Math Review

Algorithms

Corazza

A course in Algorithms requires some familiarity with logarithms, properties of sets and operations on them, summation notation, basic number theory and probability theory, and the concept of recursion. It also requires the ability to understand, and carry out, simple arguments using mathematical induction. We review most of these concepts here (basics of probability theory will be introduced in a separate lesson) and we provide some sample problems. Students should study the samples and work as many of the exercises associated with this lesson as necessary.

Laws Of Logarithms

$$\begin{aligned}
y = \log_b x &\text{ means } x = b^y \\
\log x &\text{ means } \log_2 x \\
\log^n x &\text{ means } (\log x)^n \\
\ln x &\text{ means } \log_e x \ (e \approx 2.71828) \\
\log_b(xy) &= \log_b x + \log_b y \\
\log_b(x^y) &= y \log_b x \\
\log_b(\frac{x}{y}) &= \log_b x - \log_b y \\
\log_b x &= \frac{\log_a x}{\log_a b} \\
\log_b 1 &= 0 \\
\log_b b &= 1 \\
\log_b x &< x \text{ (for } b \geq 2 \text{ and } x > 0) \\
\log 1024 &= 10 \\
\ln 2 &\approx .693 \\
\log e &\approx 1.44
\end{aligned}$$

Example. Show that the following is not true in general, for $k > 1$:

$$(\log n)^k = k \log n.$$

Solution. To show this, it is enough to give an example for which the equation does not hold true. Consider $n = 4$ and $k = 3$. Then $(\log n)^k = (\log 4)^3 = 2^3 = 8$, but $k \log n = 3 \log 4 = 3 \cdot 2 = 6$.

Example. Show that the following is not true in general:

$$\log_b(x + y) = \log_b x + \log_b y$$

Solution. To show this, it is enough to give an example for which the equation does not hold true. Consider $b = 2, x = 4, y = 4$. Then $\log_b(x + y) = \log(4 + 4) = \log 8 = 3$, but $\log_b x + \log_b y = \log 4 + \log 4 = 2 + 2 = 4$.

Sets

A. A *set* is a collection of objects (this is only approximately correct!).

- The notation $x \in A$ signifies that x is an element of A.
- *Set notation.* The set containing just the elements 1, 2, 3 is denoted $\{1, 2, 3\}$. Elliptical notation can be used to denote larger sets, such as $\mathbf{N} = \{1, 2, 3, \ldots\}$. Set-builder notation defines a set by specifying properties; for instance:

$$E = \{n \mid n \text{ is a natural number and for some } x,\ n = 2 * x\}.$$

- Two sets are *equal* if and only if they have the same elements. Therefore, duplicate elements are not allowed in a set when viewed as a data structure.

B. B is a *subset* of A, $B \subseteq A$, if every element of B is also an element of A. The empty set, denoted $\emptyset$, is a subset of every set (but is *not* an element of every set!).

C. If A and B are sets, $A \cup B$ ("the union of A and B") consists of all objects that belong to at least one of A and B; and $A \cap B$ ("the intersection of A and B") consist of all objects that belong to both A and B. Example:

$$\{1, 2, 3\} \cup \{2, 3, 4\} = \{1, 2, 3, 4\}$$
$$\{1, 2, 3\} \cap \{2, 3, 4\} = \{2, 3\}$$

D. Suppose each of A, B is a set. Then A, B are *disjoint* if A and B have no element in common (that is, $A \cap B = \emptyset$). Similarly, $A_i (i \in I)$ are disjoint if no two of the sets have an element in common.

E. The *cardinality* or *size* of a set A is denoted $|A|$.

F. The *power set* of a set A, denoted $\mathcal{P}(A)$, is the set whose elements are all the subsets of A. Note: If A has n elements, $\mathcal{P}(A)$ has 2^n elements. That is, a set with n elements has 2^n subsets.

