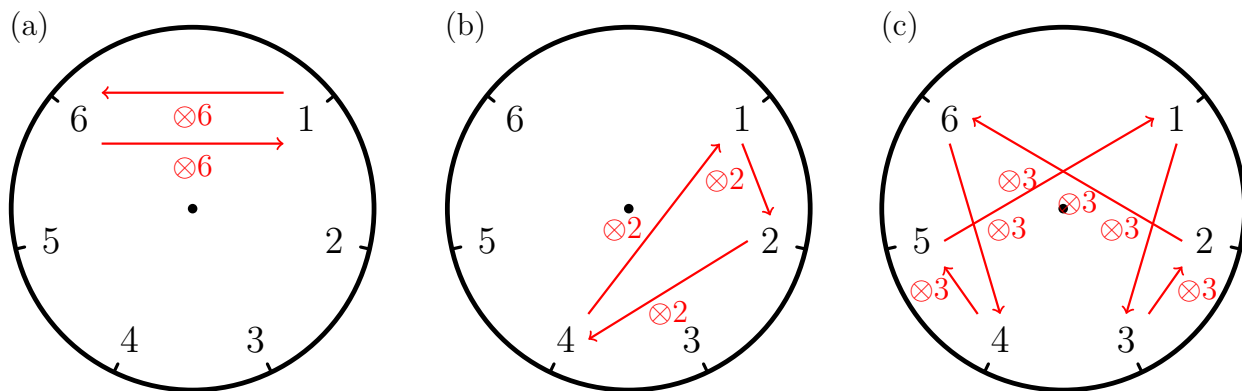
Let $(G, \star)$ be a group. For two a and b in G the discrete logarithm of a to base b is the answer to the following question. For which $n \in \mathbb{W}$ do we have:

$$b^{n\star} = a?$$

Before we introduce a notation for the answer to this question, we look back at Example 15.4.1 and see that the answer does not always exist.

Figure 15.4.1: Powers of elements in the group $(\mathbb{Z}_7^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 7$.

- (a) The powers of 6, namely $6^{0\otimes} = 1$ and $6^{1\otimes} = 6$
- (b) The powers of 2, namely $2^{0\otimes} = 1$, $2^{1\otimes} = 2$, and $2^{2\otimes} = 4$
- (c) The powers of 3, namely $3^{0\otimes} = 1$, $3^{1\otimes} = 3$, $3^{2\otimes} = 2$, $3^{3\otimes} = 6$, $3^{4\otimes} = 4$, $3^{5\otimes} = 5$



Example 15.4.2. In $(\mathbb{Z}_7^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 7$ there is no $n \in \mathbb{W}$ such that $2^{n\otimes} = 3$, because the only powers of 2 in $(\mathbb{Z}_7^\otimes, \otimes)$ are 1, 2, and 4 (compare Example 15.4.1 powers of 2).

Definition 15.4.3. Let $(G, \star)$ be a group and let $b \in G$ and $a \in G$. The *discrete logarithm* of a to base b with respect to $\star$ is the smallest non-negative integer n such that $b^{n\star} = a$. If such an n does not exist we say that the discrete logarithm does not exist.

We denote the discrete logarithm of a to base b with respect to $\star$ by $\log_b^\star a$.

Example 15.4.4. In the group $(\mathbb{Z}_5^\otimes)$ where $a \otimes b := (a \cdot b) \bmod 5$ we have:

- (i) $\log_2^\otimes 1 = 0$ because $2^{0\otimes} = 1$.
- (ii) $\log_2^\otimes 2 = 1$ because $2^{1\otimes} = 2$.
- (iii) $\log_2^\otimes 3 = 3$ because $2^{3\otimes} = (2^3) \bmod 5 = 8 \bmod 5 = 3$.

To find discrete logarithms we often have to try out several possible answers. Sometimes we cannot find an answer and we conclude that the discrete logarithm does not exist.

Problem 15.4.5. In the group $(\mathbb{Z}_5^\otimes)$ where $a \otimes b := (a \cdot b) \bmod 5$ find the following discrete logarithms provided they exist.

- (i) $\log_3^\otimes 2$
- (ii) $\log_4^\otimes 2$

Solution. (i) We try out powers of 3 until we obtain 2.

$$\begin{aligned}
 3^{0\otimes} &= 1 \\
 3^{1\otimes} &= 3 \bmod 5 = 3 \\
 3^{2\otimes} &= 3 \otimes 3 = (3 \cdot 3) \bmod 5 = 9 \bmod 5 = 4 \\
 3^{3\otimes} &= 3^{2\otimes} \otimes 3 = 4 \otimes 3 = (4 \cdot 3) \bmod 5 = 12 \bmod 5 = 2
 \end{aligned}$$

Thus $\log_3^\otimes 2 = 3$.

(ii) We try out powers of 4 until we obtain 2.

$$\begin{aligned} 4^{0\otimes} &= 1 \\ 4^{1\otimes} &= 4 \bmod 5 = 4 \\ 4^{2\otimes} &= 4 \otimes 4 = (4 \cdot 4) \bmod 5 = 16 \bmod 5 = 1 \\ 4^{3\otimes} &= 4^{2\otimes} \otimes 4 = (1 \cdot 4) \bmod 5 = 4 \bmod 5 = 4 \end{aligned}$$

Continuing this we get $4^{4\otimes} = 1$, $4^{5\otimes} = 4$, $4^{6\otimes} = 1$. As $4 \otimes 4 = 1$ and $1 \otimes 4 = 4$ further multiplication by 4 only yields 1 or 4. So the only numbers that can be written as powers of 4 in $(\mathbb{Z}_5^\otimes)$ are 1 and 4. This means that there is no non-negative integer n such that $4^{n\otimes} = 2$. We have found that $\log_4^\otimes 2$ does not exist.

The following follows from the definition of exponentiation and discrete logarithm.

Theorem 15.4.6. *Let $(G, \star)$ be a group and let $a \in G$ and $b \in G$. We have*

- (i) $\log_b^\star 1 = 0$ because $b^{0\star} = 1$.
- (ii) $\log_b^\star b = 1$ because $b^{1\star} = b$.

Example 15.4.7. We give powers of elements and the corresponding discrete logarithm to base 5 for the elements of the group $(\mathbb{Z}_7^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 7$. We see that all elements of $\mathbb{Z}_7^\otimes$ can be written as powers of 5.

- (i) We have $5^{0\otimes} = 1$. Thus $\log_5^\otimes 1 = 0$.
- (ii) We have $5^{1\otimes} = 5$. Thus $\log_5^\otimes 5 = 1$.
- (iii) We have $5^{2\otimes} = 5 \otimes 5 = (5 \cdot 5) \bmod 7 = 4$. Thus $\log_5^\otimes 4 = 2$.
- (iv) We have $5^{3\otimes} = 4 \otimes 5 = (4 \cdot 5) \bmod 7 = 6$. Thus $\log_5^\otimes 6 = 3$.
- (v) We have $5^{4\otimes} = 6 \otimes 5 = (6 \cdot 5) \bmod 7 = 2$. Thus $\log_5^\otimes 2 = 4$.
- (vi) We have $5^{5\otimes} = 2 \otimes 5 = (2 \cdot 5) \bmod 7 = 3$. Thus $\log_5^\otimes 3 = 5$.

Exponentiation and discrete logarithm to the same base are inverse functions. This is illustrated in the next example.

