2 What do mathematicians study? [3]

The short answer is: mathematicians study *mathematical objects* that inhabit the *mathematical universe*, with the goal of discovering and confirming the universal principles of this universe: the patterns, symmetry and laws that individual objects obey. In this section we'll take our first look at this abstract universe, and at the universal principles it obeys.

2.1 Introducing the mathematical universe

The mathematical universe is populated by countless *mathematical objects*, which are classified according to *object types*. Here are some examples of object types:

- Natural numbers
- Real numbers
- Finite Sets whose members are natural numbers
- Lists of length 5 whose entries are real numbers
- Functions mapping each list of real numbers of length 5 to a single real number
- Matrices whose entries are polynomials in the variable x having real numbercoefficients,
- *Groups*,
- *Rings*,
- *Topological Spaces*.

In mathematics, simple words such as groups and rings, represent types of objects with very precise and intricate structure.

Objects belonging to a given object type can be manipulated and transformed in ways that are particular to that type. Natural numbers can be added together and multiplied together, and we can compare two natural numbers to decide whether they are same or different, and if they are different we can determine which is greater than the other. We can add two matrices of the same shape, but we can't add two matrices of different shapes. We can take the derivative of a (differentiable) function, but we can't take the derivative of a matrix of real numbers. We can compute the determinant of a square matrix, but not of a non-square matrix. We can take the union of two sets, but we can't take the union of two real numbers.

In the mathematical universe, complicated types of objects are built up from simple types of objects. In fact, mathematicians have shown that starting only from the concept of a set, one can build up all other types of objects (including, numbers, functions, and all of the other objects listed above). Capturing all of mathematics within the theory of sets is a fascinating

[3] Version 10/1/2016.

achievement of mathematical logic, but is not so important for what we do here. Our goal is to learn how to think like a professional mathematician, and a mathematician does not think of numbers as built up from sets. He thinks of numbers as ... well ... numbers. So we start with a larger (but still fairly small) list of basic object types that serve as out building blocks for the mathematical universe.

The following object types will be the central focus of this course:

Numbers : *Integers* and *Real Numbers*

Collections : *Sets* and *Lists* and *Indexed Families*

Correspondences between sets : *Functions*

Most readers have a great deal of famiiarity with numbers, and some famiiarity with sets, lists and functions. In this section, we'll carefully introduce some basics about these types. For many readers, these basics will mostly be review. However, the general framework we present concerning how to think about mathematical objects, will be new to most readers.

A conceptual framework for all types of mathematical objects. Different types of mathematical objects can be very different, but there is a common framework that applies to every object type. We'll state this framework as a series of four questions that every mathematician thinks about (consciously or unconsciously) when meeting a new type T of object.

- *Specification.* How do we refer to a particular object of type T. We need notation for naming, or *specifying* individual objects of type T.
- *Equality.* What does it mean that two objects of type T are equal?
- *Comparison.* What are the natural ways for comparing two objects of type T.
- *Operations.* An operation on objects of type T is a rule that takes as input one or more objects of that type, and produces as output an object of the same type. If the operation takes as input exactly one object it is called a *unary operation* or an *operator*. If it takes as input two objects of the type it is a *binary operation*. What are the important operations for type T?

We'll now use this framework to introduce each of the basic types mentioned above. Depending on the type we are considering, we may consider these questions in a different order.

Numbers: Real numbers, integers, and rational numbers

The integers consist of the number 0, the positive integers 1,2,3,... and the negative integers -1,-2,-3,.... Let's consider the above questions for the integers. The rational numbers include the integers, as well as ratios of integers, i.e., numbers written as fractions. The real numbers

include the rational numbers, and also other numbers. We have the useful mental picture that the real numbers are in correspondence with the points on an infinite horizontal line.

Ancient mathematicians originally thought that the rational numbers were the only real numbers, but the Greeks realized that there are numbers, such as $\sqrt{2}$ that are not rational, and later it was realized that the numbers π (the ratio of the circumference of a circle to its diameter) and e (the base of the natural logarithm) are also irrational. In fact, in the 19th century, the mathematician Georg Cantor established that (in a precise sense) the vast majority of numbers are irrational. (We'll come back to this in a later chapter.)

Teaser 2.1. What is the justification for saying that $\sqrt{2}$ and π are irrational? How can we be sure that neither of these numbers can be expressed as a ratio of integers?

(Note: Here and throughout these notes, "Teasers" are questions for you to think about, but that you are not necessarily ready to answer. In most cases, we will return to the question later in the notes, and provide an answer, which will require a *mathematical proof*.)

We consider integers, rational numbers and reals, to be three distinct object types. When we say that these types are *distinct* that doesn't mean that they involve different objects. In fact every integer is a rational number, and every rational number is a real. So the number 7 is of type integer, and also of type rational, and also of type real number. Many systems for classifying individuals share this feature, for example, in zoology, where the terms *primate*, *mammal* and *vertebrate* represent types of animals, and every primate is a mammal, and every mammal is a vertebrate.

Operations on Numbers. There are several familiar binary operations for real numbers: addition ($+$), subtraction ($-$), multiplication ($\times$), and division ($\div$); as usual we often write a/b instead of $a \div b$. Notice that there are restrictions on the use of $\div$; $a \div b$ is not defined if $b = 0$. If the operation is only defined under some restrictions on the input we say it is a *partially defined* operation.

We say that a type is *closed* under an operation if the combination of two items of the type is another item of the type. So the integers are closed under addition, subtraction and multiplication, but not under division since $a \div b$ need not be an integer even if a and b are. The rational numbers and real numbers are also closed under all four operatons, keeping in mind that $a \div b$ is undefined (and therefore has no meaning) when $b = 0$.

There is also the unary operation of negation $x \longrightarrow -x$. For the real numbers, we also have the power operation $x \longrightarrow x^p$.

Exercise 2.1.

1. If p is a fixed positive integer, for which values x is the operation $x \longrightarrow x^p$ defined?
2. If p is a fixed negative integer, for which values of x is the operation $x \longrightarrow x^p$ defined?
3. If p is a positive rational number, what are the restrictions on x needed so that $x \longrightarrow x^p$ is defined?

Numerical and variable expressions. The sequence of symbols "$(5+7)\times(9-(4\div 3))$" is called a *numerical expression.* It gives instructions for combining numbers into a single number, and so is a way of representing a single number. More generally, we have expressions that also involve variables such as $(x-y\times z)\div(x+y+4\times z)$. This expression also provides instructions for combining number into a single number, but here the numbers to be combined are not specified. We call this kind of expression a *variable expression.* This expression does not represent a single number, but if we were to choose values for the variables, such as $x=5$, $y=-3$ and $z=7$ then the expression does represent a specific number.

Comparison of numbers The natural way to compare numbers is via the relationships "less than", "less than or equal to", "greater than" and "greater than or equal to". These have the familiar symbols: $<$,$\leq$, $>$ and $\geq$.

There is another important way to compare numbers: by *divisibility.* We say that a is a *divisor* of b or b is a *multiple* of a, provided that $b\div a$ is an integer, and this is represented by the notation $a|b$. Note that we usually use the notation $a|b$ only in the case that a and b are integers, but it makes sense for real numbers, so $\frac{7}{12}|\frac{35}{6}$ and $3\pi|15\pi$.

Equality of numbers It may seem odd that we even have to discuss equality of numbers. What is there to say? Two numbers are either the same of they're not.

The issue of equality is interesting because we can describe numbers in different ways, and when we do this it requires work to tell whether they are equal. To tell whether 18×24 and $192+140$ can be done by straightforward computation. To show that $10257\times(98431+43572)$ is equal to $10257\times 98431+43572\times 10257$, we can show that these are equal by computation, but a simpler way is to invoke the familiar *distributive* and *commutative* laws of arithmetic. More generally, the dcommutative and associative laws of addition and multiplication, and the fact that multiplication distributes over addition give us a way to change expressions involving numbers and the operations $+,-,\times$ and $\div$, into other expressions that represent the same number. (We'll review these laws in detail later.)