G. If a set A having n elements is totally ordered, then a *permutation* of A is a re-arrangement of the elements of A.

- Example: The following are two of the permuations of $\{1, 2, 3, 4\}$:

$$[1, 2, 4, 3], [4, 3, 2, 1]$$

- The permutation of A that does not re-arrange any of the elements is called the *identity permutation.*
- The number of permutations of an n-element set is $n!$.

Example. Are the following sets equal? Explain.

$$\{1, 1, 2\}, \{1, 2\}, \{2, 1\}.$$

Solution. Yes, they are all equal. $\{1, 1, 2\} = \{1, 2\}$ because these sets have the same elements, namely, 1, 2, and two sets are equal if and only if they have the same elements (duplicates are irrelevant). Similarly, $\{1, 2\} = \{2, 1\}$ because these sets have the same elements (order is irrelevant).

Example. Is the following statement true or false?

$$\{1, \{2, 3\}\} \subseteq \{1, 2, 3, 4, 5, \ldots\}$$

Solution. False. The first set contains an element that is *not* an element of the second set — namely, $\{2, 3\}$.

Example. What is the powerset of the set $\{1, 2\}$?

Solution. $\{\emptyset, \{1\}, \{2\}, \{1, 2\}\}$.

Example. List all the permutations of the set $\{1, 3, 4\}$.

Solution. $[1, 3, 4], [1, 4, 3], [3, 1, 4], [3, 4, 1], [4, 1, 3], [4, 3, 1]$.

Functions

Definition [*function*] Suppose A and B are sets. A function f from A to B, written $f : A \to B$, is a rule that assigns to each element of A a unique element of B. A is called the *domain* of f and is sometimes denoted dom f. The domain of f is often referred to as the set of *input values for* f. B is called the *codomain* of f and is written cod$f$. All *output values* for f lie in B.

Example Let $A = \{1, 2, 3\}$ and $B = \{4, 5, 6\}$. Let's define $f : A \to B$ as follows:

$\underline{A}$	f	$\underline{B}$
1	$\to$	4
2	$\to$	5
3	$\to$	6

In other words, f takes the number 1 to 4, the number 2 to 5, and the number 3 to 6. Here is notation that can be used to state this fact:

$$\begin{gathered} f(1) = 4 \\ f(2) = 5 \\ f(3) = 6 \end{gathered}$$

Example. We define a function g, also having domain A and codomain B (defined in the previous example), as follows:

$$\begin{gathered} g(1) = 4 \\ g(2) = 4 \\ g(3) = 4 \end{gathered}$$

Here g is also a function. In the previous example, the function f was *one-one* — no two elements of the domain were assigned the same value by f. Clearly, g does not have that property; in fact, all elements of the domain A of g are assigned the single value 4.

We formally introduce the concept of *one-one* and *onto* functions, and also the *range* of a function:

Definition Suppose $f : A \to B$ is a function. f is one-one if, whenever x, y are distinct elements of A, the values $f(x)$ and $f(y)$ are also distinct; that is, whenever $x \neq y$, we have $f(x) \neq f(y)$. The *range* of f, denoted ran f, is the set of all output values of f; that is

$$\text{ran } f \ = \{f(x) \mid x \in A\}$$

f is said to be *onto* if the range of f is the same as the codomain of f. Finally, f is called a *1-1 correspondence between A and B* if f is both 1-1 and onto.

Example. Returning to the functions f and g of the previous examples, notice that the range of f is precisely equal to B, so f is onto. On the other hand, the range of g is just the singleton set $\{4\}$, and so g is *not* onto.

Example. Consider the function $f(n) = n^2$, where the domain of f is the set $\mathbf{N}$ of all natural numbers. Is f one-one? What is the range of f? Is f onto?

Solution. First we verify that f is one-one. This is done by showing that if $k \neq m$ are distinct elements of the domain of f (in this case, elements of $\mathbf{N}$) then $f(k) \neq f(m)$. In other words, we need to show that $k^2 \neq m^2$ — but this is obvious since $k \neq m$.

