

Logic

It's really sort of amazing that people manage to communicate in the English language. Here are some typical sentences:

1. "You may have cake or you may have ice cream."
2. "If pigs can fly, then you can understand the Chernoff bound."
3. "If you can solve any problem we come up with, then you get an A for the course."
4. "Every American has a dream."

What *precisely* do these sentences mean? Can you have both cake and ice cream or must you choose just one desert? If the second sentence is true, then is the Chernoff bound incomprehensible? If you can solve some problems we come up with but not all, then do you get an A for the course? And can you still get an A even if you can't solve any of the problems? Does the last sentence imply that all Americans have the same dream or might they each have a different dream?

Some uncertainty is tolerable in normal conversation. But when we need to formulate ideas precisely— as in mathematics— the ambiguities inherent in everyday language become a real problem. We can't hope to make an exact argument if we're not sure exactly what the individual words mean. (And, not to alarm you, but it is *possible* that we'll be making an *awful lot* of exacting mathematical arguments in the weeks ahead.) So before we start into mathematics, we need to investigate the problem of how to talk about mathematics.

To get around the ambiguity of English, mathematicians have devised a special mini-language for talking about logical relationships. This language mostly uses ordinary English words and phrases such as "or", "implies", and "for all". But mathematicians endow these words with definitions more precise than those found in an ordinary dictionary. Without knowing these definitions, you could sort of read this language, but you would miss all the subtleties and sometimes have trouble following along.

Surprisingly, in the midst of learning the language of logic, we'll come across the most important open problem in computer science— a problem whose solution could change the world.

1 Propositions

A *proposition* is a statement that is either true or false. This definition is a little vague, but it does exclude sentences such as, “What’s a surjection, again?” and “Learn logarithms!” Here are some examples of propositions.

“All Greeks are human.”

“All humans are mortal.”

“All Greeks are mortal.”

Archimedes spent a lot of time fussing with such propositions in the 4th century BC while developing an early form of logic. These are perfectly good examples, but we’ll be more concerned with propositions of a more mathematical flavor:

$$“2 + 3 = 5”$$

This proposition happens to be true. Sometimes the truth of a proposition is more difficult to determine:

$$“a^4 + b^4 + c^4 = d^4 \text{ has no solution where } a, b, c \text{ are positive integers.}”$$

Euler made this conjecture in 1769. For 218 years no one knew whether this proposition was true or false. Finally, Noam Elkies who works at the liberal arts school up Mass Ave. found a solution to the equation: $a = 2682440$, $b = 15365639$, $c = 18796760$, $d = 20615673$. So the proposition is false!

There are many famous propositions about primes. Recall that a *prime* is an integer greater than 1 not divisible by any positive integer other than 1 and itself. The first few primes are 2, 3, 5, 7, 11, and 13. On the other hand, 9 and 77 are not primes, because 9 is divisible by 3 and 77 is divisible by 7. The following proposition is called Goldbach’s Conjecture, after Christian Goldbach who first stated it in 1742.

“Every even integer greater than 2 is the sum of two primes.”

Even today, no one knows whether Goldbach’s Conjecture is true or false. Every even integer ever checked is a sum of two primes, but just one exception would disprove the claim.

For the next while, we won’t be much concerned with the internals of propositions—whether they involve mathematics or Greek mortality— but rather with how propositions are combined and related. So we’ll frequently use variables such as P and Q in place of specific propositions such as “All humans are mortal” and “ $2+3 = 5$ ”. The understanding is that these variables, like propositions, can take on only the values “true” and “false”. Such true/false variables are sometimes called *Boolean variables* after their inventor, George something-or-other.

1.1 Combining Propositions

In English, we can modify, combine, and relate propositions with words such as “not”, “and”, “or”, “implies”, and “if-then”. For example, we can combine three propositions into one like this:

If all humans are mortal **and** all Greeks are human, **then** all Greeks are mortal.

1.1.1 “Not”, “And” and “Or”

We can precisely define these special words using *truth tables*. For example, if P denotes an arbitrary proposition, then the validity of the proposition “not P ” is defined by the following truth table:

P	not P
T	F
F	T

The first row of the table indicates that when proposition P is true (T), the proposition “not P ” is false (F). The second line indicates that when P is false, “not P ” is true. This is probably what you would expect.