Example 15.4.8. In the group $(\mathbb{Z}_5^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 5$ we consider exponentiation and logarithm with base 3. Let the function $e : \mathbb{Z}_4 \rightarrow \mathbb{Z}_5^\otimes$ be given by $e(x) = 3^{x\otimes}$. The function e is the exponentiation function with base 3. We have

$$e(0) = 3^{0\otimes} = 1, e(1) = 3^{1\otimes} = 3, e(2) = 3^{2\otimes} = 4, e(3) = 3^{3\otimes} = 2$$

The discrete logarithm $\log_3^\otimes y$ of $y \in \mathbb{Z}_5^\otimes$ to base 3 is the smallest non-negative integer n such that $3^{n\otimes} = y$. Let the function given by $l : \mathbb{Z}_5^\otimes \rightarrow \mathbb{Z}_4$ be given by

$$l(1) = \log_3^\otimes 1 = 0, l(2) = \log_3^\otimes 2 = 3, l(3) = \log_3^\otimes 3 = 1, l(4) = \log_3^\otimes 4 = 2$$

As $l(e(x)) = x$ for all $x \in \mathbb{Z}_4$, the function l is the inverse function of e .

Depending on the group, the effort of finding discrete logarithms varies considerably. Different approaches can be used to find discrete logarithms. For small groups, we can produce a table where we can quickly look up the values.

Problem 15.4.9. *In the group $(\mathbb{Z}_7^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 7$ find the discrete logarithm to base 3 of 6.*

Solution. We need to find $n \in \mathbb{W}$ such that $3^{n\otimes} = 6$. From Figure 15.4.1(c) we see that $3^{3\otimes} = 3 \otimes 3 \otimes 3 = 6$. Thus $\log_3^\otimes 6 = 3$.

In general we try out exponents until we find the right one. We never have to try out more exponents than our group has elements, so we know when to stop in case the discrete logarithm does not exist.

Problem 15.4.10. *In the group $(\mathbb{Z}_{11}^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 11$ find $\log_7^\otimes 9$.*

Solution. We need to find $n \in \mathbb{W}$ such that $7^{n\otimes} = 9$. We compute

$$\begin{array}{ll} 7^{1\otimes} = 7 & 7^{2\otimes} = (7 \cdot 7) \bmod 11 = 5 \\ 7^{3\otimes} = 5 \otimes 7 = (5 \cdot 7) \bmod 11 = 2 & 7^{4\otimes} = (2 \otimes 7) = (2 \cdot 7) \bmod 11 = 3 \\ 7^{5\otimes} = 3 \otimes 7 = (3 \cdot 7) \bmod 11 = 10 & 7^{6\otimes} = (10 \otimes 7) = (10 \cdot 7) \bmod 11 = 4 \\ 7^{7\otimes} = 4 \otimes 7 = (4 \cdot 7) \bmod 11 = 6 & 7^{8\otimes} = (6 \otimes 7) = (6 \cdot 7) \bmod 11 = 9 \end{array}$$

Thus $\log_7^\otimes 9 = 8$.

The method that we applied to find the discrete logarithm is called the naive method. We formulate it as an algorithm. To assure that our algorithm terminates we assume that our group is finite. When the group is finite, the number of possible distinct powers of any element of the group is at most the number of elements in the group. We make this the termination criterion for the loop in our algorithm.

Algorithm 15.4.11 (*Naive Discrete Logarithm*).

Input: A finite group $(G, \star)$, $b \in G$, $b \neq 0$ and $a \in G$, $a \neq 0$

Output: $\log_b^\star a$, that is, $n \in \mathbb{N}$ such that $b^{n\star} = a$

- (1) **let** $n := 0$
- (2) **let** $c := 1$
- (3) **repeat**
 - (a) **if** $c = a$ **then return** n
 - (b) **let** $c := c \star b$
 - (c) **let** $n := n + 1$
- (4) **until** $n = \#G$
- (5) **return** “ $\log_b^\star a$ does not exist.”

Next we illustrate with an example that computing powers using fast exponentiation is considerably faster than finding discrete logarithms with the naive method.

Problem 15.4.12. In the group $(\mathbb{Z}_{101}^{\otimes}, \otimes)$ where $\otimes : \mathbb{Z}_{101}^{\otimes} \times \mathbb{Z}_{101}^{\otimes} \rightarrow \mathbb{Z}_{101}^{\otimes}$ is defined by $a \otimes b = (a \cdot b) \bmod 101$.

- (i) Find $\log_2^{\otimes} 5$.
- (ii) Count the number of group operation $\otimes$ you need to find $\log_2^{\otimes} 5$.
- (iii) How many group operations $\otimes$ are needed to compute $2^{24\otimes}$ using fast exponentiation.

Solution. (i) We check all powers of 2 until we obtain 5. We get:

$$\begin{array}{llll}
 2^{1\otimes} = 2 & 2^{2\otimes} = 2 \otimes 2 = 4 & 2^{3\otimes} = 4 \otimes 2 = 8 & 2^{4\otimes} = 8 \otimes 2 = 16 \\
 2^{5\otimes} = 16 \otimes 2 = 32 & 2^{6\otimes} = 32 \otimes 2 = 64 & 2^{7\otimes} = 64 \otimes 2 = 27 & 2^{8\otimes} = 27 \otimes 2 = 54 \\
 2^{9\otimes} = 54 \otimes 2 = 7 & 2^{10\otimes} = 7 \otimes 2 = 14 & 2^{11\otimes} = 14 \otimes 2 = 28 & 2^{12\otimes} = 18 \otimes 2 = 56 \\
 2^{13\otimes} = 56 \otimes 2 = 11 & 2^{14\otimes} = 11 \otimes 2 = 22 & 2^{15\otimes} = 22 \otimes 2 = 44 & 2^{16\otimes} = 44 \otimes 2 = 88 \\
 2^{17\otimes} = 88 \otimes 2 = 75 & 2^{18\otimes} = 75 \otimes 2 = 49 & 2^{19\otimes} = 49 \otimes 2 = 98 & 2^{20\otimes} = 98 \otimes 2 = 95 \\
 2^{21\otimes} = 95 \otimes 2 = 89 & 2^{22\otimes} = 89 \otimes 2 = 77 & 2^{23\otimes} = 77 \otimes 2 = 53 & 2^{24\otimes} = 53 \otimes 2 = 5
 \end{array}$$

Thus the discrete logarithm to base 2 of 5 is 24.

- (ii) We found the solution with 23 group operations $\otimes$.
- (iii) To compute $2^{24\otimes}$ using fast exponentiation we first write 24 as a sum of powers of 2. We find $24 = 16 + 8 = 2^4 + 2^3$ and compute

$$2^{2\otimes} = 2 \otimes 2 = 4, \quad 2^{4\otimes} = 4 \otimes 4 = 16, \quad 2^{8\otimes} = 16 \otimes 16 = 54, \quad 2^{16\otimes} = 54 \otimes 54 = 88.$$

These are 4 group operations $\otimes$. Now one more group operation $\otimes$ yields

$$2^{24\otimes} = 2^{16\otimes} \otimes 2^{8\otimes} = 88 \otimes 54 = 5.$$

So we need 5 group operations $\otimes$ to compute $2^{24\otimes}$.

The previous problem illustrates that more group operations are needed to find the discrete logarithm than for computing the corresponding power with the fast exponentiation algorithm (Algorithm 15.3.6). In Problem 15.3.8, we computed $2^{28\otimes}$ in 5 group operations $\otimes$ while it took 23 group operations $\otimes$ to find $\log_2^{\otimes} 5$. In general, computing discrete logarithms in the group $(\mathbb{Z}_p^{\otimes}, \otimes)$ is difficult.

There are methods for computing discrete logarithms in the group $(\mathbb{Z}_p^{\otimes}, \otimes)$ that are faster than checking all powers of the generator. Some of the methods are the Baby Step Giant Step algorithm, index calculus algorithm, and the number field sieve, all of which are outside the scope of this course. Even with these algorithms computing discrete logarithm is much slower than exponentiation, see table 15.4.2.