Next we determine the range of f. Notice that each output of f is a square, a number of the form m^2. But the squares are $0, 1, 4, 9, 16, 25, \ldots$. Therefore, the range of f is the set

$$\{0, 1, 4, 9, 16, \ldots\} = \{r \mid \text{for some } m, r = m^2\}$$

Finally, we see that, since many natural numbers are not squares, f is not onto. In particular, notice that 2 is an element of the codomain $\mathbf{N}$ of f that is not one of the output values of f.

Some functions from $\mathbf{N}$ to $\mathbf{N}$ have the convenient property of being *increasing*. This means that as input values increase, output values also increase. More precisely, we have the following definition:

Definition. A function $f : \mathbf{N} \to \mathbf{N}$ is said to be *increasing* if, whenever $m < n$, we have $f(m) < f(n)$. A function $g : \mathbf{N} \to \mathbf{N}$ is said to be *nondecreasing* if, whenever $m < n$, we have $g(m) \leq g(n)$.

Example. Obviously, the identity function $f(n) = n$ is increasing. It is equally easy to see that the function $g(n) = kn$ for any integer $k > 1$ is also increasing. This can be verified by simple algebra: if $m < n$, then multiplying on both sides by k gives us $km < kn$, which establishes that $g(m) < g(n)$.

Example. Show that the function $f(n) = n^2$ is increasing.

Solution. Suppose $m < n$. We must show $m^2 < n^2$. We have

$$\begin{aligned} m < n \;&\Rightarrow\; m \cdot m < n \cdot m \\ &\Rightarrow\; m^2 < n \cdot n = n^2 \end{aligned}$$

Summations

$$\sum_{i=1}^{N} 1 = N$$

$$\sum_{i=1}^{N} i = \frac{N(N+1)}{2}$$

$$\sum_{i=1}^{N} i^2 = \frac{N(N+1)(2N+1)}{6}$$

$$\sum_{i=0}^{N} 2^i = 2^{N+1} - 1$$

$$\sum_{i=0}^{N} a^i = \frac{a^{N+1} - 1}{a - 1}$$

$$\sum_{i=0}^{N} a^i < \frac{1}{1-a} \text{ (whenever } 0 < a < 1\text{)}$$

$$\sum_{i=1}^{N} \frac{1}{i} \approx \ln 2 \, \log N \text{ (the difference between these falls below 0.58 as } N \text{ tends to infinity)}$$

Example. Rewrite the following in terms of the variable N, using the formulas above.

$$\sum_{i=1}^{N} 2i^2 + 3i - 4.$$

Solution.

$$\begin{aligned}
\sum_{i=1}^{N} 2i^2 + 3i - 4 &= 2\sum_{i=1}^{N} i^2 + 3\sum_{i=1}^{N} i + 4\sum_{i=1}^{N} 1 \\
&= 2 \cdot \frac{N(N+1)(2N+1)}{6} + 3 \cdot \frac{N(N+1)}{2} + 4N \\
&= \frac{N}{6} \cdot (4N^2 + 15N + 35) \\
&= \frac{1}{6} \cdot (4N^3 + 15N^2 + 35N)
\end{aligned}$$

Mathematical Induction

Mathematical induction is a technique for proving mathematical results having the general form "for all natural numbers n, . . ." For example, suppose you would like to prove that for all natural numbers $n > 1$, $n^2 > n+1$. You might try a few values for n to see if the statement makes sense. Certainly $2^2 > 2+1, 3^2 > 3+1, 10^2 > 10+1$. These examples suggest that the statement always holds true. But how do we know for sure? It is at least conceivable that for certain very large numbers that we are unlikely to consider, the statement is no longer true. Mathematical induction is a technique for demonstrating that such a formula must hold true for every natural number > 1, without exception.