Here are two other important criteria for numbers a and b to be equal:

If both $a\leq b$ and $b\leq a$, then a and b must be equal.

If both a is a divisor of b and b is a divisor of a then either $a=b$ or $a=-b$.

Specification of numbers. The familiar base 10 (or decimal) system uses 10 symbols 0,1,2,3,4,5,6,7,8, and 9 (called *digits*) and uses a string of these symbols, such as 483299, to represent a positive integer. Most of us have used the base 10 representation since we were very young, and it is common to think of the integer and its base 10 representation as being the same thing. Strictly speaking, however, 1001 is *not* an integer, it is a string of symbols that is used to specify a number. In the binary system for representing integers, every positive integer is represented using strings consisting just of 0's and 1's. In the binary system 1001 represents a different integer than in the base 10 system.

The decimal system establishes a correspondence between the set of strings of digits, and the set of integers; every string of digits corresponds to an integer. It is possible for two strings of digits to represent the same integer, for example 00456 specifies the same integer as 456. A zero at the beginning of the string is called a *leading zero* and we usually don't use leading 0s in our string since they are not needed. If we disallow leading 0's then we have the remarkable fact that the correspondence between the set of decimal strings without leading 0's, and the set of positive integers is a *bijection* or *one-to-one correspondence*. This statement means two things:

1. For every positive integer there is a decimal string without leading 0's that represents it (so there is no positive integer that doesn't have a string representing it.)

2. Two different decimal strings without leading 0's represent different integers (so no integer can be represented in this way by two different strings).

Teaser 2.2.

What is the justification for the above two claims?

The representation described so far only allows us to specify positive integers and 0. To specify negative integers we use the familiar negation symbol "−". Using this we get a unique representation for all integers.

Using the operations on numbers, we can have more complicated specifications: for example $11 + 5 \times (9 - 3)$ specifies an integer. Once we allow operations, the representation of numbers is no longer unique, in fact there are infinitely many ways to represent any integer.

What about the rational numbers? A rational number is the ratio of two integers, so (as we all know) we represent a rational number by writing s/t where s and t are decimal strings.

What about real numbers? For those real numbers that are rational, we can represent them as a fraction as above. We have special symbols like π and e to name some of the most important irrational numbers. Also, we can use the square root operator to specify numbers such as $\sqrt{2}$ and $\sqrt{47}$. We can build up more complicated numbers such as:

$$\sqrt{5\sqrt[3]{97} + 7\sqrt{3}}.$$

We can use expressions like this, involving addition, subtraction, multiplication, division, square roots, cube roots and higher roots to greatly enlarge the collection of numbers we can represent.

Teaser 2.3. Can all real numbers be represented by expressions of this kind? If yes, how can we be sure that this is true. If no, this means that there are numbers that can't be represented by such an expression. How do we know that there are such numbers?

We'll see in a later chapter that the answer to this teaser is "no"; there are real numbers that can't be represented this way.

So we still have the problem of how to specify real numbers. You probably learned that every real number can be represented by an infinitely long decimal such as $359.04370098\ldots$.

The catch is that we haven't really fully specified the number. The "..." at the end means that we are omitting the digits that follow. We're not specifiying the real number, only an approximation to it.

We can exactly represent every rational number by its decimal representation. In fact, we have the following two facts:

- For every rational number its decimal representation either *terminates* (which means that from some point on every digit is 0), for example 42.896 or is *repeating* (which means that from some point on, there is a single pattern of digits that repeats), for example $28.543\overline{6723}$. Here the notation $\overline{6723}$ means the pattern 6723 repeats endlessly.
- Every terminating or repeating decimal represents a rational number.

By now, the reader should be able to anticipate the following question:

Teaser 2.4. How do we justify the two statements above?

These two statements imply that the notation of terminating or repeating decimals gives another way of specifying rational numbers. However, when we consider all real numbers we remain with the question:

Teaser 2.5. Is there *any* notational system that allows us to specify every real number exactly?

It turns out that the answer is "no", there is no notational system that allows us to specify every real number exactly! This is a mind-boggling statement: we are not just saying that we don't know such a system, but that it is *impossible* to ever discover such a system. Later, we'll see how mathematicians have established this statement to be true.

But now we're left in this situation: we have a type of mathematical object, the real numbers, but we are only able to specify a small number of the objects of this type. This turns out not to be such a problem. The "interesting" real numbers, those that have important mathematical properties, and are solutions to mathematical problems, can all be specified. All the others exist anonymously in the background of mathematics, and individually play almost no role in our exploration of the mathematical universe.

Collection objects: Sets, Lists and Indexed Families

In astronomy, there are many basic types of objects such as stars, planets, moons, asteroids, and comets. Then there are also more complex objects, such as *solar systems*. A solar system consists of a star, together with all of the objects (planets, moons, comets, etc.) that orbit around that star. A solar system is a *single astronomical object* that consists of a collection of other astronomical objects.

In the mathematical universe, we can also collect together objects to create a single new object There are two main types that represent collection objects: *sets* and *lists*; another important type of object is *indexed family*, which generalizes lists.

Sets

The idea of a set. A *set* is a collection of objects where we ignore both the ordering of the objects, and ignore duplicate copies of any object. For a set, every object is either a *member* of the set or a *non-member* of the set For a set A and an object x we have the following important notation:

$x \in A$ means that x is a member of A

$x \notin A$ means that x is not a member of A.

Comparison of sets, and Equality of sets. The most important way to compare two sets is to ask whether one is a *subset* of another. We have the following definitions.

Definition 2.1. *A is a subset of B*, written $A \subseteq B$ means that every element of A belongs to B.

A is a superset of B, written $A \supseteq B$ means that B is a subset of A.

$A = B$ means that $A \subseteq B$ and $B \subseteq A$.

Some examples of sets. Numbers were the first basic mathematical type we introduced. Now that we've introduced sets, it is natural to first consider sets whose members are numbers. Here are some examples of sets, specified using *list notation*.

$A = \{2, 4, 6, 8, 10\}$

$B = \{10, 8, 6, 4, 2\}$

$C = \{2, 2, 4, 4, 6, 6, 8, 8, 10, 10\}$

$D = \{2, 6, 10\}$

$E = \{2, 4, 8, 10\}$.

$F = \{2\}$

$G = \{\}$

In thie above notation, we use one of the standard notations for sets, where we list the members, separated by commas and surround the set by braces $\{\}$.

The final set G has no members, and is called the *empty set*. It may seem strange to consider this a set, but it's quite an important set. Think of a set as a *container* which may or may not have things in it. Even if the container is empty it is still something. An alternative notation for the empty set is $\emptyset$, so $\emptyset$ and $\{\}$ are both acceptable ways to represent the empty set, but $\{\emptyset\}$ *is not the empty set*. This set is a set, that has one member, and that member is $\emptyset$. Think of this set as a box that has one object inside of it, and the object inside of it is an empty box.

Set F has one element. A set with one element is called a *singleton set*. It is tempting to think of $\{2\}$ and 2 as the same mathematical object, but they are not! The object 2 is an object whose type is number. The object $\{2\}$ is an object whose type is set.

Notice that sets A, B and C are all the same set, so $A = B = C$, because our definition of equality of sets ignores the order in which members are listed, and ignores repeated appearances of a member. The notation used to describe set C is strictly speaking legal, but in practice there is rarely a good reason to list members more than once in a set, so we almost do it. We only use it here to make the point that C is exactly the same set as A.

Advanced remark 2.1. (Note: An *advanced remark* presents some related information that goes beyond the scope of this course, that are included for students who want to delve more deeply.) There is another type of mathematical object called a *multi-set*, which we mention now, but won't consider further. In a multiset, the same object may appear multiple times, and we care not just about whether an object is a member of the set or not, but also how many copies of the element belong to the set. If we are dealing with the multi-set object, then A and B are still the same set, but C is different from A and B.