Since the discrete logarithm is much harder to compute than exponentiation, in the next section we will present two public key crypto systems whose security depends on the fact that powers are faster to compute than discrete logarithms.

Figure 15.4.2: Comparison of the number of operations in $(\mathbb{Z}_p^\otimes, \otimes)$ for exponentiation and computing discrete logarithms. The numbers in the table are the numbers of decimal digits of the number of operations. For example computing the discrete logarithm in $\mathbb{Z}_p$ where p has 309 digits with the number field sieve needs approximately 10^{43} group operations while fast exponentiation in the same group needs only 10^{10} group operations.

p	expected number of operations in $\mathbb{Z}_p^\otimes$ for				
	exponentiation		discrete logarithm		
	naive exp.	fast exp.	naive log.	baby step giant step	number field sieve
in number of decimal digits					
20	20	5	20	10	8
39	39	6	39	20	12
78	78	7	78	39	19
155	155	8	155	78	28
309	309	9	309	155	43
617	617	10	617	309	63
1234	1234	11	1234	617	94
2467	2467	12	2467	1234	139
4933	4933	13	4933	2467	205

Chapter 16

Public Key Cryptography

Student Learning Outcomes

Upon completion of the work on this section, students will be able to

- (1) Distinguish between symmetric key and public key crypto systems.
- (2) Apply the Diffie-Hellman key exchange to exchange secrets.
- (3) Apply the ElGamal cryptosystem to the encryption and decryption of text.

We bring concepts from all chapters of the course together in the presentation of public key cryptosystems. These concepts are

Chapter 1 Integers and Algorithm: The operation mod , the Euclidean algorithm, and Bézout's identity for finding inverse in the group $(\mathbb{Z}_p^\otimes, \otimes)$.

Chapter 2 Sets and Functions: The encoding function C for converting text into a sequence of numbers.

Chapter 3 Numbers and Counting: Prime numbers, binary numbers needed for fast exponentiation, and the representation of text by numbers.

Chapter 4 Groups: The groups $(\mathbb{Z}_p^\otimes, \otimes)$, (fast) exponentiation, and the discrete logarithm needed to attack the cryptosystems.

In particular we present the Diffie-Hellman key exchange and the ElGamal crypto system, which are both widely used in practice. The Diffie-Hellman key exchange is used to initiate secure connections such as the secure communication between web browser and web server. ElGamal is applied in the encryption of email and other forms a secure communication.

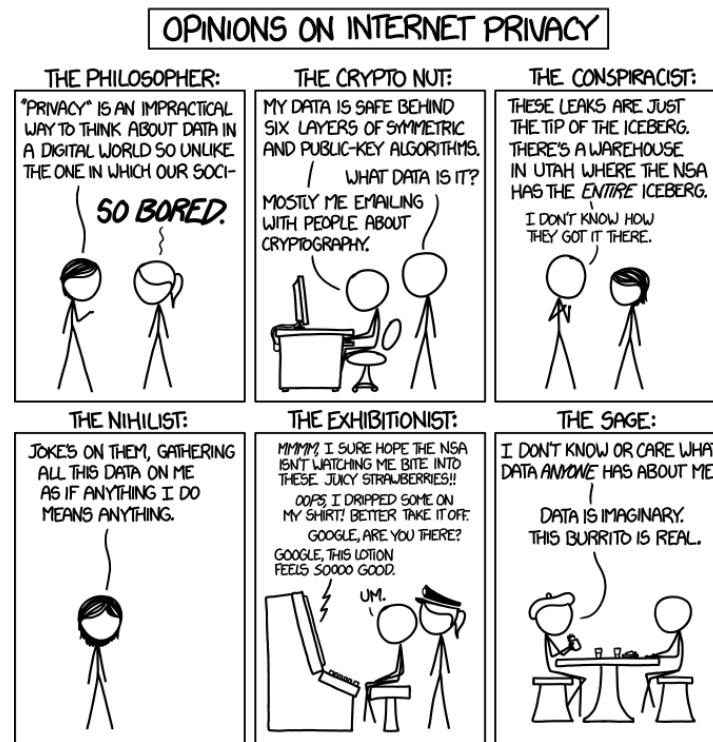
16.1 Introduction

Symmetric-key cryptosystems, like the Caesar cipher in Section 8, use the same key for encryption and decryption of a message. So, both parties need to share a key to be able to encrypt and decrypt messages. A significant disadvantage is that the key has to be distributed through secure channels.

In public-key cryptosystems each person has two keys. The *public key* is used for encryption and may be freely distributed. The corresponding *private key* is used for decryption and must remain secret.

Public-key cryptography is widely used for all secure digital communication over the Internet, such as secure web sites and online banking. It is used to secure your privacy and your finances. Public-key cryptography is realized by using certain mathematical functions called *trapdoor functions*, that can be evaluated quickly but cannot be inverted in a reasonable amount of time. If one knows the secret, the function can be inverted efficiently.

Figure 16.0.1: *Privacy Opinions* by R. Munroe (<https://xkcd.com/1269>).



I'm the Philosopher until someone hands me a burrito.

Definition 16.1.1 (Trapdoor Function). An invertible function E is called a *trapdoor function* if the inverse of E can only be evaluated efficiently when in possession of some additional information.

Example 16.1.2. Examples for functions that are easy to evaluate but whose inverse is difficult to evaluate are:

- (i) Multiplication of integers is much easier than factorization of integers.
- (ii) Exponentiation in the groups $(\mathbb{Z}_p^\otimes, \otimes)$ is much easier than finding discrete logarithms (compare Figure 15.4.2).

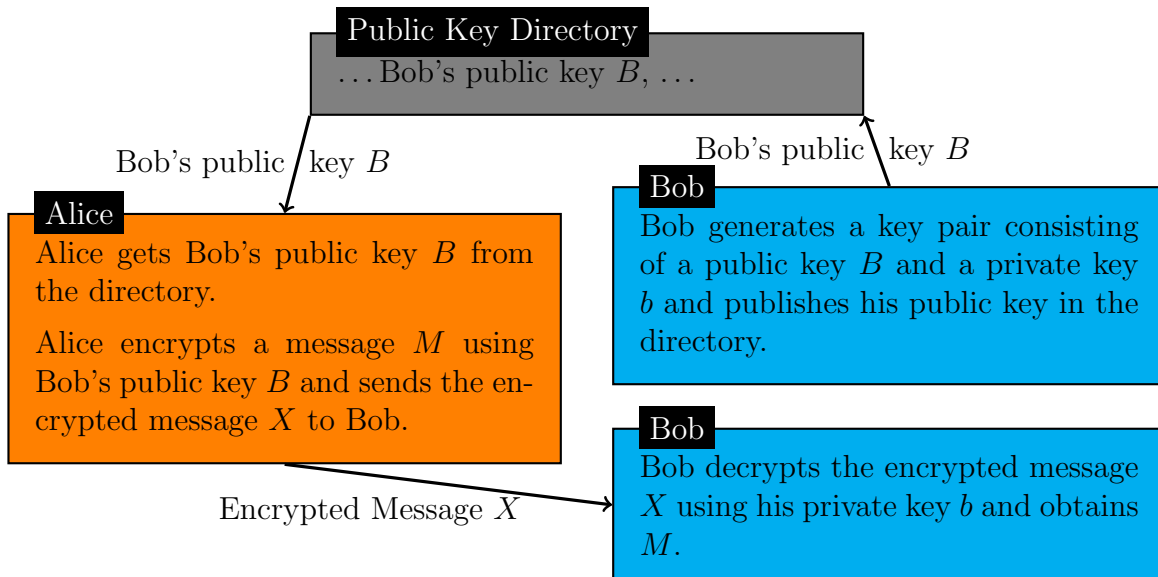
The RSA cryptosystem¹² is based on the difficulty of factorization. In 16.3 and 16.2 we describe cryptosystems that rely on the difficulty of efficiently computing discrete logarithms, namely the Diffie-Hellman key exchange and the ElGamal public key cryptosystem.