The intuitive idea behind Mathematical Induction is this: Suppose you wish to prove that some statement $P(n)$, which asserts something about each whole number n, is true for every n. For example, to prove that for all $n \geq 0$, $n < 2^n$, we would use "$n < 2^n$" as our statement $P(n)$. We wish to show that this statement holds for every n. Suppose now that we can prove two things:

(1) that $P(0)$ is true (in our example, this would mean that we can prove $0 < 2^0$);
(2) that, for any n, if $P(n)$ happens to be true, then $P(n{+}1)$ must also be true (in our example, this would mean that, if it happens to be true that $n < 2^n$, then it must be true that $n{+}1 < 2^{n+1}$).

Mathematical Induction says that, if you can prove both (1) and (2), then you have proven that, for every n, $P(n)$ is indeed true.

Below are several forms of induction. Each provides a valid approach to proving the correctness of a statement about natural numbers. Different forms are useful in different contexts. We include an example of each.

Standard Induction. Suppose $P(n)$ is a statement depending on n. If

- $P(0)$ is true, and
- under the assumption that $n \geq 0$ and $P(n)$ is true, you can prove that $P(n+1)$ is also true,

then $P(n)$ holds true for all natural numbers n.

In Standard Induction, the step in the proof where $P(0)$ is verified is called the *Basis Step*. The second step, where $P(n+1)$ is proved assuming $P(n)$, is called the *Induction Step*. As we reason during this second step, we will typically need to make use of $P(n)$ as an assumption; in this context, $P(n)$ is called the *induction hypothesis*.

Note. Standard Induction allows you to establish that a statement $P(n)$ holds for all natural numbers 0,1,2, ... However, sometimes the objective is to show that $P(n)$ holds for all numbers n that are larger than a fixed number k. Standard Induction may still be used. Here is a precise statement:

Standard Induction (General Form). Let $k \geq 0$. Suppose $P(n)$ is a statement depending on n. If

- $P(k)$ is true, and
- under the assumption that $n \geq k$ and $P(n)$ is true, you can prove that $P(n+1)$ is also true,

then $P(n)$ holds true for all natural numbers $n \geq k$.

Example. Prove that, for every natural number $n \geq 1$,

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

Solution. The statement $P(n)$ to be established for all $n \geq 1$ is:

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

For the Basis Step, notice that $P(1)$ is the statement

$$\sum_{i=1}^{1} i = \frac{1(1+1)}{2}$$

which is obviously true. For the Induction Step, we assume $P(n)$ is true, and we prove $P(n+1)$. $P(n+1)$ is the following statement:

$$\sum_{i=1}^{n+1} i = \frac{(n+1)(n+2)}{2}$$

To prove $P(n+1)$ is true, we follow these steps:

$$\begin{aligned}
\sum_{i=1}^{n+1} i &= \big(\sum_{i=1}^{n} i\big) + (n+1) \\
&= \frac{n(n+1)}{2} + (n+1) \qquad \text{(by Induction Hypothesis)} \\
&= \frac{n(n+1)}{2} + \frac{2(n+1)}{2} \\
&= \frac{(n+1)(n+2)}{2}
\end{aligned}$$

Total Induction. Suppose $P(n)$ is a statement depending on n and $k \geq 0$. If

- $P(k)$ is true, and
- under the assumption that $n > k$ and that each of $P(k), P(k+1), \ldots, P(n-1)$ are true, you can prove that $P(n)$ is also true,

then $P(n)$ holds true for all $n \geq k$.

Example. Prove that if $f(n) = 2^n$, then f is increasing.

Solution. We craft a statement $P(n)$ for this problem:

$$P(n) : \text{for all } m, \text{ if } m < n, \text{ then } f(m) < f(n)$$

Notice that $P(0)$ is vacuously true (since there are no m that are < 0) and that $P(1)$ holds since $2^0 = 1 < 2 = 2^1$. This takes care of the Basis Step.

For the Induction Step, let $n > 1$; we assume $P(k)$ holds true for all $k < n$. We prove $P(n)$ is true. Suppose $m < n$. We must show that $2^m < 2^n$. If we can show that $2^{n-1} < 2^n$, we will be done, because by the induction hypothesis, if $m < n - 1$, then $2^m < 2^{n-1}$. To verify $2^{n-1} < 2^n$, we proceed as follows:

$$2^{n-1} < 2 \cdot 2^{n-1} = 2^n.$$

We have, by Total Induction, shown that $P(n)$ holds for all n. We can use this to show that f is increasing: Suppose $m < n$. We must show that $f(m) < f(n)$. However, this is guaranteed by $P(n)$, and so we are done.