Notice that D and E are subsets of A, B and C and neither is a subset of the other. Also F is a subset of all the sets that precede it, and G is a subset of every set. This last fact follows from the general principle that the empty set is a subset of every set.

The above sets are all *finite*. We'll define more precisely what we mean by finite later, but for now we'll use the intuitive idea that a finite set is one where you can count all the members to obtain a definite number called the *size* or *cardinality* of the set. The size of a finite set S is denoted $|S|$. Notice that all three of the sets A, B, C in the above exampe have size 5. Set F has size 1, and set G has size 0.

Remark 2.2. This is the first of many examples of notation that has a different meaning depending on the type of object it is applied to $|Q|$ means "absolute value of Q" if Q is a number, and means "the size of Q" if Q is a finite set. So to understand what $|Q|$ means it is crucial to know what kind of object Q is.

Using the same notation in two different ways depending on context is called *overloading* the notation. Mathematicians need to represent many different ideas, and there are a limited number of symbols to choose from, so overloading is quite common.

Operations for combining sets. The main operations for combining sets are *union* (written $\cup$), *intersection* (written $\cap$) and *difference* (written $\setminus$). For sets A and B we define:

$A \cup B$ is the set obtained by merging A and B. An object x is a member of $A \cup B$ provided that $x \in A$ or $x \in B$. (Keep in mind that in mathematics "or" includes the possibility that both $x \in A$ and $x \in B$.)

$A \cap B$ is the set of elements that belong to both A and B. Thus an object x is a member of $A \cap B$ if $x \in A$ and $x \in B$.

$A \setminus B$ is the set of elements that belong to A but not B. Thus an object x is a member of $A \setminus B$ if $x \in A$ and $x \notin B$.

If A is a subset then the *complement of* A denoted A^c is the set $U \backslash A$ where U is the universe under discussion.

Question 2.2. (In these notes, a *question* is something to think about that you may not be ready to fully solve yet. Questions are similar to teasers, but usually easier.)

1. Is the operation $\cup$ commutative? Is it associative?
2. Is the operation $\cap$ commutative? Is it associative?
3. Is the operation $\backslash$ commutative? Is it associative?
4. Does $\cap$ distribute over $\cup$, that is, is it true for any three sets A,B,C that $A \cap (B \cup C) = (A \cup B) \cap (A \cup C)$?
5. Does $\cup$ distribute over $\cap$, that is, is it true for any three sets A,B,C that $A \cup (B \cap C) = (A \cap B) \cup (A \cap C)$?
6. Does $\cap$ distribute over $\backslash$? Does $\cup$ distribute over $\backslash$?
7. Does $\backslash$ distribute over $\cap$? Does $\backslash$ over $\cup$

We'll return to these questions when we start doing mathematical proofs.

Using the sets in the previous example to illustrate these concepts we have $C \cup D = C$, $C \cap D = D$, $C \backslash D = \{4, 8\}$ and $D \backslash C = \emptyset$. Also $D \cup E = \{2, 4, 6, 8, 10\}$, $D \cap E = \{2, 10\}$, $D \backslash E = \{6\}$ and $E \backslash D = \{4, 8\}$.

Two sets A and B are said to be *disjoint* if $A \cap B = \emptyset$ and said to be *intersecting* otherwise.

"Simple sets" of real numbers. For now we're focusing on sets whose members are numbers. The examples given above are small finite sets. Here we'll introduce notation for some subsets of real numbers that are particularly important and, despite being infinite in most cases, are conceptually quite simple

- $\mathbb{R}$ is the set of real numbers, and $\mathbb{R}_{>0}$, $\mathbb{R}_{\geq 0}$, $\mathbb{R}_{<0}$ and $\mathbb{R}_{\leq 0}$ denote, respectively, the sets of positive real numbers, nonnegative real numbers, negative real numbers and nonpositive real numbers.
- $\mathbb{Z}$ is the set of integers, and $\mathbb{Z}_{>0}$, $\mathbb{Z}_{\geq 0}$, $\mathbb{Z}_{<0}$ and $\mathbb{Z}_{\leq 0}$ denote, respectively, the sets of positive integers, nonnegative integers, negative integers and nonpositive integers.
- $\mathbb{Q}$ is the set of rational numbers, and $\mathbb{Q}_{>0}$, $\mathbb{Q}_{\geq 0}$, $\mathbb{Z}_{<0}$ and $\mathbb{Z}_{\leq 0}$ denote, respectively, the sets of positive integers, nonnegative integers, negative integers and nonpositive integers.

There are some other subsets of $\mathbb{R}$, for which we have special notation.

Intervals of real numbers For any two real numbers a and b we have the following notation:

$(-\infty, b]$ is the set of real numbers x satisfying $x \leq b$.

$(-\infty, b)$ is the set of real numbers x satisfying $x < b$

$[a, \infty)$ is the set of real numbers x satisfying $x \geq a$.

(a, ∞) is the set of real numbers x satisfying $x > a$.

$[a, b]$, is the set $[a, \infty) \cap (\infty, b]$, which is the set of real numbers x satisfying $a \leq x$ and $x \leq b$.

$[a, b)$ is the set $[a, \infty) \cap (\infty, b)$, which is the set of real numbers x satisfying $a \leq x$ and $x < b$.

$(a, b]$ is the set $(a, \infty) \cap (\infty, b]$, which is the set of real numbers x satisfying $a < x$ and $x \leq b$.

(a, b) is the set $(a, \infty) \cap (\infty, b)$, which is the set of real numbers x satisfying $a < x$ and $x < b$.

Question 2.3. What can be said about the last four sets in the case that $a > b$? What happens when $a = b$?

The first four sets are always nonempty, but depending on the values of a and b, the second four sets may be empty. We're usually only interested in the second four sets when they are nonempty.

The final six sets contain no member smaller than a. We say that each of these sets is *bounded below*. We also say that a is the *lower endpoint* of the set (except in the case that the set is empty). The third, fifth and sixth sets all contain the lower endpoint, and are said to be *lower closed*. The fourth, seventh and eighth sets do not contain the lower endpoint, and are therefore said to be *lower open*. The first and second sets have no lower restriction on the numbers in the set, and we say they are *unbounded below*.

Similarly, the first two sets and the last four are said to be *bounded above* and b is said to be an *upper endpoint* of the set (unless the set is empty). The first, fifth and seventh sets contain the upper endpoint and are therefore *upper closed*, while the second, sixth and eighth sets are upper open.

The final four sets are bound bounded above and bounded below, and so are said to be *bounded*. The fifth set is both upper closed and lower closed, so is said to be *closed*. The eighth set is both upper open and lower open and so is said to be *open*.

Notice that if $a \geq b$ then each of the sets $[a, b)$, $(a, b]$, (a, b) is empty. If $a > b$ then $[a, b]$ is empty and if $a = b$ then $[a, b] = \{a\}$. If $a < b$ then the sets $[a, b]$, $[a, b)$, $(a, b]$ and (a, b) are all infinite.

Question 2.4. Is the intersection of two intervals always an interval. Is the union of two intervals always an interval? Is the difference between two intervals always an interval?

Two intervals whose union is not a single interval are said to be *separated*. A set that is the union of intervals, where any two are separated, such as $[-9, 3) \cup (5, 8) \cup [10, \infty)$ is the union of three intervals, is said to be in *union of separated intervals* form.

Intervals of integers There is an analog of interval for the set of integers. For any integers a and b we have the following notation:

$\{a, \ldots, \infty\}$ is the set of integers n satisfying $n \geq a$.

$\{-\infty, \ldots, b\}$ is the set of integers n satisfying $n \leq b$.

$\{a, \ldots, b\}$ is the set of integers n satisfying $n \geq a$ and $n \leq b$.