See Figure 16.1.1 for the general steps of a public-key cryptosystem.

¹rsa.

²Clifford Cocks described an equivalent system in 1973, but it was classified by the UK intelligence agency GCHQ until 1997

Figure 16.1.1: Public key cryptography. A message encrypted with Bob’s public key B can only be decrypted (in reasonable time) when in possession of his private key b .



16.1.1 Padlock Analogue

To compare symmetric key and public key cryptography we use the mechanical analogue of a padlock. Let’s assume that secure messages are sent in a box that can be locked with a padlock.

In symmetric key cryptography both Alice and Bob have a key to the same padlock. To send a secure message to Bob, Alice places the message into a box and locks it with the padlock using her key. Bob receives the box, opens it with his key, and reads the message.

In public key cryptography Bob makes his own padlocks to which only he can open, and distributes them to everyone who wants to send him secure messages. Alice writes her message, puts it in a Box, and locks it with one of Bob’s padlocks. She sends it to Bob, and only Bob, who has the key to unlock the padlock, open the box to read the message.

16.1.2 Digital Signatures

A digital signature is a mathematical scheme for demonstrating the authenticity of a digital message. A valid digital signature gives a recipient reason to believe that the message was created by a known sender.

Assume that Bob has generated a public key and a private key and published his public key. Now he can sign messages and others can verify his signature.

- (i) Bob encrypts a message with his private key.

Figure 16.1.2: The first paragraph of the article *New Directions in Cryptography* by Whitfield Diffie and Martin Hellman

I. INTRODUCTION

WE STAND TODAY on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.

- (ii) Bob sends the message to Alice.
- (iii) Alice obtains Bob's public key form the key directory.
- (iv) Alice decrypts Bob's message using Bob's public key. Thereby confirming its authenticity.

In the next section we present realizations of the ideas described above. We start with the Diffie-Hellman key exchange and continue with he ElGamal public key cryptosystem that is based on the Diffie-Hellman key exchange.

16.2 Diffie-Hellman key exchange

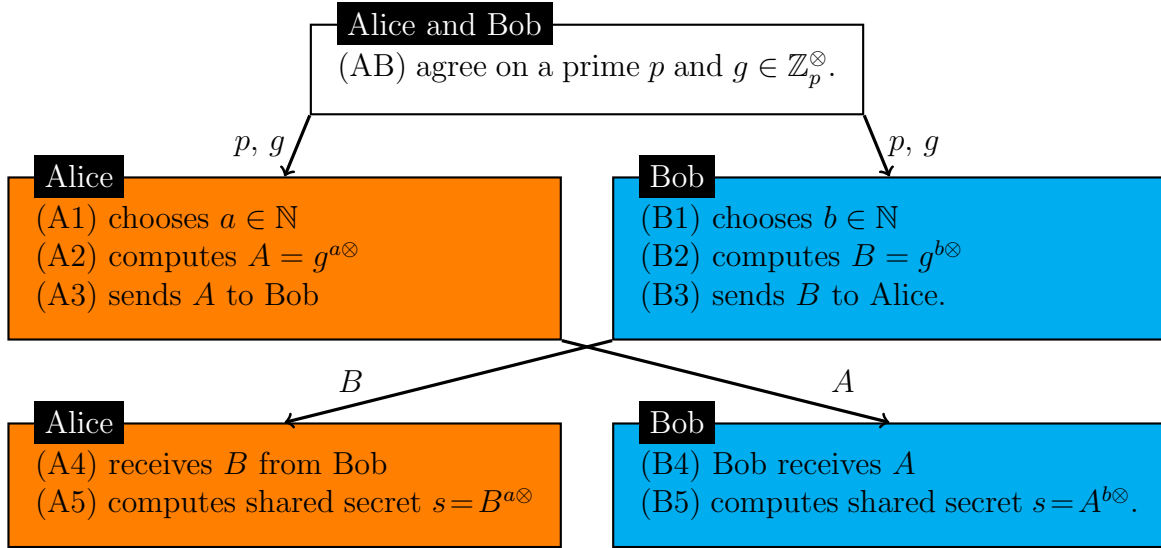
To initiate secure communication it is sufficient to determine a shared secret in the form of a cryptographic key. This key can be used for communication using a symmetric cryptographic protocol, such as AES, which requires less resources than communicating using a public key protocol. The *Diffie-Hellman key exchange* is a cryptographic protocol for exchanging cryptographic keys over a public channel. It was proposed by Ralph Merkle³ and is named after Whitfield Diffie and Martin Hellman⁴.

If there is no doubt about the identity of the other party, the Diffie-Hellman key exchange does not need any additional infrastructure, such as a key directory.

³merkle.

⁴diffiehellman.

Figure 16.2.1: Diffie-Hellman key exchange. The agreement on p and g takes place over an insecure channel. Alice and Bob generate a shared secret s without sending the secret. All computations take place in the group $(\mathbb{Z}_p^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod p$.



To create a shared secret Alice (A) and Bob (B) follow the following steps (also see Figure 16.2.1). First Alice and Bob agree on the group they want to work in. All powers are to be computed in this group.

The Group

(AB) Alice and Bob agree on a prime number p and a $g \in \mathbb{Z}_p^\otimes$. They will work in $(\mathbb{Z}_p^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod p$.

Alice: Secret

- (A1) Alice randomly chooses her secret $a \in \mathbb{N}$.
- (A2) Alice computes $A = g^{a \otimes} = (g^a) \bmod p$ to Bob.
- (A3) Alice sends A to Bob.

Bob: Secret

- (B1) Bob randomly his secret $b \in \mathbb{N}$.
- (B2) Bob computes $B = g^{b \otimes} = (g^b) \bmod p$.
- (B3) Bob sends B to Alice.

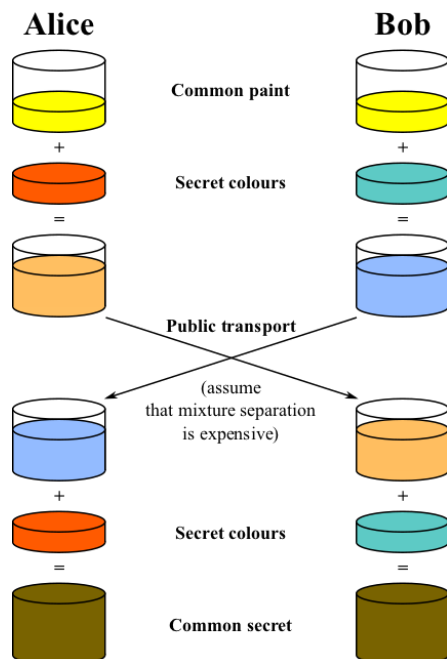
Alice: Shared secret

- (A4) Alice receives B from Bob. Alice now knows the values of p , g , a , and B .
- (A5) Alice computes the shared secret $s_A = B^{a \otimes} = (B^a) \bmod p$.

Bob: Shared secret

- (B4) Bob receives A from Alice. Bob now knows the values of p , g , b , and A .

Figure 16.2.2: The Diffie-Hellman key exchange illustrated as color mixing. From Wikimedia Commons, the free media repository, licensed under Creative Commons Attribution-Share Alike 4.0 International



(B5) Bob computes the shared secret $s_B = A^{b\otimes} = (A^b) \bmod p$.