Finite Induction. Suppose $0 \leq k \leq n$, and suppose $P(i)$ is a statement depending on i, where $k \leq i \leq n$. If

- $P(k)$ is true, and
- under the assumption that $k \leq i < n$ and that $P(i)$ is true, you can prove $P(i + 1)$ is true,

then $P(i)$ holds true for all i with $k \leq i \leq n$.

Note. Another equally valid variant of Finite Induction uses an induction hypothesis that is essentially the same as the one used for Total Induction.

Example. The following is a Java method for computing $n!$ for any n.

```
int factorial(int n) {
  if(n==0 || n==1) return 1;
  int accum = 1;
  for(int i = 2; i <= n; ++i) {
    accum *= i;
  }
  return accum;
 }
```

Prove that for every n, the output of `factorial(n)` is $n!$.

Solution. Clearly `factorial(0)` and `factorial(1)` produce correct outputs. We proceed by finite induction on i, where $2 \leq i \leq n$, where $P(i)$ is the following statement:

$P(i)$: the value stored in accum after the iteration for i has finished is $i!$

For the Basis Step, we notice that before the loop begins with $i = 2$, the value stored in `accum` is 1. In the loop, this value is multiplied by i. The value in `accum` at the end is $2 = 2!$, as required.

For the Induction Step, assume that $P(i)$ holds, for $2 \leq i < n$. This means that at the end of the loop with value i, the value in `accum` is $i!$. When the iterator is incremented to $i + 1$, the current value of `accum` is multiplied by $i + 1$, which is $i + 1!$, and this value is stored in `accum`, as required.

Therefore, we have shown by Finite Induction that $P(i)$ holds for $2 \leq i \leq n$. But the value returned by `factorial(n)` is the final value stored in `accum`, which, as we have just shown, must be $n!$.

Basic Number Theory

We review some basics about number theory. Assume $a, b, c, \ldots$ are integers.

- [*divides*] $a \mid b$ means a divides b, i.e., for some c, $b = ac$
- [*floor and ceiling*] $\lfloor a \rfloor$ is the largest integer not greater than a ($\lfloor \cdot \rfloor$ is called the *floor function*) and $\lceil a \rceil$ is the smallest integer not less than a ($\lceil \cdot \rceil$ is called the *ceiling function*).

  Note. The floor function applied to rational numbers a/b yields the same results as Java's integer division when both a and b are positive. However, when one is negative and the other positive, the results differ:

$$\begin{aligned} -5/4 = -(5/4) = -1 &\quad \text{(Java integer division)} \\ \lfloor -5/4 \rfloor = -2 &\quad \text{(mathematics)} \end{aligned}$$

- [*greatest common divisor*] $c = \gcd(a, b)$ means c is the largest integer that divides both a and b
- [*least common multiple*] $c = \mathrm{lcm}(a, b)$ means c is the smallest integer for which $a \mid c$ and $b \mid c$
- [*modulus*] If $a > 0$, then $b \bmod a$ equals the (nonnegative) remainder on dividing b by a. ($b \bmod a$ is a nonnegative number less than a.)

  Note. Java's mod function `%` is the same as mod for positive inputs, but if $a, b > 0$, then $-a \,\%\, b = -(a \bmod b)$.

  Example: $8 \,\%\, 3 = 8 \bmod 3 = 2$

  Example: $-8 \bmod 3 = 1$ but $-8 \,\%\, 3 = -(8 \bmod 3) = -2$

- [*congruence*] $b \equiv a (\bmod\, n)$ means $b \bmod n = a \bmod n$. Equivalently $n \mid (b - a)$ (see one of the examples below for a proof of this equivalence).
- **The Division Algorithm.** For each pair of integers a, b with $a > 0$, there is a unique pair q, r such that
  - ○ $b = aq + r$ (q is the *quotient*, r is the *remainder*), and
  - ○ $0 \leq r < a$.