Notice that $\{a, \ldots, b\}$ is always a finite set. It is empty if $a > b$ and otherwise has size $b - a + 1$.

Question 2.5. You'll notice that we defined 8 types of real intervals, while only 3 types of integer intervals. In the case of real intervals we distinguished between whether the variable x satisfied $x \geq a$ or $x > a$, while in the integer interval case we only considered whether the variable n satisfied $n \geq a$. Why?

The power set For any set S, the *power set of* S, denoted $\mathcal{P}(S)$, is the set of all subsets of S. For example $\mathcal{P}(\{1, 2, 3\})$ is the set $\{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$. We also write $\mathcal{P}_{\mathbf{fin}}(S)$ for the set of all finite subsets of S. Thus $\mathcal{P}_{\mathbf{fin}}(\mathbb{Z}\}$ is a set which has a member the set of all integers from 1 and 100, but does not have as a member the set of even integers.

Fixing a universe Whenever we are discussing sets, we will assume that the sets under consideration are all subsets of one common set, called the *universe*. The universe depends on the context. All of the sets discussed so far in this section are sets of real numbers, and so we can take the universe to be $\mathbb{R}$. If all sets under discussion consist of integers, we might take $\mathbb{Z}$ as the universe. Normally, we'll always have some universe in mind. If we don't specify the universe, we often use the letter U to denote the universe. When the universe is U then "subset of U" is a type of mathematical object. Notice that the set of all objects of type "subset of U" is $\mathcal{P}(U)$.

W e have seen one way to specify a set is by listing all of its members. This is possible only for finite sets. Even for finite but large sets (say more than 100 members) it is impractical to list all of the elements.

Mathematicians have a general notation for specifying sets. The format of this notation is:

$$\{[\text{STUFF}] : [\text{STUFF}]\}$$

The format consists of an left set bracket "{" followed by some "STUFF" followed by a colon ":" followed by more "STUFF" and ending with a right set bracket "}". Of course, we need to say something about what kind of "STUFF" is allowed. Let's look at two examples.

Example 1. $\{x : x \in \mathbb{R}, x - \sqrt{x} \geq 80 \text{ and } x^2 \leq 10000\}$.

This notation is read as:

The set of those objects x such that x is a member of $\mathbb{R}$ and satisfies the requirements $x - \sqrt{x} \geq 80$ and $x^2 \leq 10000$.

This is often shortened as:

The set of all real number x that satisfy $x - \sqrt{x} \geq 80$ and $x^2 \leq 10000$.

The colon abbreviates the phrase "such that". Before the colon is a single variable, and after the colon there is a list of requirements that the object represented by the variable must satisfy to be in the set. There is nothing special about the use of the variable x. We could have used any letter in its place, but *it is essential that the letter used before the colon matches the letter used after the colon.*

Example 2. $\{a^2 - b^2 : a \in \mathbb{Z} \text{ and } b \in \mathbb{Z}\}$

This is read as:

The set of those objects that can be expressed as $a^2 - b^2$ where a is an arbitrary integer and b is an arbitrary integer.

This is often shortened to:

The set of numbers of the form $a^2 - b^2$ where a and b are aribtrary integers.

In this format, the part before the colon represents an expression involving one or more variables. After the colon are the restrictions placed on the variables.

The general format of both of these examples is:

{ [expression involving one or more variables] : [constraints on the variables] }

In the first example, the expression on the left is very simple, it is just a single variable x, while the constraints on the right are somewhat involved. We call this a *constraint specification* of the set.

In the second example, the expression on the left is complicated, while the contstraints on the left are simple, in that each variable is restricted separately to be in some "simple" set. We call this a *parametric specification* of the set. What do we mean by a "simple" set? This is somewhat vague, but in the case of real numbers, we usually think of a "simple" set as either all of $\mathbb{R}$ or a union of intervals of real numbers, while in the case of integers, a simple set is either all of $\mathbb{Z}$ or a union of intervals of integers..

To understand the difference between specification by constraint and parametric specification, let's consider two basic problems associated to a set S:

Membership testing problem Given a particular object, is that object a member of S?

Example generation problem Produce an example of a member of S (or determine that S is empty).

Question 2.6. The union of separated intervals form is a particularly convenient form for solving both of these problems. Explain how you can solve these problems for a set given as a union of a finite list of intervals.

Consider these problems for the two sets specified above. The first set is given by a constraint specification, and if we're given a number, say 50, we can easily check whether it is a member of the set by by testing whether 50 satisfies the two requirements. However, if I ask you to generate an example from the first set, it's not obvious how to do it. You can start making guesses and check your guesses by seeing if its in the set. But if all of our guesses fail to be in the set, because each fails to satisfy one of the constraints, we still don't know whether some example might be in the set.

For the second example, the set is given by parameteric specification, if I want to generate a single member of the set, that's easy: just pick any two integers a and b, say 9 and 3 and compute $9^2 - 3^2 = 72$. But suppose I give you a number like 6774 and I ask whether it belongs to the set. How can you check this? You can search for value of a and b that work, but it may take a long time to find such values. Ane if you don't manage to find a and b such that $a^2 - b^2 = 6774$, you still can't answer the question; maybe there are a and b that work but you just didn't look at them.

In general:

- If the set S is given by a constraint specification then the membership testing problem tends to be easy (and the example generation problem may be easy or hard).
- If the set S is given by a parametric specification, then the example generation problem is easy, but the membership testing problem may be hard or easy.

It turns out that using the above notation, there are many different ways to specify the same set. Very often mathematicians are given a set in one form, say by constraint satisfaction, and wish to find an alternate specification of the same set, for example by parametric specification.

We can relate this to the kind of mathematics that you've done in previous classes. In a typical mathematical problem, you are given one or more variables, and one or more requirements or constraints, and you are asked to "solve" for the variables. For example, you are given the variable x and asked to find all real number solutions to the inequality $x^2x \geq 8$. The set of solutions can be expressed by constraint specification as:

$$\{x : x \in \mathbb{R} \text{ and } x^2 - 2x \geq 8\}.$$

Now, this completely specifies the set of solutions, but it is not considered to be a satisfactory solution. All we've done is reformulate the information given in the problem. As discussed above, a constraint specification often doesn't allow us to tell whether the set is nonempty. What we are asked to do in such a problem find an *alternative way to specify the set* that makes it easy to tell whether the set is empty, and if it's not empty makes it easy to find solutions. For sets of real numbers, often the most desirable specification is as a finite union of intervals, but such a specification is not always possible. For this problem (and many like it)

it is possible, and we can rewrite the above set as $(-\infty, -2] \cup [4, \infty)\}$. In cases where we don't have a representation as a finite union of intervals, we may still be able to give a parameteric representation, which enables us to tell whether the set is empty, and to generate members.

Constraint specification and parametric representation are two examples of a general method of specifying sets. The general format is:

{[Expression involving one or more variables] : [Requirements involving the same set of variables]}

Here's an example of the use of this more general format:

$$\{x \times y : x \in \mathbb{Z}, y \in \mathbb{Z}, x + y \text{is equal to a power of } 2\}.$$

For example, 12 belongs to this set since $12 = 6 \times 2$, and $6 + 2$ is a power of 2. On the other hand 5 does not belong to this set since 5 can only be written as a product as 5×1 and $5 + 1$ is not a power of 2.

The more general specification is *more expressive* (it can describe more sets than either constraint specification or parameteric specification) but it is often *less useful*. For example, given a set with this more general specification, it may not be easy to solve either the membership testing problem or example generation problem.

All of the examples we've seen are sets of numbers. We can specify sets of other objects. Here's a specification for a set whose members are sets:

$$\{[a, b] : a, b \in \mathbb{R} \text{ and } a^2 \leq b\}$$

Here, the expression on the left is $[a, b]$, which as defined earlier is an interval of real numbers. So the set being defined is a set of intervals. The restriction implies that only those intervals with $a^2 \leq b$ are included, so this would include the intervals $[-2, 7]$, $[2, 7]$ and $[3, 10]$ but not $[4, 8]$.