Alice has computed the secret

$$s_A = B^{a\otimes} = (g^{b\otimes})^{a\otimes} = g^{(a\cdot b)\otimes}.$$

Bob has computed

$$s_B = A^{b\otimes} = (g^{a\otimes})^{b\otimes} = g^{(b\cdot a)\otimes}.$$

Since $a \cdot b = b \cdot a$ now Alice and Bob share the secret $s_A = s_B$.

Assume Eve has eavesdropped on the communication between Alice and Bob and now knows the p , g , A , and B . To obtain the shared secret s , Eve needs either Alice's secret a or Bob's secret b , which can only be obtained by finding the discrete logarithm of A to base g in $(\mathbb{Z}_p^\otimes, \otimes)$ or the discrete logarithm of B to base g in $(\mathbb{Z}_p^\otimes, \otimes)$. So, the security of the Diffie-Hellman key exchange depends on the difficulty of computing discrete logarithms in $(\mathbb{Z}_p^\otimes, \otimes)$.

Example 16.2.1. We give an example for a Diffie-Hellman key exchange with small numbers.

(AB) Alice and Bob agree on the prime $p = 11$ and the generator $g = 2$. They work in the group $(\mathbb{Z}_{11}^\otimes, \otimes)$ where $a \otimes b = (a \cdot b) \bmod 11$.

(A1) Alice randomly chooses $a = 8$.

(A2) Alice computes $A = 2^{a\otimes} = 2^{8\otimes} = 2^8 \bmod 11 = 256 \bmod 11 = 3$.

Figure 16.2.3: *Random Number* by R. Munroe (<https://xkcd.com/221>).

```
int getRandomNumber()
{
    return 4; // chosen by fair dice roll.
              // guaranteed to be random.
}
```

RFC 1149.5 specifies 4 as the standard IEEE-vetted random number.

- (A3) Alice sends $A = 3$ to Bob.
- (B1) Bob randomly chooses $b = 6$.
- (B2) Bob computes $B = g^{b\otimes} = 2^{6\otimes} = 2^6 \bmod 11 = 64 \bmod 11 = 9$.
- (B3) Bob sends $B = 9$ to Alice.
- (A4) Alice receives $B = 9$ from Bob.
- (A5) Alice computes the shared secret

$$s = B^{a\otimes} = 9^{8\otimes} = \left((9^{2\otimes})^{2\otimes} \right)^{2\otimes} = (4^{2\otimes})^{2\otimes} = 5^{2\otimes} = 3.$$

- (B4) Bob receives $A = 3$ from Alice.
- (B5) Bob computes the shared secret

$$s = A^{b\otimes} = 3^{6\otimes} = 3^{2\otimes} \otimes 3^{4\otimes} = 9 \otimes 9^{2\otimes} = 9 \otimes 4 = 3.$$

Now Alice and Bob share the secret $s = 3$.

From Bob's perspective a key exchange looks as follows.

Problem 16.2.2. *Alice and Bob agree to use the prime number $p = 17$ and the generator $g = 5$ for their Diffie-Hellman key exchange. Alice sends Bob $A = 2$. Bob chooses the random number $b = 3$. What is Alice and Bob's shared secret?*

Solution. The shared secret is $s = A^b \bmod p = 2^3 \bmod 17 = 8$.

We demonstrate the importance of random numbers in the Diffie-Hellman key exchange.

Problem 16.2.3. *The software company DH insecurity has implemented the random number generator from Figure 16.2.3, that is, the random numbers are always 4. Alice and Bob both use software from DH insecurity and Eve knows this. Eve eavesdrops on Bob's communication when Alice and Bob are agreeing on the prime p and the generator g . She learns that $p = 19$ and $g = 15$. So Alice and Bob are working in the subgroup of $(\mathbb{Z}_{19}, \otimes)$ generated by 15. Eve now can find Alice's and Bob's shared secret generated by the Diffie-Hellman key exchange. What is Alice and Bob's shared secret?*

Solution. As the random number generator always returns 4, Alice's secret a is 4 and Bob's secret b is 4. Thus

$$A = g^{a\otimes} = 15^{4\otimes} = (15^{2\otimes})^{2\otimes} = (15^2 \bmod 19)^{2\otimes} = (225 \bmod 19)^{2\otimes} = 16^{2\otimes} = 256 \bmod 19 = 9$$

Alice and Bob's shared secret is

$$s = A^{b\otimes} = 9^{4\otimes} = (9^{2\otimes})^{2\otimes} = (81 \bmod 19)^{2\otimes} = 5^{2\otimes} = 25 \bmod 19 = 6.$$

To give an idea how large the prime p should be for the Diffie-Hellman key exchange to be secure, we present an example for a discrete logarithm that was computed in 2014.

Example 16.2.4. Let p be the 80 decimal digit (596 digits in base 2) prime above:

$$\begin{aligned} p = & 191147927718986609689229466631454649812986246276667354864188 \\ & 503638807260703436799058776201365135161278134258296128109200 \\ & 046702912984568752800330221777752773957404540495707852046983. \end{aligned}$$

The group $(\mathbb{Z}_p^\otimes, \otimes)$ is generated by $g = 5$, that is $\mathbb{Z}_p^\otimes = \{5^n \mid n \in \mathbb{N}\}$. In 2014 the researchers Cyril Bouvier, Pierrick Gaudry, Laurent Imbert, Hamza Jeljeli, and Emmanuel Thomé announced that they computed the discrete logarithm to base 5 of

$$\begin{aligned} a = & 68188080109582330879868861330998506151774854600403700625797 \\ & 299927558995162740321112260973638619757922646242302104885437 \\ & 536745080299248852065080008358309735875192480724496530325927. \end{aligned}$$

It took them under 130 core years (that is, on a single core computer it would take that long) on a parallel computing cluster to find the solution

$$\begin{aligned} n = & 138670566126823584879625861326333326312363943825621039220215 \\ & 583346153783336272559955521970357301302912046310782908659450 \\ & 758549108092918331352215751346054755216673005939933186397777. \end{aligned}$$

They applied the data computed for finding this discrete logarithm in the computation of further discrete logarithms which only required a few hours each.

The example illustrates that a 180 decimal digits (596 digits in base 2) prime is too small for cryptographic purposes, as the discrete logarithm problem can be solved in a (relatively) short amount of time, provided enough computation power is available. The commonly recommended size of the prime for the Diffie-Hellman key exchange is 2048 base 2 digits.

16.3 ElGamal Encryption System

The *ElGamal encryption system* is a public key encryption algorithm by Taher Elgamal⁵ in 1985 that is based on the Diffie-Hellman key exchange.

We assume that the message m that Alice encrypts and sends to Bob is an integer. In Section 12 we saw how a message can be encoded into integers. We describe the three components of ElGamal encryption, namely key generation, encryption, and decryption.

Bob: Key Generation

To generate his private key and his public key Bob does the following.

⁵elgamal.

- (B1) Bob chooses a prime p and a generator $g \in \mathbb{Z}_p^\otimes$.
- (B2) Bob chooses a random $b \in \mathbb{N}$.
- (B3) Bob computes $B = g^{b\otimes}$ in $(\mathbb{Z}_p^\otimes, \otimes)$.
- (B4) Bob publishes his public key p, g, B in the key directory.

Alice: Encryption

To encrypt a message $m \in \mathbb{Z}_p^\otimes$ Alice does the following.