  Moreover, $q = \lfloor \frac{b}{a} \rfloor$ and $r = b \bmod a$.

  Note. The equation $b = aq + r$ also holds with $q = \frac{b}{a}$ (integer division) and $r = b \,\%\, a$, but in the case where $a > 0$ and $b < 0$, it turns out that $r < 0$ (so the inequality $0 \leq r < a$ given above fails if these computations are used).

- [primes] A positive integer p is *prime* if its only positive divisors are 1 and p. A positive integer c is *composite* if there are positive integers m, n, both greater than 1, such that $c = m \cdot n$.

  Example. Show that every integer > 1 is a product of primes. (A prime itself is considered a product of primes.)

  Solution. Proceed by induction on natural numbers $n \geq 2$. Since 2 is prime, 2 is a product of primes. This takes care of the base case. Proceeding with Total Induction, assume $n > 2$ and every number $< n$ is a product of primes. Consider n. If n is already prime, we are done.

If n is composite, $n = m \cdot k$, then since both m, k are $< n$, by the induction hypothesis, each of m, k is a product of primes. It follows that n is a product of primes. This completes the induction and the proof.

Example. Prove that there are infinitely many primes.

Solution. Suppose there were only finitely many primes. Let $p_0, p_1, p_2, \ldots, p_m$ be a list of all primes in increasing order. Let P be the product of these primes; that is, let $P = p_0 \cdot p_1 \cdot p_2 \cdot \ldots \cdot p_m$. Since $P+1$ is larger than all the primes in the list, $P+1$ must be composite and we can write $P+1 = k \cdot n$ for some k, n. By the previous example, k must be a product of primes; in particular, some p_i divides k; it follows that p_i divides $P+1$. But p_i also divides P (recall the definition of P). Hence, p_i divides the difference $(P+1) - P$, which is impossible. Therefore, there cannot be only finitely many primes.

- *Fibonacci Numbers.* The sequence $F_0, F_1, F_2, \ldots, F_n, \ldots$ of Fibonacci numbers is defined by

$$\begin{aligned} F_0 &= 0; \\ F_1 &= 1; \\ F_n &= F_{n-1} + F_{n-2}. \end{aligned}$$

Example. Suppose $g = \gcd(a, b)$. Show that there are integers x, y so that $g = ax + by$.

Solution. Let $S = \{z \mid z \text{ is positive, having the form } ax + by \text{ for some } x, y\}$, and let $d = \min S$. Then for some x, y we have $d = ax + by$. We first show that $d|a$ and $d|b$. Suppose $d \not| a$. Then there are q, r with $a = dq + r$ and $0 < r < d$. But now we have

$$r = a - dq = a - (ax + by)q = a(1 - xq) + b(-q).$$

Writing r in this way shows that $r \in S$; but this is impossible because $0 < r < d$ and $d = \min S$. Therefore, in fact, $d|a$. A similar argument shows that $d|b$.

Next we show that $d = \gcd(a, b)$ by showing that $d \geq c$ for any common divisor c of a, b. So, let c be a common divisor of a, b. As above, we write $d = ax + by$. Since $c|a$ and $c|b$, we also have $c|(ax + by)$, and so $c|d$. Therefore $d = \gcd(a, b)$.

Example. Suppose $g = \gcd(a, b)$ and suppose d is some other common divisor of a, b; that is, suppose $d|a$ and $d|b$. Show that $d|g$.

Solution. By the previous Example, we may write $g = ax + by$ for some integers x, y. Since $d|a$ and $d|b$, there are integers s, t such that $a = ds$ and $b = dt$. Therefore

$$g = ax + by = dsx + dty = d(sx + ty).$$

It follows that $d|g$.

Example. Show that $a \equiv b \bmod n$ if and only if $n|(a - b)$.