Remark 2.3. *A commonly used variant for constraint specification.* When we use constraint specification: $\{z : [\text{constaints on } z]\}$, it is often the case that one of the constraints is a *type specification* of the form"$z \in T$" where T is a type of mathematical object. This constraint simply tells us that the set is a subset of T. For example in the specification $\{n : n \in \mathbb{Z}, n \text{ has at most 2 prime divisors}\}$, the first constraint just tells us that the set consists only of integers.

There is a commonly accepted way to shorten the specification by replacing "$z : z \in T$" simply by "$z \in T$:". Using this format the above set is written $\{n \in \mathbb{Z} : n \text{ has at most 2 prime divisors}\}$, which is read as "The set of integers n that have at most 2 prime divisors".

Lists

A *list* is a different kind of collection of objects, in which the objects are presented in a specific order and we pay attention to repeated elements.. Each separate appearance of an object in a list is called an *entry* or a *coordinate* of the list. Here are some examples of lists:

$a = (2, 4, 6, 8, 10)$

$b = (10, 8, 6, 4, 2)$

$c = (3, 3, 3, 3)$

$d = (1, 3, 1, 3, 1, 3)$

$e = (17)$.

$f = ()$.

These lists are represented in the standard notation for lists, in which the entries are separated by commas and the list is surrounded by parentheses (). Lists a and b each have 5 entries. Notice that entries of these two lists are the same, but in different order, and for this reason $a \neq b$.

List c has 4 entries, all of which are the same. List e has a single entry, and list f has no entries, and is called the *empty list*.

All of the above lists have entries that are numbers, but as we'll see, the entries of a list can be any mathematical object. You've probably already encountered lists of numbers in prevous courses, where they were called *vectors*.

Unlike sets, lists are restricted to have a finite number of entries. For a list v, the number of entries is called its *length* and is denoted $\mathbf{length}(v)$. Each entry of v appears in a specific *position* that is a number from 1 to $\mathbf{length}(v)$. Thus in the list a, 8 appears in positon 4, and in list b, 8 appears in position 2. We often write L_j for the entry of L appearing in position j. Thus for list d we have $d_1 = d_3 = d_5 = 1$ and $d_2 = d_4 = d_6 = 3$.

Notation for lists If a is a list of length k we denote this by $(a_i : 1 \leq i \leq k)$ or $(a_i : i \in \{1, \ldots, k\})$. A common, but less formal way to denote a list is $(a_1, \ldots, a_k)$. Sometimes the parentheses are omitted and we write simply $a_1, \ldots, a_k$ (although we won't do this in these notes.)

Specification of lists. We can specify a list by writing it out as above. However, for long lists this is impractical. Most lists we study have a pattern where the ith entry can be determined by a simple rule that depends on i. For example, the list $a = (1, 4, 8, 9, 16, 25)$ is the list of length 5 whose entries are given by the rule $a_i = i^2$.

Equality of lists. Two lists v and w are considered to be equal if they have the same length, and for each j between 1 and $\mathbf{length}(v)$ we have $v_j = w_j$ (so that the entries in matching positions are the same).

Combining lists There is a natural operation for combining two lists into one list called *concatenation*; the concatenation of list a wth list b, denoted $a * b$, is obtained by appending the items of b to the items of a. The length of $a * b$ is the sum $\mathbf{length}(a) + \mathbf{length}(b)$. The entries of $a * b$ satisfy:

$$(a * b)_i = \begin{cases} a_i & \text{if } i \leq \mathbf{length}(a) \\ b_{i- \mathbf{length}(a)} & \text{if } \mathbf{length}(a) < i \leq \mathbf{length}(a) + \mathbf{length}(b). \end{cases}$$

For example, if $a = (1, 6, 1)$ and $b = (2, 3)$ then $a * b = (1, 6, 1, 2, 3)$.

Comparing lists If a and b are lists we say that a is a *prefix* of b, denoted $a <_{pre} b$ provided that $\mathbf{length}(a) \leq \mathbf{length}(b)$ and for each $i \in \{1, \ldots, \mathbf{length}(a)\}$, $a_i = b_i$. For example, $(1, 3, 5)$ is a prefix of $(1, 3, 5, 2, 4)$. Another way to express this definition is that a is a prefix of b provided that there is a list c such that $b = a \circ c$.

Sets of lists We now introduce an important construction, which allows us to describe certain sets whose members are lists.

If A is any set, we write A^* for the set of all lists with entries coming form A. If k is an integer, we write A^k for the set of all possible lists of length k whose entries are from the set A For example, as you probably learned in previous courses, $\mathbb{R}^3$ is the set of lists (vectors) of length 3 with real number entries, and $\mathbb{R}^*$ is the set of lists of real numbers of all possible (finite) lengths.

Remark 2.4. Notice that the meaning of A^k depends on the type of object that is represented by A. If A is a number, then A^k means the number obtained by multiplying k A's together. If A is a set then A^k is a set of lists of length k This is an example of a common feature of mathematical terminology: the meaning of notation depends on the types of objects involved This is one of many reasons why in a mathematical discussion, it is crucial to be aware of the types of each object under consideration. There is an interesting connection between the notation A^k when A is a set and A^k when A is a number If A is a finite set, then A^k is also a finite set (whose members are lists of length k) It turns out that the size of the set A^k, denoted $|A^k|$, is equal to $|A|^k$, which is the size of A raised to the k power.

Lists of size 2 are called *ordered pairs*. If A and B are sets, then $A \times B$, the *product of A and B* denotes the set of all ordered pairs (a, b) such that $a \in A$ and $b \in B$. Thus, $A \times A$ is the same as A^2.

More generally, if $A_1, A_2, \ldots, A_k$ is a list of k sets then the*product of* $A_1, \ldots, A_k$, denoted

$$\prod_{i=1}^{k} A_i$$

is the set of lists $(a_1, \ldots, a_k)$ of length k such that $a_i \in A_i$ for each $i \in \{1, \ldots, k\}$. For example if A_i is the set $\{1, i\}$ then $\prod_{i=1}^{4} A_i$ includes the lists $(1, 2, 3, 4)$ and $(1, 1, 1, 1)$ but not $(4, 3, 2, 1)$.

If $A_1, \ldots, A_k$ are all the same set A, then $\prod_{i=1}^{k} A_i$ is equal to A^k.

Remark 2.5. Mathematicians often use the notation $A_1 \times A_2 \times A_3$ instead of $\prod_{i=1}^{3} A_i$. However, the reader should be aware that this notation, while common, creates some possible difficulties. If we write $C = A_1 \times A_2 \times A_3$, we might think that this is equal to $D = (A_1 \times A_2) \times A_3$ and also to $E = A_1 \times (A_2 \times A_3)$. Technically, though, these three sets are different! The members of C are lists of length 3. The members of D are ordered pairs whose first entry is an ordered pair in $A_1 \times A_2$, and whose third coordinate is in A_3. Similarly E consists of ordered pairs, whose first entry belongs to A_1 and whose second entry belongs to $A_2 \times A_3$. For example, suppose that $A_1 = A_2 = A_3 = \mathbb{Z}$. Then $(1, 2, 3) \in C$, $((1, 2), 3) \in D$ and $(1, (2, 3)) \in E$.

Even though the lists $(1, 2, 3)$, $((1, 2), 3)$ and $(1, (2, 3))$ are different, for some purposes we can treat them as the same. It is common in mathematics that for some purposes we want to treat certain objects as the same. This is formally accomplished by the idea of an *equivalence relation*. In a given context, we can declare that we will be treating certain groups of objects as the same, provided that we precisely define which objects will be treated as the same. In the above example, we can define two lists as equivalent if they are the same if you eliminate all internal parentheses. The notion of equivalence relation is fundamental to mathematics, and we will deal with it in detail later.