- (A1) Alice gets Bob's public key p, g, B from the key directory.
- (A2) Alice chooses a random $a \in \mathbb{N}$.
- (A3) Alice computes the shared secret $s = B^{a\otimes}$.
- (A4) Alice computes $A = g^{a\otimes}$.
- (A5) Alice encrypts m by computing $X = m \otimes s$.
- (A6) Alice sends (A, X) to Bob.

Bob: Decryption

The information available to Bob to decrypt a message are his private key b and his public key consisting of the prime p , the generator g , and $B = g^b$. To decrypt a message (A, X) Bob does the following.

- (B5) Bob receives (A, X) from Alice.
- (B6) Bob computes the shared secret $s = A^{b\otimes}$.
- (B7) Bob computes the inverse $s^{-1\otimes}$ of s in $(\mathbb{Z}_p^\otimes, \otimes)$.
- (B8) Bob decrypts the message by computing $M = X \otimes s^{-1\otimes}$.

We now show that the message M that Bob obtained in (B7) is equal to Alice's plain text message m . We have

$$M = X \otimes s^{-1\otimes} = (m \otimes s) \otimes s^{-1\otimes} = m \otimes (s \otimes s^{-1\otimes}) = m \otimes 1 = m.$$

We work through a small example.

Example 16.3.1. We follow the steps above to generate a private and a public key, encrypt a message, and decrypt a message.

Bob: Key Generation

First Bob chooses the group, the generator, and his private key and computes and publishes his public key.

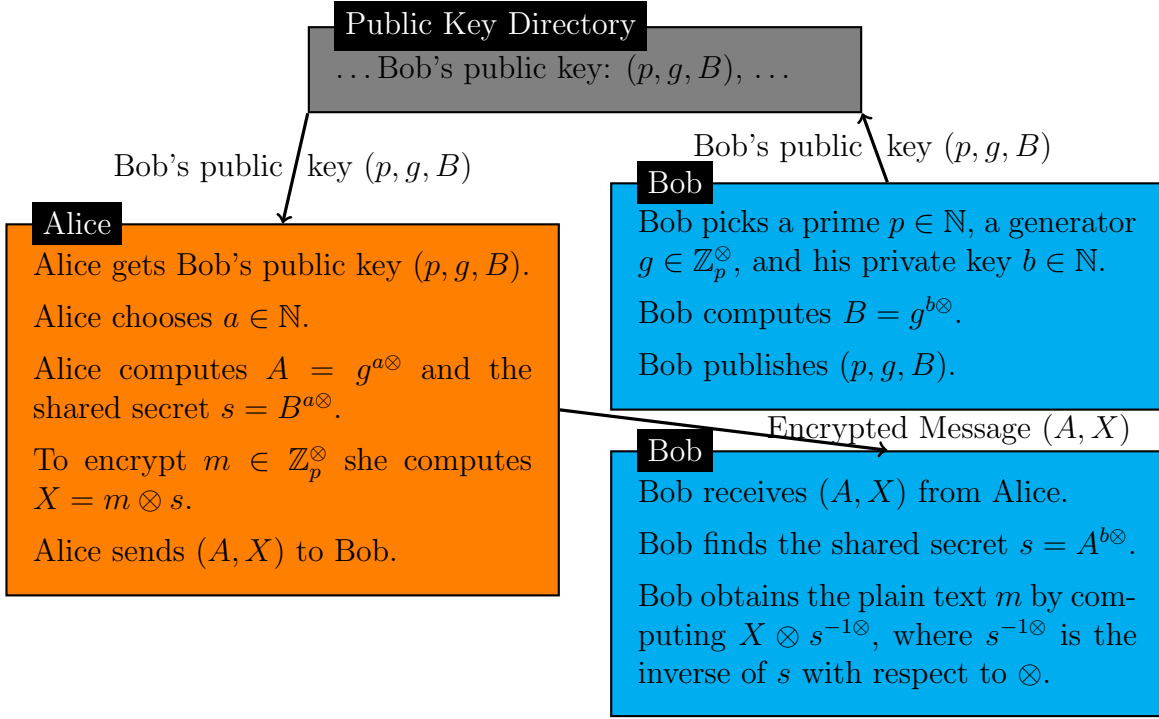
- (B1) Bob chooses the prime $p = 29$ and $g = 2$.
- (B2) Bob chooses $b = 5$ as his private key,
- (B3) Bob computes $B = 2^{5\otimes} = (2^5) \bmod 29 = 32 \bmod 29 = 3$.
- (B4) Bob publishes his public key $p = 29, g = 2, B = 3$ in the public key directory.

Alice: Encryption

Alice wants to send the secret message $m = 6$ to Bob.

- (A1) Alice obtains $p = 29, g = 2, B = 3$ from the public key directory.
- (A2) Alice chooses her secret $a = 4$.
- (A3) Alice computes the shared secret $s = B^{a\otimes} = 3^{4\otimes} = (3^4) \bmod 29 = 81 \bmod 29 = 23$.

Figure 16.3.1: ElGamal Encryption System using the group $(\mathbb{Z}_p^\otimes, \otimes)$ where $\otimes : \mathbb{Z}_p^\otimes \times \mathbb{Z}_p^\otimes \rightarrow \mathbb{Z}_p^\otimes$ is given by $a \otimes b = (a \cdot b) \bmod p$. The inverse of $s \in \mathbb{Z}_p^\otimes$ with respect to $\otimes$ is denoted by $s^{-1\otimes}$ and $b^{n\otimes} = (b^n) \bmod p$.



(A4) Alice computes $A = g^a = 2^4 = 16$.

(A5) Alice encrypts the message $m = 6$ as $X = m \otimes s = 6 \otimes 23 = 138 \bmod 29 = 22$.

(A6) Alice sends $(A, X) = (16, 22)$ to Bob.

Bob: Decryption

Bob uses A and his private key b to decrypt the message

(B5) Bob computes the shared secret

$$s = A^{b\otimes} = 16^{5\otimes} = 16^{4\otimes} \otimes 16 = (16^{2\otimes})^{2\otimes} = 24^{2\otimes} \otimes 16 = 25 \otimes 16 = 400 \bmod 29 = 23.$$

(B6) Bob finds the inverse $s^{-1\otimes} = 24$ of $s = 23$ in $(\mathbb{Z}_{29}^\otimes, \otimes)$. This can be done with the Euclidean algorithm and Bézout's identity.

(B7) Bob decrypts the message by computing $X \otimes s^{-1\otimes} = 22 \otimes 24 = 528 \bmod 29 = 6$, which is Alice's original message.

Problem 16.3.2. Bob has published his public key $p = 13$, $g = 7$, $B = 10$. His private key is $b = 2$. Alice sends him the encrypted message $(3, 8)$. What is the plain text of the message?

Solution. From Alice's message Bob gets $A = 3$ and $X = 8$. So the shared secret is $s = A^b = 3^2 = 9$. The inverse of $s = 9$ is $s^{-1\otimes} = 3$, since $9 \otimes 3 = 27 \bmod 13 = 1$. Bob decrypts the message by computing $X \otimes s^{-1} = 8 \otimes 3 = 24 \bmod 13 = 11$.

Problem 16.3.3. Alice and Bob use the ElGamal crypto system for their secure communication. From the key directory Alice obtains Bob's public key is $p = 5$, $g = 2$, $B = 4$. Alice chooses her secret $a = 2$. Alice encrypts the message $m = 4$. What does she send to Bob ?

Solution. Alice computes:

$$\begin{aligned} A &= (g^a) \bmod 5 = 2^2 \bmod 5 = 4 \bmod 5 = 4 \\ s &= (B^a) \bmod 5 = 4^2 \bmod 5 = 16 \bmod 5 = 1 \\ X &= (m \cdot s) \bmod p = (4 \cdot 1) \bmod 5 = 4 \end{aligned}$$

Thus Alice sends $(A, X) = (4, 4)$ to Bob.