In general, a truth table indicates the true/false value of a proposition for each possible setting of the variables. For example, the truth table for the proposition “ P and Q ” has four lines, since the two variables can be set in four different ways:

P	Q	P and Q
T	T	T
T	F	F
F	T	F
F	F	F

According to this table, the proposition “ P and Q ” is true only when P and Q are both true. This probably reflects the way you think about the word “and”.

There is a subtlety in the truth table for “ P or Q ”:

P	Q	P or Q
T	T	T
T	F	T
F	T	T
F	F	F

This says that “ P or Q ” is true when P is true, Q is true, or *both* are true. This isn’t always the intended meaning of “or” in everyday speech, but this is the standard definition in mathematical writing. So if a mathematician says, “You may have cake or you may have ice cream”, then you *can* have both.

1.1.2 “Implies”

The least intuitive connecting word is “implies”. Mathematicians regard the propositions “ P implies Q ” and “if P then Q ” as synonymous, so both have the same truth table. (The lines are numbered so we can refer to the them later.)

	P	Q	P implies Q , if P then Q
1.	T	T	T
2.	T	F	F
3.	F	T	T
4.	F	F	T

Let’s experiment with this definition. For example, is the following proposition true or false?

“If Goldbach’s Conjecture is true, then $x^2 \geq 0$ for every real number x .”

Now, we told you before that no one knows whether Goldbach’s Conjecture is true or false. But that doesn’t prevent you from answering the question! This proposition has the form $P \Rightarrow Q$ where P is “Goldbach’s Conjecture is true” and Q is “ $x^2 \geq 0$ for every real number x ”. Since Q is definitely true, we’re on either line 1 or line 3 of the truth table. Either way, the proposition as a whole is *true*!

One of our original examples demonstrates an even stranger side of implications.

“If pigs fly, then you can understand the Chernoff bound.”

Don’t take this as an insult; we just need to figure out whether this proposition is true or false. Curiously, the answer has *nothing* to do with whether or not you can understand the Chernoff bound. Pigs do not fly, so we’re on either line 3 or line 4 of the truth table. In both cases, the proposition is *true*!

In contrast, here’s an example of a false implication:

“If the moon is white, then the moon is made of white cheddar.”

Yes, the moon is white. But, no, the moon is not made of white cheddar cheese. So we’re on line 2 of the truth table, and the proposition is false.

The truth table for implications can be summarized in words as follows:

An implication is true when the if-part is false or the then-part is true.

This sentence is worth remembering; a large fraction of all mathematical statements are of the if-then form!

1.1.3 “If and Only If”

Mathematicians commonly join propositions in one additional way that doesn’t arise in ordinary speech. The proposition “ P if and only if Q ” asserts that P and Q are logically equivalent; that is, either both are true or both are false.

P	Q	P if and only if Q
T	T	T
T	F	F
F	T	F
F	F	T

The following if-and-only-if statement is true for every real number x :

$$“x^2 - 4 \geq 0 \text{ if and only if } |x| \geq 2”$$

For some values of x , *both* inequalities are true. For other values of x , *neither* inequality is true. In every case, however, the proposition as a whole is true.

The phrase “if and only if” comes up so often that it is often abbreviated “iff”.

1.2 Propositional Logic in Computer Programs

Propositions and logical connectives arise all the time in computer programs. For example, consider the following snippet, which could be either C, C++, or Java:

```
if ( x > 0 || (x <= 0 && y > 100) )
    (further instructions)
```

The symbol `||` denotes “or”, and the symbol `&&` denotes “and”. The *further instructions* are carried out only if the proposition following the word `if` is true. On closer inspection, this big expression is built from two simpler propositions. Let A be the proposition that $x > 0$, and let B be the proposition that $y > 100$. Then we can rewrite the condition this way:

$$A \text{ or } ((\text{not } A) \text{ and } B)$$

A truth table reveals that this complicated expression is logically equivalent to “ A or B ”.

A	B	A or $((\text{not } A) \text{ and } B)$	A or B
T	T	T	T
T	F	T	T
F	T	T	T
F	F	F	F

This means that we can simplify the code snippet without changing the program’s behavior:

```
if ( x > 0 || y > 100 )
    (further instructions)
```

Rewriting a logical expression involving many variables in the simplest form is both difficult and important. Simplifying expressions in software might slightly increase the speed of your program. But, more significantly, chip designers face essentially the same challenge. However, instead of minimizing `&&` and `||` symbols in a program, their job is to minimize the number of analogous physical devices on a chip. The payoff is potentially enormous: a chip with fewer devices is smaller, consumes less power, has a lower defect rate, and is cheaper to manufacture.