Nested types

We have now discussed three types: numbers, and the two "collection" types, set and list. All of the examples of sets and lists we've seen are sets of numbers or lists of number. Now we are in position to give examples of sets with more complicated members and lists with more complex entries. For example:

- $a = \{\{1, 2\}, \{2, 3\}, \{3, 5\}\}$
- $b = \{\{3, 2\}, \{1, 2\}, \{5, 3\}\}$
- $c = (\{1\}, \{1, 2\}, \{1, 2, 3\}, \{1, 2, 3, 4\})$
- $d = (\{1\}, \{2, 1\}, \{3, 2, 1\}, \{4, 3, 2, 1\})$
- $e = (\{1, 2, 3, 4\}, \{1, 2, 3\}, \{1, 2\}, \{1\})$
- $f = \{\emptyset\}$
- $g = \{\}$
- $h = (\emptyset)$
- $j = \{(\{1, 2, 3\}, \{1, 3\}), (\{2, 4\}, \{\}), (\{1, 3, 5, 6\}, \{2, 4, 6, 8\})$

The object a is a set with three members. Each member of a is itself a set of numbers having two members. The object b is also a set with three members, with each member being a set of numbers having two members. In fact we have $a = b$! This is because each member of a is a member b and vice versa.

Objects c,d and e are lists, each having four entries, where each entry is a set. We have $c = d$ because they have the same length, and $c_j = d_j$ for j between 1 and 4. However, $c \neq e$ since, for example, $c_1 \neq e_1$.

Object f and g are sets. g is the empty set and has size 0. Note that f is not equal to g and is not the empty set. Instead f is a set having exactly one member, and that member is the empty set. Object h is the list that has one entry (the empty set) and is not the same object as f because h is a list and f is a set. Object j is a set, whose members are lists of size 2 (order pairs) where the entries of each list are sets of integers.

Indexed families

We now consider a generalization of lists called *indexed families*. If d is a list of length 5 whose entries belong to set S then the entries d_1, d_2, d_3,d_4, d_5 are each members of S, The subscripts 1,2,3,4, and 5 serve as reference labels. These reference labels are called *indices*, and one can think of each of index as pointing to the appropriate item on the list. The set of indices, $\{1, 2, 3, 4, 5\}$ is called the *index set* of the list.

In a list the index set is always of the form $\{1, \ldots, m\}$ where m is the length of the list. When we think of a list in this way, it is natural to consider the possibility of using a different index set for the collection. If we call our index set J, then for each $j \in J$ we have an object d_j. If all of the objects in the collection are of some type T, we call this collection an *indexed family* of objects from T with index set J. We denote such a collection using the notation $d = (d_j : j \in J)$, which means that d has one entry for each index j belonging to J.

Example 2.1. An indexed family of objects with index set $\mathbb{Z}_{>0}$ or $\mathbb{Z}_{\geq 0}$ is called a *sequence*. For example $(j^2 : j \in \mathbb{Z}_{\geq 0})$ is the sequence of positive perfect squares 0,1,4,9,16, ..., and $(\{1, \ldots, j\} : j \in \mathbb{Z}_{\geq 0}$ is the sequence of sets $\emptyset, \{1\}, \{1, 2\}, \{1, 2, 3\}, \ldots$.

Example 2.2. Suppose we take an indexed collection of numbers m with the index set the set $J = \{1, 2, 3\} \times \{1, 2, 3\}$ which (as discussed earlier) is the set of ordered pairs (i, j) with both entries in $\{1, 2, 3\}$. For example, our indexed collection might be given by $m_{(1,1)} = 7, m_{(1,2)} = 4, m_{(1,3)} = 1, m_{(2,1)} = 8, m_{(2,2)} = 5, m_{(2,3)} = 2, m_{(3,1)} = 9, m_{(3,2)} = 6, m_{3,3)} = 3$. Because the index set is ordered pairs, it is natural to visualize this collection by arranging the entries in a 3 by 3 table:

i/j	1	2	2
1	7	4	1
2	8	5	2
3	9	6	3

Thus an indexed collection with index set $\{1, 2, 3\} \times \{1, 2, 3\}$ is naturally thought of as a *3 by 3 matrix*. In general an indexed collection with index set $\{1, \ldots, m\} \times \{1, \ldots, n\}$ is called an m by n matrix.

We can take any other index set. Another index set that arises commonly is the index set $\mathbb{Z}_{>0}$ of positive integers. An indexed collection with this index set has the form $a = (a_j : j \in \mathbb{Z}_{>0})$

and is called a *sequence*, which can be thought of as a list that doesn't end. Sequences of real numbers are commonly encountered, for example, in calculus.

Remark 2.6. For sequences, it is common to use the less formal notation $a_1, a_2, \ldots$, in place of the $(a_j : j \in \mathbb{Z}_{>0})$.

Specification To specify an indexed family b, we specify the index set J. amd for $j \in J$, we specify an object b_j, which is the object in family b indexed by $j \in J$. As with lists, it is common to provide a rule so that for each $j \in J$,the rulle specifies the entry b_j.

Normally, we are interested in indexed families in which all of the objects are of some common type T. In this case we refer to b as an indexed family of objects from T. Indexed collections are commonly used in mathematics where the objects in the collection are sets. For example $\mathcal{A} = ([a, \infty) : a \in \mathbb{R})$ is an indexed collection of subsets of the reals. The index set is the set of real numbers, and the number a indexes the interval $[a, \infty)$.

Equality Two indexed families a and b are the same provided that they have the same index set , and for each index i in the index set, $a_i = b_i$.

Reindexing an indexed family Given an indexed family $a = (a_j : j \in J)$ it is sometimes convenient to change the index set as follows. Let J' be a set such that there is a one-to-one correspondence f from I and J. We can associate a to an indexed family $b = (b_i : i \in I)$, given for $i \in I$ by $b_i = a_{f(i)}$. We say that b is a reindexing of a. For most purposes, we often think of b as the same as a; technically they are equivalent under an appropriate notion of equivalence.

Combining Indexed famlies The operation "concatenation" which was introduced for lists, can be extended to indexed families.. Suppose $a = (a_j : j \in I)$ and $b = (b_j : j \in J)$ are indexed families where I and J are two disjoint index sets. Then $a * b$ is the indexed family with index set $I \cup J$, given by the rule that for $i \in I \cup J$, $a * b(i) = a(i)$ if $i \in I$ and is $b(i)$ if $i \in J$.

Advanced remark 2.7. If the index sets I and J are not disjoint, then before combining the two indexed families, we reindex them so that the sets are disjoint. A common way to do this is to change the index set of I to the set of ordered pairs $\{(i, 1) : i \in I\}$ and change the index set J to the set of ordered pairs $\{(i, 2) : i \in J\}$.

Given sets S and J we can form the set of all indexed families of objects from S using index set J. The notation for this is S^J. Here we are using "exponential" notation, but where the objects S and J are sets not numbers.

Remark 2.8. If J is the set $\{1, \ldots, k\}$ then $S^{\{1,\ldots,k\}}$ is a list of length k with entries in S. Using the notation for lists introduced earlier, we have that $S_{\{1,\ldots,k\}}$ and S^k mean the same thing.

In general, if S and J are finite sets then $|S^J|$, which counts the number of indexed families from S indexed by J is equal to $|S|^{|J|}$. (Why?)

Functions

The final group of basic objects are mathematical objects that represent a correspondence between two sets of objects. These are called *functions* and are at the heart of modern mathematics. Most students start this course with some familiarity with functions, and view functions as formulas such as $f(x) = x^2 - 2x + 1$ or as curves drawn in the x-y plane. These examples and pictures are useful, but may give a misleading picture of what kind of object a function is.