We end with an example that includes the encoding of a message.

Example 16.3.4. Alice and Bob use the ElGamal crypto system for their secure communication. In the following we present all steps involved in Alice sending an encrypted message to Bob. For encoding text into numbers we apply the method from Section 12.

Bob: Key Generation

Bob chooses the prime $p = 19777$ and the generator $g = 11 \in \mathbb{Z}_{19777}^\times$. Bob chooses his secret key $b = 3$ and computes $B = (g^b) \bmod p = 1331$. Bob publishes p , g , and B in the public key directory.

Directory of Public Keys

Aaron:	$p = 19841$	$g = 243$	$B = 4821$
Beth:	$p = 19867$	$g = 128$	$B = 15522$
Bob:	$p = 19777$	$g = 11$	$B = 1331$
Sebastian:	$p = 19891$	$g = 32$	$B = 7297$
Victoria:	$p = 19913$	$g = 2187$	$B = 5531$

Alice: Encoding and Encryption

Alice wants to send the message **bat** to Bob. Alice gets Bob's public key from the directory: $p = 19777$, $g = 11$, $B = 1331$. She applies the encoding function

$$C : \{-, a, b, c, \dots, z\} \rightarrow \{0, 1, 2, 3, \dots, 26\} \text{ with } C(-) = 0, C(a) = 1, \dots, C(z) = 26$$

to the characters in the message. She obtains $C(\mathbf{b}) = 2$, $C(\mathbf{a}) = 1$, and $C(\mathbf{t}) = 20$. She encodes this into one number by computing $m = C(\mathbf{b}) \cdot 27^2 + C(\mathbf{a}) \cdot 27 + C(\mathbf{t}) = 1505$.

Alice chooses her secret $a = 2$. Alice computes the shared secret $s = (B^a) \bmod p = 11408$. She computes $A = (g^a) \bmod p = 121$

Alice encrypts the message by computing $X = (m \cdot s) \bmod p = 2604$. Alice sends A and X to Bob.

Bob: Decryption and Decoding

Bob receives A and X from Alice.

Bob computes the shared secret $s = (A^b) \bmod p = 11408$ Bob computes the inverse $s^{-1\otimes} = 14727$ of s in the group $(\mathbb{Z}_{19777}^{\otimes}, \otimes)$.

Bob decrypts the message by computing $M = (X \cdot s^{-1}) \bmod p = 1505$.

Bob finds the expanded base 27 form of M , namely $M = 2 \cdot 27^2 + 1 \cdot 27 + 20$. Decoding these numbers with C^{-1} yields the message **bat**.

In real world applications p is chosen much larger.

Symbols

$\mathbb{A} = \{-, \mathbf{a}, \mathbf{b}, \mathbf{c}, \dots, \mathbf{z}\}$	set of characters	Def. 5.4.1
$a \operatorname{div} b$	the quotient of the division of a by b	Def. 3.2.10
$a \in A$	a is an element of set A	Def. 5.3.1
$a \notin A$	a is not an element of set A	Def. 5.3.1
$\{\}$	the empty set	Def. 5.2.4
$=, \neq, <, \leq, >, \geq$	comparison of integers	Ex. 1.1.1
$A = B$	set A is equal to set B	Def. 5.3.3
$f = g$	function f is equal to function g	Def. 7.3.1
$a := b$	assign the value of b to the variable a	Ex. 1.3.1
$f : A \rightarrow B$	function f from set A to set B	Def. 7.1.1
$\star : G \times G \rightarrow G$	binary operation $\star$ on the set G	Def. 13.1.1
$f(a) = b$	function f maps a to b	Def. 7.1.1
$\gcd(a, b)$	greatest common divisor of a and b	Def. 4.2.1
$\operatorname{id}_A : A \rightarrow A, \operatorname{id}(b) = b$	the identity function on the set A	Def. 7.5.1
$a^{-1\star}$	inverse a with respect to $\star$	Def. 13.4.1
f^{-1}	inverse of function f	Def. 7.6.1
$\log_b^{\star} a$	discrete logarithm with base b	Def. 15.4.3
$\#A$	number of elements in set A	Def. 9.1.4
$\#r_1r_2g_1g_2b_1b_2$	a RGB color as a hexadecimal triplet	Section 12.2
$a \cdot b$	product of two integers a and b	Def. 1.3.7
$a \otimes b$	modular multiplication of a and b	Def. 14.3.1
$a \bmod b$	remainder of the division of a by b	Def. 3.2.10
$\mathbb{N} = \{1, 2, 3, \dots\}$	set of natural numbers	Def. 5.4.1
$A \subseteq B$	set A is a subset of set B	Def. 6.1.1
$a \oplus b$	modular addition of a and b	Def. 14.3.1
a^n	integer a to the n -th power	Def. 1.4.1
$a^{n\star}$	exponentiation with respect to $\star$	Def. 15.1.1
$\mathbb{P}$	set of prime numbers	Def. 5.4.1,
$\mathbb{Z} = \{\dots, -1, 0, 1, \dots\}$	set of integers	Def. 5.4.1
$\mathbb{Z}_n = \{0, 1, 2, \dots, n-1\}$	set of integers modulo n	Def. 5.4.1, Thm. 14.4.2
$\mathbb{Z}_n^{\otimes} = \{1, 2, \dots, n-1\}$	set of non-zero integers modulo n	Def. 5.4.1, Thm. 14.5.9

List of Figures

0.0.1	xkcd: Certainty	2
0.0.2	xkcd: Forgot Algebra	11
1.1.1	Number line	18
1.2.1	xkcd: Mnemonics (<i>excerpt</i>)	22
1.4.1	Powers of integers	31
2.5.1	A repeat joke	41
2.5.2	xkcd: Loop	44
3.5.1	Arithmetic modulo 12	59
6.3.1	Sets from Example 6.3.2	86
6.3.2	Graphical representation of the sets from Example 6.3.3	87
7.1.1	The function <code>studentid</code>	90
7.1.2	The function <code>grade</code>	91
7.1.3	Two ways of specifying the function k used in Example 7.1.4	91
7.1.4	Three ways of specifying the function g used in 7.1.5	92
7.4.1	The composite of the functions k from Figure 7.1.3 and g from Figure 7.1.4	98
7.6.1	Invertible function $f : \mathbb{Z}_5 \rightarrow \mathbb{Z}_5$	100
7.6.2	Invertible function $e : \mathbb{Z}_4 \rightarrow \mathbb{Z}_5^\otimes$	101
8.1.1	The encoding function $C : \mathbb{A} \rightarrow \mathbb{Z}_{27}$ and its inverse $C^{-1} : \mathbb{Z}_{27} \rightarrow \mathbb{A}$	106
8.1.2	xkcd: Alice and Bob	107
8.2.1	Alice and Bob and the eavesdropper Eve	108
8.2.2	Symmetric key encryption scheme	108
8.3.1	Decoder discs for Caesar ciphers	109
8.3.2	Build your own decoder disc	110
8.4.1	xkcd: Protocol	112
8.5.1	Frequency of letters	116
10.1.1	Sieve of Eratosthenes (a) and (b)	129
10.1.2	Sieve of Eratosthenes (c) and (d)	129
10.2.1	xkcd: Factoring the Time	131