1.3 A Cryptic Notation

Programming languages use symbols like `&&` and `!` in place of words like “and” and “not”. Mathematicians have devised their own cryptic symbols to represent these words, which are summarized in the table below.

English	Cryptic Notation
“not P ”	$\neg P$ or \overline{P}
“ P and Q ”	$P \wedge Q$
“ P or Q ”	$P \vee Q$
“ P implies Q ” or “if P then Q ”	$P \Rightarrow Q$
“ P if and only if Q ”	$P \Leftrightarrow Q$

For example, using this notation, “If P and not Q , then R ” would be written:

$$(P \wedge \neg Q) \Rightarrow R$$

This symbolic language is helpful for writing complicated logical expressions compactly. But in most contexts ordinary words such as “or” and “implies” are much easier to understand than symbols such as \vee and \Rightarrow . So we’ll use this symbolic language sparingly, and we advise you to do the same.

1.4 Logically Equivalent Implications

Are these two sentences saying the same thing?

If I am hungry, then I am grumpy.
If I am not grumpy, then I am not hungry.

We can settle the issue by recasting both sentences in terms of propositional logic. Let P be the proposition "I am hungry", and let Q be "I am grumpy". The first sentence says " P implies Q " and the second says " $(\text{not } Q)$ implies $(\text{not } P)$ ". We can compare these two statements in a truth table:

P	Q	P implies Q	$(\text{not } Q)$ implies $(\text{not } P)$
T	T	T	T
T	F	F	F
F	T	T	T
F	F	T	T

Sure enough, the two statements are precisely equivalent. In general, " $(\text{not } Q)$ implies $(\text{not } P)$ " is called the *contrapositive* of " P implies Q ". And, as we've just shown, the two are just different ways of saying the same thing. This equivalence is mildly useful in programming, mathematical arguments, and even everyday speech, because you can always pick whichever of the two is easier to say or write.

In contrast, the *converse* of " P implies Q " is the statement " Q implies P ". In terms of our example, the converse is:

If I am grumpy, then I am hungry.

This sounds like a rather different contention, and a truth table confirms this suspicion:

P	Q	P implies Q	Q implies P
T	T	T	T
T	F	F	T
F	T	T	F
F	F	T	T

Thus, an implication *is* logically equivalent to its contrapositive, but is *not* equivalent to its converse.

One final relationship: an implication and its converse together are equivalent to an if and only if statement. Specifically, these two statements together:

If I am grumpy, then I am hungry.
If I am hungry, then I am grumpy.

are equivalent to the single statement:

I am grumpy if and only if I am hungry.

Once again, we can verify this with a truth table:

P	Q	$(P$ implies $Q)$ and $(Q$ implies $P)$	Q if and only if P
T	T	T	T
T	F	F	F
F	T	F	F
F	F	T	T

SAT

A proposition is **satisfiable** if some setting of the variables makes the proposition true. For example, $P \wedge \neg Q$ is satisfiable because the expression is true when P is true and Q is false. On the other hand, $P \wedge \neg P$ is not satisfiable because the expression as a whole is false for both settings of P . But determining whether or not a more complicated proposition is satisfiable is not so easy. How about this one?

$$(P \vee Q \vee R) \wedge (\neg P \vee \neg Q) \wedge (\neg P \vee \neg R) \wedge (\neg R \vee \neg Q)$$

The general problem of deciding whether a proposition is satisfiable is called *SAT*. One approach to SAT is to construct a truth table and check whether or not a T ever appears. But this approach is not very efficient; a proposition with n variables has a truth table with 2^n lines. For a proposition with just 30 variables, that's already over a billion!

Is there an *efficient* solution to SAT? Is there some ingenious procedure that *quickly* determines whether any given proposition is satisfiable or not? No one knows. And an awful lot hangs on the answer. An efficient solution to SAT would immediately imply efficient solutions to many, many other important problems involving packing, scheduling, routing, and circuit verification. This sounds fantastic, but there would also be worldwide chaos. Decrypting coded messages would also become an easy task (for most codes). Online financial transactions would be insecure and secret communications could be read by everyone.

At present, though, researchers are completely stuck. No one has a good idea how to either solve SAT more efficiently or to prove that no efficient solution exists. This is the outstanding unanswered question in computer science.