The idea of a function is of a small "computer" that can receive certain mathematical objects as input, and for each input object it might receive, produces an output that depends on the input. The function is associated with two unique sets, the *domain* of f, which is the set of allowed inputs to the function and the *target* of f, which is a set that contains all possible output values and possibly some members that are not output values. When the function f is given an input x from the domain, the function produces an output from the target. Each time you give f the same input x you get the same output. The outputs that f gives for two different inputs x and z may be the same or different.

Figure 1 illustrates the idea of a function.

Figure 1: A pictorial view of a function

Let's express this idea a bit more carefully. A function is a mathematical object g with the following characteristics:

1. Associated to g are two sets, the *domain* of g, denoted $\mathbf{Dom}(g)$, and the *target* of g, denoted $\mathbf{Target}(g)$.

2. To each permissible input $x \in \mathbf{Dom}(g)$, g associates a mathematical object denoted $g(x)$ (which is read "g of x"). $g(x)$ is called the *image of g on x* and is the output of the function g when x is input.

The notation $g : S \longrightarrow T$ means that S and T are sets and g is a function with domain S and target T. The set of all functions with domain S and target T is denoted by T^S.

1. If $S \subseteq \mathbf{Dom}(g)$, the *image of S under g*, denoted $\mathbf{Im}_g(S)$ is the set consisting of all elements of the target. Using the notation for specifying sets, we have $\mathbf{Im}_g(S) = \{g(x) : xinS\}$, which

2. The image of the entire domain of f, $\mathbf{Im}_f(\mathbf{Dom}(f))$ is called the *range* of f, and is denoted $\mathbf{Range}(f)$. Thus the range consists of those members of the target that are the output of f for at least one input from the domain.

3. If $T \subseteq \mathbf{Target}(g)$, the *inverse image of T under g* is denoted $\mathbf{PreIm}_g(T)$ is the set of all $x \in \mathbf{Dom}(g)$ such that $g(x) \in T$. If T consists of a single object y then we often write $\mathbf{PreIm}_g(y)$ instead of $\mathbf{PreIm}_g(\{y\})$.

Advanced remark 2.9. In many books, you'll see a different notation for image and inverse image. It is common to write $g(S)$ instead of $\mathbf{Im}_g(S)$, and to write $g^{-1}(T)$ instead of $\mathbf{PreIm}_g(T)$. This notation has a potential for confusion. When we write $g(x)$, x should be member of the domain of g, but S is not a member of the domain, but a subset of the domain. Also, the notation g^{-1} has other meanings, as we'll see later, and to avoid confusion, we don't use the notaiton in this situation.

This is an example where experienced mathematicians safely use notation that has multiple meanings because they have the experience to separate the different meanings, and know which applies in a given context. Since we're starting out, we avoid these ambiguities.

For most of the functions you've encountered in the past, the domain and target sets are both $\mathbb{R}$, or subsets of $\mathbb{R}$. In fact, a function is allowed to have any set as its domain, and any (nonempty) set as its target. Throughout these notes, we'll see functions with a variety of different domains.

Specifying a function. To fully describe a function g, you need to specify the set $\mathbf{Dom}(g)$ and the set $\mathbf{Target}(g)$, and for each $x \in \mathbf{Dom}(g)$ specify what $g(x)$. If the domain of g is a finite set we can simply describe the domain by listing its elements, and listing a function table.

Example 2.3. Let h be the function with $\mathbf{Dom}(h) = \{1, 2, 3\}$ given by the table:

x	1	2	3	4	5
$h(x)$	3	2	2	1	3

There are various ways to visualize a function. In figure 2, we present a *domain-target diagram*. Here the elements of the domain are pictured on the left, and the output elements are pictured on the right. There is an arrow from each domain member to its corresponding output member.

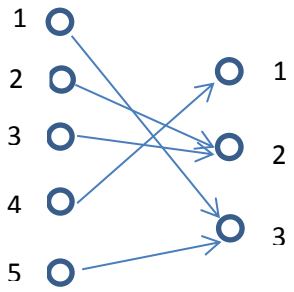

Figure 2: Domain-Target diagram of function h from Example 2.3

When the domain and target are subsets of the real numbers, we can illustrate the function using the familiar *function graph in the x-y plane* as in Figure 3. For each point plotted in the graph, the x-coordinate is a domain value, and the y coordinate is the function value.

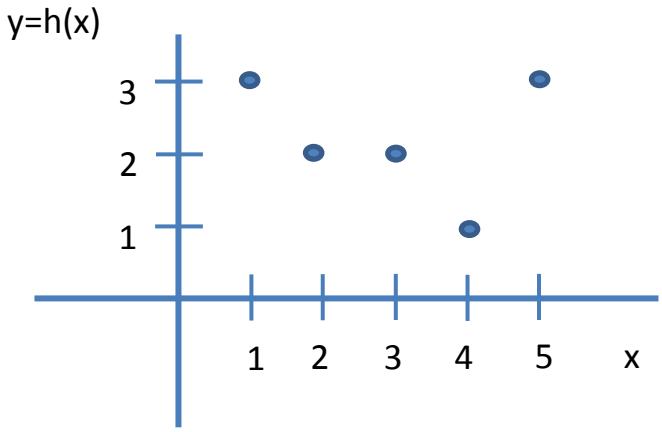

Figure 3: A graph of function h from Example 2.3

If the domain is infinite then we can't describe the function by a table, so we need other ways to specify a function. The most common way of specifying a function is to give the

domain, together with a rule that precisely describes how the output object is built from the input object. Usually the rule is described by giving an *expression* involving the input.

Example 2.4. Let f be the function with domain $\mathbb{R}$ given by the rule $f(x) = x^2$ for all $x \in \mathbb{R}$.

Here's a simple but important function.

Example 2.5. Let D be an arbitrary set. The *identity function* on D is the function $\mathbf{id}_D$ with domain D given by the rule $\mathbf{id}_D(x) = x$ for all $x \in D$.

The rule that specifies a function may be complicated.

Example 2.6. Let h be the function defined on $\mathbb{Z}$ by the rule:

$$h(x) = \begin{cases} x/2 & \text{if } x \text{ is even} \\ 3x+1 & \text{if } x \text{ is odd} \end{cases}$$

A rule of this form is expressed by dividing the domain into pieces. Here we divide the set of integers into the set of even integers and the set of odd integers.

Example 2.7. Here's a function that maps a number to a *set of numbers*. Let **Div** be the function with domain $\mathbb{Z}_{>0}$ given by the rule $\mathbf{Div}(n)$ is the set of positive integer divisors of n. Thus $\mathbf{Div}(12) = \{1, 2, 3, 4, 6, 12\}$ and $\mathbf{Div}(0) = \mathbb{Z}_{>0}$.

We can take our target set to be $\mathcal{P}_{\mathbf{fin}}(\mathbb{Z})$ and write $\mathbf{Div} : \mathbb{Z} \longrightarrow \mathcal{P}_{\mathbf{fin}}(\mathbb{Z})$, since every output value is a finite set of integers.

Example 2.8. Consider the function m whose domain is the set of finite subsets of real numbers given by the rule that for any set S, $m(S)$ is the least member of S. For example, $m(\{17, -\pi, 1, 100.25\}) = -\pi$. This function has domain $\mathcal{P}_{\mathbf{fin}}(\mathbb{R})$ and target set $\mathbb{R}$.

Example 2.9. Consider the function w whose domain is $\mathbb{R}^3$ and target is $\mathbb{R}^2$ given by the rule $f(x, y, z) = (xyz, x + y + z)$ for all $(x, y, z) \in \mathbb{R}^3$.

Remark 2.10. This is an example in which the f takes 3 inputs. However, we think of the three inputs as a single list, so the function takes a single input (x, y, z).

Strictly speaking, the correct notation is $f((x, y, z))$ rather than $f(x, y, z)$, because the notation is $f(\text{“input object”})$ and the input object is the list denoted (x, y, z) rather than x, y, z. However, it is customary to simplify the notation in this case and use only one set of parentheses.