10.3.1	All prime numbers less than 1660	133
10.3.2	All prime numbers greater than 1660 and less than 3728	134
11.0.1	Base 10 expansion	138
11.2.1	xkcd: Su Doku	139
11.2.2	Binary (base 2) numbers	139
11.3.1	xkcd: 1 to 10	142
11.4.1	Selected numbers in English, French, and bases 2, 3, 8, 10, 12, 16	143
11.4.2	Numbers in base 7	144
11.4.3	Hexadecimal (base 16) numbers	144
12.1.1	Encoding of an image	150
12.1.2	Images for Problems 12.1.2 and 12.1.3	150
12.2.1	Primary and secondary RGB colors	152
12.2.2	RGB color cube	153
12.2.3	Two examples of hex triplet color humor.	154
12.3.1	The encoding function $C : \mathbb{A} \rightarrow \mathbb{Z}_{27}$ and its inverse $C^{-1} : \mathbb{Z}_{27} \rightarrow \mathbb{A}$	155
12.3.2	xkcd: Code Talkers	157
14.3.1	Addition and multiplication modulo 7	177
15.4.1	Powers of elements in the group $(\mathbb{Z}_7^{\otimes}, \otimes)$	193
15.4.2	Cost of exponentiation and discrete logarithms	197
16.0.1	xkcd: Privacy Opinions	200
16.1.1	Public key cryptography	201
16.1.2	First paragraph of the article <i>New Directions in Cryptography</i>	202
16.2.1	Diffie-Hellman key exchange	203
16.2.2	Diffie-Hellman key exchange, mixing colors	204
16.2.3	xkcd: Random Number	205
16.3.1	ElGamal Encryption System	208

Index

- abelian, 173
- absolute value, 37
- Absolute value (Algorithm), 38
- acknowledgments, 10
- addition modulo n , 176
- additive inverse, 26
- Advanced Encryption Standard, 115
- AES, 115
- algorithm, 33
- Algorithm: Absolute value, 38
- Algorithm: Base Conversion, 145
- Algorithm: Conversion to Binary, 141
- Algorithm: Division for negative numbers, 50
- Algorithm: Division for positive numbers, 48
- Algorithm: Eierkuchen, 33
- Algorithm: Euclidean, 68
- Algorithm: Even or odd, 40
- Algorithm: Fast Exponentiation, 191
- Algorithm: Fortytwo, 36
- Algorithm: Four powers, 35
- Algorithm: Four powers fast, 39
- Algorithm: Maximum of two integers, 36
- Algorithm: Naive Discrete Logarithm, 195
- Algorithm: Naive Exponentiation, 44, 185
- Algorithm: Repeated Squaring, 187
- Algorithm: Sum of two integers, 35
- Algorithm: Sum up to, 40
- Alice, 107
- American Standard Code for Information Interchange, 105
- ASCII, 105
- associative, 164
- associative property of addition, 25
- associative property of multiplication, 25
- base, 27, 183
- base 10 expansion, 137
- Base Conversion (Algorithm), 145
- binary numbers, 138
- binary operation, 163
- bitmap, 149
- Bob, 107
- Caesar cipher, 107
- cardinality, 122
- Cartesian product, 84
- characters, 80
- cipher text, 107
- clock arithmetic, 59
- code, 105
- coding theory, 61
- codomain, 89
- cofactors, 70
- commutative, 170
- commutative group, 173
- commutative property, 25
- commutative property of addition, 25
- commutative property of multiplication, 25
- complexity analysis, 186
- components, 84
- composite, 127
- composite function, 96
- conjecture, 133
- Conversion to Binary (Algorithm), 141
- coprime, 66
- countable, 123
- counterexample, 25
- De Bello Gallico, 115
- decimal, 137
- decoder disc, 108, 110
- decoding, 105
- decryption, 107
- definitions, 9
- DH insecurity, 205
- Diffie-Hellman key exchange, 202
- digital signatures, 201
- discrete logarithm, 193
- distributive property, 25
- div, 51
- divides, 65
- divisible, 65
- division algorithm, 51
- Division for negative numbers (Algorithm), 50

Division for positive numbers
 (Algorithm), 48
 division of negative numbers, 49
 divisor, 65
 domain, 89

 Eierkuchen (Algorithm), 33
 elements, 78
 ElGamal encryption system, 206
 ellipses, 17
 empty set, 78
 encoding, 105
 encryption, 107
 equal (integers), 18
 equal functions, 94
 equality of ordered pairs, 85
 equality of sets, 79
 Euclidean (Algorithm), 68
 evaluation, 27
 Eve, 107
 even, 56
 Even or odd (Algorithm), 40
 examples, 9
 exercises, 9
 expansion, base 10, 137
 explain xkcd, 10
 exponent, 27, 183
 exponentiation, 27
 expression, 21

 factor, 65
 factorial, 43
 false, 19
 fast exponentiation, 188, 190
 Fast Exponentiation (Algorithm), 191
 finite, 123
 for all, 24
 Fortytwo (Algorithm), 36
 Four powers (Algorithm), 35
 Four powers fast (Algorithm), 39
 frequency analysis, 114
 function, 89

 gcd, 66
 given any, 26
 graph of a function, 93

 greatest common divisor, 66
 group, 173
 group operation, 173

 hex triplet, 151

 identity element, 166, 173
 identity function, 98
 if (**if** __ **then**), 36
 image of a function, 92
 image of an element, 89
 infinite, 123
 integers, 17, 80
 inverse, 100, 168, 173
 inverse, additive, 26
 invertible, 100
 is in, 79
 ISBN, 61

 Julius Caesar, 107

 let, 23
 let (**let** __ :=), 38

 Maximum of two integers (Algorithm), 36
 mod, 51, 55
 multiple, 65
 multiplication modulo n , 176

 Naive Discrete Logarithm (Algorithm),
 195
 Naive Exponentiation, 44
 naive exponentiation, 185
 Naive Exponentiation (Algorithm), 44,
 185
 natural numbers, 17, 80
 navigation, 10
 negative integers, 17
 non-abelian, 173
 non-commutative, 173
 non-negative, 24
 null set, 78

 odd, 56
 operation, 163
 order of operations, 61
 ordered pair, 84

 perfect square, 30

- pixels, 86, 149
- plain text, 107
- positive integers, 17
- preimage, 89
- prime, 127
- prime factorization, 128
- private key, 199
- problems, 9
- public key, 199
- public key cryptography, 199
- quotient, 47, 48, 51
- raster, 85
- Red Green Blue, 151
- remainder, 48, 51
- repeat (**repeat**____**until**), 39
- repeated squaring, 185, 186
- Repeated Squaring (Algorithm), 187
- return (**return**), 35
- RGB, 151
- RGB hex triplet, 151
- roster form, 78
- set, 78
- Set-builder notation, 81
- Sieve of Eratosthenes, 128
- space, 105
- square root, 30
- step, 34
- student learning outcomes, 7
- subset, 83
- Sum of two integers (Algorithm), 35
- Sum up to (Algorithm), 40
- symmetric key cryptography, 107
- then (**if**____**then**), 36
- theorems, 9
- there exists, 26
- trapdoor function, 200
- trapdoor functions, 199
- trial division, 132
- triple, 151
- true, 19
- twin prime pair, 134
- Unicode, 105
- uniqueness of identity, 167
- uniqueness of inverses, 174
- until (**repeat**____**until**), 39
- values of the places, 137
- variable, 22
- well-defined, 77
- whole numbers, 80
- World Wide Web, 151
- xkcd comic strip, 10
- xkcd: 1 to 10, 142
- xkcd: Alice and Bob, 107
- xkcd: Certainty, 2
- xkcd: Code Talkers, 157
- xkcd: Factoring the Time, 131
- xkcd: Forgot Algebra, 11
- xkcd: Loop, 44
- xkcd: Mnemonics (*excerpt*), 22
- xkcd: Privacy Opinions, 200
- xkcd: Protocol, 112
- xkcd: Random Number, 205
- xkcd: Su Doku, 139