2 Sets and Sequences

Propositions of the sort we've considered so far are good for reasoning about individual statements, but not so good for reasoning about a collection of objects. Let's first review a couple mathematical tools for grouping objects and then extend our logical language to cope with such collections.

A *set* is a collection of distinct objects, which are called *elements*. The elements of a set can be just about anything: numbers, points in space, or even other sets. The conventional way to write down a set is to list the elements inside curly-braces. For example, here are

some sets:

\mathbb{N}	$= \{0, 1, 2, 3, \dots\}$	the natural numbers
C	$= \{\text{red, blue, yellow}\}$	primary colors
D	$= \{\text{Nifty, Friend, Horatio, Pretty-Pretty}\}$	dead pets
P	$= \{\{a, b\}, \{a, c\}, \{b, c\}\}$	a set of sets

The elements of a set are required to be distinct; a set can not contain multiple copies of the same element. The order of elements is not significant, so $\{x, y\}$ and $\{y, x\}$ are the same set written two different ways. The expression $e \in S$ asserts that e is an element of set S . For example, $7 \in \mathbb{N}$ and $\text{blue} \in C$, but $\text{Wilbur} \notin D$ yet.

Sets are simple, flexible, and everywhere. You'll find at least one set mentioned on almost every page in these notes.

2.1 Some Popular Sets

Mathematicians have devised special symbols to represent some common sets.

symbol	set	elements
\emptyset	the empty set	none
\mathbb{N}	natural numbers	$\{0, 1, 2, 3, \dots\}$
\mathbb{Z}	integers	$\{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$
\mathbb{Q}	rational numbers	$\frac{1}{2}, -\frac{5}{3}, 16, \text{ etc.}$
\mathbb{R}	real numbers	$\pi, e, -9, \text{ etc.}$
\mathbb{C}	complex numbers	$i, \frac{19}{2}, \sqrt{2} - 2i, \text{ etc.}$

A superscript $+$ restricts a set to its positive elements; for example, \mathbb{R}^+ is the set of positive real numbers.

2.2 Comparing and Combining Sets

The expression $S \subseteq T$ indicates that set S is a **subset** of set T , which means that every element of S is also an element of T . For example, $\mathbb{N} \subseteq \mathbb{Z}$ (every natural number is an integer) and $\mathbb{Q} \subseteq \mathbb{R}$ (every rational number is a real number), but $\mathbb{C} \not\subseteq \mathbb{Z}$ (not every complex number is an integer).

As a memory trick, notice that the \subseteq points to the smaller set, just like a \leq sign points to the smaller number. Actually, this connection goes a little further: there is a symbol \subset analogous to $<$. Thus, $S \subset T$ means that S is a subset of T , but the two are not equal. So for every set A , $A \subseteq A$, but $A \not\subset A$.

There are several ways to combine sets. Let's define a couple for use in examples:

$$X = \{1, 2, 3\}$$

$$Y = \{2, 3, 4\}$$

- The **union** of sets X and Y (denoted $X \cup Y$) contains all elements appearing in X or Y or both. Thus, $X \cup Y = \{1, 2, 3, 4\}$.
- The **intersection** of X and Y (denoted $X \cap Y$) consists of all elements that appear in *both* X and Y . So $X \cap Y = \{2, 3\}$.
- The **difference** of X and Y (denoted $X - Y$) consists of all elements that are in X , but not in Y . Therefore, $X - Y = \{1\}$ and $Y - X = \{4\}$.

Sometimes one is exclusively concerned with subsets of a certain large set. For example, we might be focused on subsets of the real numbers. In this case, we can talk about the **complement** of a set Z , which consists of all elements *not* in Z and is denoted \bar{Z} . For example, when we're working with the real numbers \mathbb{R} , the complement of the positive real numbers is the set of negative real numbers together with zero.

2.3 Sequences

Sets provide one way to group a collection of objects. Another way is in a *sequence*, which is a list of objects called *terms* or *components*. Short sequences are commonly described by listing the elements between parentheses; for example, (a, b, c) is a sequence with three terms.

While both sets and sequences perform a gathering role, there are several differences.