When are two functions equal? The criterion for two functions f and g to be equal is that they have the same domain, and that for each member x of the domain $f(x) = g(x)$.

Well-defined rules for functions. A proper description of f by a rule should clearly state the domain of f, and provide clear instructions that given x allows one to determine $f(x)$. A rule that does this is said to be *well-defined*. A function rule may fail to be well-defined for two main reasons: (1) The function is undefined for some element of the domain, which means that rule supplied does not make sense for that element, or (2) The rule is ambiguous for some element of the domain, so that for that domain element the rule either produces more than one output value.

Example 2.10. (An invalid function definition.) Let $f : \mathbb{R} \longrightarrow \mathbb{R}$ be defined by $f(x) = x/(x+1)$.

This function definition is invalid because the rule does not make sense for all values of the domain, namely if $x = -1$ then the rule does not produce a real number.

Example 2.11. (An ambiguous function definition.) Let $r : \mathbb{R}_{\geq 0} \longrightarrow \mathtt{R}$ be defined by the rule $r(x)$ is the real number such that $r(x)^2 = x$.

This rule is ambiguous because for each input x, there are two choices of $r(x)$ such that $r(x)^2 = x$.

Natural domain of a rule and partial functions. Example 2.10 is an example where the rule of definition for the function does not make sense for the point $x = -1$ in the domain. Still, we want to make sense of the rule $f(x) = x/(x+1)$ as representing a function. The obvious way to do this is to *restrict the domain* by eliminating the domain elements for which the rule does not make sense. In this example, we would restrict the domain to $\mathbb{R} - \{-1\}$. We then say that f is a *partial function* on $\mathbb{R}$, which means that the domain is a subset of $\mathbb{R}$.

More generally given a rule that makes sense only for some values of the input, we can use the rule to define a function by restricting the domain to those values for which the input makes sense. The resulting domain is called the *natural domain* for the rule.

For example, the rule $f(x) = \sqrt{1 + 1/x}$ makes sense only if the quantity to which square root is applied is nonnegative. This will be true if $x \leq -1$ or if $x >$ − but fails if $x \in (-1, 0]$. So we restrict the domain to the set $(-\infty, -1] \cup [0, \infty)$.

In some cases, we might want to have a function that is defined for all real numbers.

For Example 2.11, the rule is ambiguous for every member of the domain, so removing all of these inputs from the domain will leave a very uninteresting function. In the case of an ambiguous rule, we can try to restrict the target so as to eliminate the ambiguity. In Example 2.11, if we reduce the target set to $\mathbb{R}_{\geq 0}$ then the rule becomes unambiguous, since now for each input x there is one and only one choice of $r(x) \in \mathbb{R}_{\geq 0}$ so that $r(x)^2 = x$.

In the case of a rule that is undefined for some values, we can simply specify a value for the function for those values. For example, the function in Example 2.10 could be modified to:

$$f(x) = \begin{cases} x/(x+1) & \text{if } x \neq -1 \\ 0 & \text{if } x = -1 \end{cases}$$

In this case, our choice to set $f(-1)$ to 0 is arbitrary, which may or may not be okay depending on the context. Later we'll see that in some cases we'll see that there is a natural choice for filling in undefined values.

Example 2.12. (Another invalid function definition.) Suppose we define the function $h : \mathbb{Z} \longrightarrow \mathbb{Z}$ be the function given by the rule $h(n) = n/2$. If we take an odd input, then the output is not an integer and therefore it is outside the specified target set. If our given situation allows for h to output numbers that are not integers, one easy repair is to change the target set to $\mathbb{R}$. If we need the function to output integers, then either we have to restrict the domain to the set of even integers, or we have to modify the function rule for odd integers.

Composition of functions The primary way to combine two functions into one is by *composition*. The idea of function composition is represented in Figure 4, namely the composition of two functions is the function built by chaining the functions together so that the output of one becomes the input of the other.

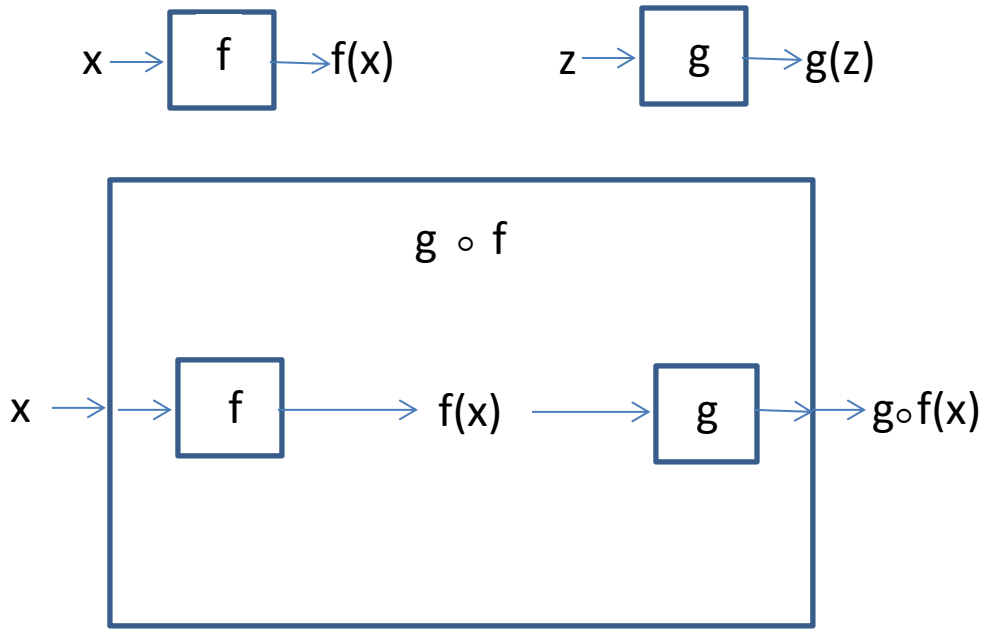

Figure 4: The function $g \circ f$ obtained by composing functions g with f. Notice that when applying $g \circ f$, f is applied first, then g is applied.

The normal situation for composing functions is when we have two functions: $f : S \longrightarrow T$ and $g : U \longrightarrow V$, where the target T of f is a subset of the domain U of g. We define $g \circ f : S \longrightarrow V$ to be the function given by the rule $g \circ f(s) = g(f(s))$ for all $s \in S$. For this rule to make sense we need that $f(s)$ belong to $\mathbf{Dom}(g)$. this rule makes sense for all $s \in S$,

Advanced remark 2.11. **Functions and Indexed collections**

The observant reader may notice a close similarity between functions and indexed collections. A function f associates to each member of the domain an element in the target. An indexed collection C of objects in T with index set J consists of a collection of objects $(C_j : j \in J)$ where each $C_j \in T$.

In fact functions and indexed collections are really just two different ways of representing the same thing. Given a function $f : X \longrightarrow Y$ we can build an indexed collection $(C_x : x \in X)$ of objects in Y with index set x where $C_x = f(x)$, so we can think of the function as an indexed collection. Similarly, if we start with an indexed collection $(C_j : j \in J)$ of objects in T, we can

build from it a function $g : J \longrightarrow T$ defined by $g(j) = C_j$. In particular, a list b of length m can be associated to a function whose domain is $\{1, \ldots, m\}$.

Since functions and indexed families are essentially the same, we use the same notation for the set of functions from X and Y, and the set of indexed families Given that functions and indexed families are essentially the same thing, why do we bother defining both of them? We do this because both concepts appear frequently in the literature (with functions being much more common.) There is a subtle difference in how they are used. When we study indexed families, we are often (though not always) mostly interested in the objects in the families and not in which indices correspond to which objects. The indices are simply a convenient label to reference all of the objects. With functions, we generally care very much about the correspondence between domain elements and target elements.