- The elements of a set are required to be distinct, but terms in a sequence can be the same. Thus, (a, b, a) is a valid sequence, but $\{a, b, a\}$ is not a valid set.
- The terms in a sequence have a specified order, but the elements of a set do not. For example, (a, b, c) and (a, c, b) are different sequences, but $\{a, b, c\}$ and $\{a, c, b\}$ are the same set.
- The empty set is usually denoted \emptyset , and the empty sequence is typically λ .

The product operation is one link between sets and sequences. A *product* of sets, $S_1 \times S_2 \times \dots \times S_n$, is a new set consisting of all sequences where the first component is drawn from S_1 , the second from S_2 , and so forth. For example, $\mathbb{N} \times \mathbb{N}$ is the set of all pairs of natural numbers:

$$\mathbb{N} \times \mathbb{N} = \{(0, 0), (0, 1), (1, 0), (0, 2), (1, 1), (2, 0), \dots\}$$

A product of n copies of a set S is denoted S^n . For example, $\{0, 1\}^3$ is the set of all 3-bit sequences:

$$\{0, 1\}^3 = \{(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1), (1, 1, 0), (1, 1, 1)\}$$

3 Predicates

A *predicate* is a proposition whose truth depends on the value of one or more variables. For example,

“ n is a perfect square”

is a predicate whose truth depends on the value of n . The predicate is true for $n = 4$ since 4 is a perfect square, but false for $n = 5$ since 5 is not a perfect square.

Like other propositions, predicates are often named with a letter. Furthermore, a function-like notation is used to denote a predicate supplied with specific variable values. For example, we might name our earlier predicate P :

$P(n) = \text{“}n \text{ is a perfect square”}$

Now $P(4)$ is true, and $P(5)$ is false.

This notation for predicates is confusingly similar to ordinary function notation. If P is a predicate, then $P(n)$ is either *true* or *false*, depending on the value of n . On the other hand, if P is an ordinary function, like $n^2 + 1$, then $P(n)$ is a *numerical quantity*. Students frequently confuse these two, but that’s what thumbscrews are for.

3.1 Quantifying a Predicate

There are a couple kinds of assertion one commonly makes about a predicate: that it is *sometimes* true and that it is *always* true. For example, the predicate

$x^2 \geq 0$

is always true when x is a real number. On the other hand, the predicate

$5x^2 - 7 = 0$

is only sometimes true; specifically, when $x = \pm\sqrt{7/5}$.

There are several ways to express the notions of “always true” and “sometimes true” in English. The table below gives some general formats on the left and specific examples using those formats on the right. You can expect to see such phrases hundreds of times in mathematical writing!

Always True

“For all n , $P(n)$ is true.”
“ $P(n)$ is true for every n .”

“For all x , $x^2 \geq 0$.
“ $x^2 \geq 0$ for every x .”

Sometimes True

<p>“There exists an n such that $P(n)$ is true.” “$P(n)$ is true for some n.” “$P(n)$ is true for at least one n.”</p>	<p>“There exists an x such that $5x^2 - 7 = 0$.” “$5x^2 - 7 = 0$ for some x.” “$5x^2 - 7 = 0$ for at least one x.”</p>
--	--

All these sentences quantify how often the predicate is true. Specifically, an assertion that a predicate is always true is called a *universal* quantification, and an assertion that a predicate is sometimes true is an *existential* quantification. Sometimes the English sentences are unclear with respect to quantification:

“If you can solve any problem we come up with, then you get an A for the course.”

The phrase “you can solve any problem we can come up with” could reasonably be interpreted as either a universal or existential quantification:

“you can solve *every* problem we come up with”
 “you can solve *at least one* problem we come up with”

In any case, notice that this quantified phrase appears inside a larger if-then statement. This is quite normal; quantified statements are themselves propositions and can be combined with and, or, implies, etc. just like any other proposition.

3.2 More Cryptic Notation

There are symbols to represent universal and existential quantification, just as there are symbols for “and” (\wedge), “implies” (\Rightarrow), and so forth. In particular, to say that a predicate $P(n)$ is true for all values of n in some set S , one writes:

$$\forall n \in S P(n)$$

The symbol \forall is read “for all”, so this whole expression is read “for all n in S , $P(n)$ is true”. To say that a predicate $P(n)$ is true for at least one value of n in S , one writes:

$$\exists n \in S P(n)$$

The backward-E is read “there exists”. So this expression would be read, “There exists an n in S such that $P(n)$ is true. The symbols \forall and \exists are always followed by a variable and then a predicate, as in the two examples above.

As an example, let P be the set of problems we come up with, $S(x)$ be the predicate “You can solve problem x ”, and A be the proposition, “You get an A for the course.” Then the two different interpretations of

“If you can solve any problem we come up with,
 then you get an A for the course.”

can be written as follows:

$$(\forall x \in P S(x)) \Rightarrow A \qquad (\exists x \in P S(x)) \Rightarrow A$$

3.3 Mixing Quantifiers

Many mathematical statements involve several quantifiers. For example, Goldbach's Conjecture states:

“Every even integer greater than 2 is the sum of two primes.”

Let's write this more verbosely to make the use of quantification more clear:

“For every even integer n greater than 2,
there exist primes p and q such that $n = p + q$.”

Let E be the set of even integers greater than 2, and let P be the set of primes. Then we can write Goldbach's Conjecture in logic notation as follows:

$$\underbrace{\forall n \in E}_{\substack{\text{for every even} \\ \text{integer } n \geq 2}} \quad \underbrace{\exists p \in P \exists q \in P}_{\substack{\text{there exist primes} \\ p \text{ and } q \text{ such that}}} \quad n = p + q$$

3.4 Order of Quantifiers

Swapping the order of different kinds of quantifiers (existential or universal) changes the meaning of a proposition. For another example, let's return to one of our initial, confusing statements:

“Every American has a dream.”

This sentence is ambiguous because the order of quantifiers is unclear. Let A be the set of Americans, let D be the set of dreams, and define the predicate $H(a, d)$ to be “American a has dream d .”. Now the sentence could mean there is a single dream that every American shares:

$$\exists d \in D \forall a \in A H(a, d)$$

Or it could mean that every American has an individual dream:

$$\forall a \in A \exists d \in D H(a, d)$$

Swapping quantifiers in Goldbach's Conjecture creates a patently false statement:

$$\underbrace{\exists p \in P \exists q \in P}_{\substack{\text{there exist primes} \\ p \text{ and } q \text{ such that}}} \quad \underbrace{\forall n \in E}_{\substack{\text{for every even} \\ \text{integer } n \geq 2}} \quad n = p + q$$

3.5 Negating Quantifiers

There is a duality between the two kinds of quantifiers. The following two sentences mean the same thing:

“It is not the case that everyone likes to snowboard.”
 “There exists someone who does not like to snowboard.”

In terms of logic notation, this follows from a general equivalence:

$$\neg \forall x P(x) \quad \Leftrightarrow \quad \exists x \neg P(x)$$

Similarly, these sentences mean the same thing:

“There does not exist anyone who likes skiing over magma.”
 “Everyone dislikes skiing over magma.”

We can express the equivalence in logic notation this way:

$$\neg \exists x Q(x) \quad \Leftrightarrow \quad \forall x \neg Q(x)$$

The general principle is that *moving a “not” across a quantifier changes the kind of quantifier.*

3.6 Set Builder Notation

One specialized, but important use of predicates is in *set builder notation*. We’ll often want to talk about sets that can not be described very well by listing the elements explicitly or by taking unions, intersections, etc. of easily-described sets. Set builder notation often comes to the rescue. The idea is to define a *set* using a *predicate*; in particular, the set consists of all values that make the predicate true. Here are some examples of set builder notation:

$$\begin{aligned} A &= \{n \in \mathbb{N} \mid n \text{ is a prime and } n = 4k + 1 \text{ for some integer } k\} \\ B &= \{x \in \mathbb{R} \mid x^3 - 3x + 1 > 0\} \\ C &= \{a + bi \in \mathbb{C} \mid a^2 + 2b^2 \leq 1\} \end{aligned}$$

The set A consists of all natural numbers n for which the predicate

“ n is a prime and $n = 4k + 1$ for some integer k ”

is true. Thus, the smallest elements of A are:

$$5, 13, 17, 29, 37, 41, 53, 57, 61, 73, \dots$$

Trying to indicate the set A by listing these first few elements wouldn't work very well; even after ten terms, the pattern is not obvious! Similarly, the set B consists of all real numbers x for which the predicate

$$x^3 - 3x + 1 > 0$$

is true. In this case, an explicit description of the set B in terms of intervals would require solving a cubic equation. Finally, set C consists of all complex numbers $a + bi$ such that:

$$a^2 + 2b^2 \leq 1$$

This is an oval-shaped region around the origin in the complex plane.