Solution. We may write

$$a = \lfloor \frac{a}{n} \rfloor \cdot n + a \bmod n$$
$$b = \lfloor \frac{b}{n} \rfloor \cdot n + b \bmod n$$

Subtracting,

$$(*) \qquad a - b = \big(\lfloor \frac{a}{n} \rfloor \cdot n - \lfloor \frac{b}{n} \rfloor \cdot n\big) + \big(a \bmod n - b \bmod n\big).$$

If $n|(a-b)$, then since the expression $\big(\lfloor \frac{a}{n} \rfloor \cdot n - \lfloor \frac{b}{n} \rfloor \cdot n\big)$ in $(*)$ is also divisible by n, it follows that $\big(a \bmod n - b \bmod n\big)$ is divisible by n too. But since both $a \bmod n$ and $b \bmod n$ lies in the range $[0, n-1]$, their absolute difference must also lie in the range $[0, n-1]$; hence, the only way this difference could be divisible by n is if it equals 0. In other words, we must have that $a \bmod n = b \bmod n$.

Conversely, if $a \bmod n = b \bmod n$, then their difference is 0 and so divisible by n. It follows that the entire right-hand side of the expression in $(*)$ is divisible by n. Therefore, $n|(a-b)$.

Example. Find the unique q and r guaranteed by the Division Algorithm, where $a = 7$ and $b = -20$; that is, find q, r with $b = aq + r$ and $0 \leq r < a$. Then obtain values q' and r' such that $b = aq' + r'$ that makes use of Java's mod function.

Solution. Recall that $q = \lfloor b/a \rfloor = \lfloor -20/7 \rfloor = -3$ and $r = b \bmod a = 1$. Therefore $-20 = 7q + r = 7(-3) + 1$.

Computing values q' and r' using Java's approach yields:

$$q' = b/a \text{ (integer division)} = -20/7 = -(20/7) = -2$$
$$r' = b\% a = -20\% 7 = -(20\% 7) = -6$$

Therefore, we can write $-20 = 7q' + r' = 7(-2) - 6$.

Example. Show that if a, b are nonzero integers and $\ell = \mathrm{lcm}(a, b)$, then the rational number $\frac{ab}{\ell}$ is an integer.

Solution. Suppose $\frac{ab}{\ell}$ is not an integer; this means that $\ell \not| ab$. Use the Division Algorithm to find q, r with

$$ab = \ell q + r \quad \text{and} \quad 0 < r < \ell.$$

Subtracting ℓq from both sides yields

$$r = ab - \ell q.$$

Since ℓ and ab are both multiples of a, r must also be a multiple of a. Since ℓ and ab are both multiples of b, r must also be a multiple of b. Therefore, r is a common multiple of a, b, and yet $0 < r < \ell$ — but this is impossible since $\ell = \mathrm{lcm}(a, b)$. Therefore, $\frac{ab}{\ell}$ is in fact an integer.

Example. Show that if a, b are nonzero integers and $\ell = \mathrm{lcm}(a, b)$, then the integer $\frac{ab}{\ell}$ divides both a and b.

Solution. By the previous example, $k = \frac{ab}{\ell}$ is an integer. Let s, t be integers such that $\ell = as = bt$ (using the definition of ℓ). Then we have

$$\begin{aligned} ks &= s \cdot \frac{ab}{\ell} = s \cdot \frac{ab}{as} = s \cdot \frac{b}{s} = b, \text{ and} \\ kt &= t \cdot \frac{ab}{\ell} = t \cdot \frac{ab}{bt} = t \cdot \frac{a}{t} = a \end{aligned}$$

This shows that $k|a$ and $k|b$, as required.

Example. Show that for all nonzero a, b, $\gcd(a, b) \cdot \operatorname{lcm}(a, b) = |ab|$.

Solution. Let $g = \gcd(a, b)$ and $\ell = \operatorname{lcm}(a, b)$. Let $\ell' = \frac{|ab|}{g}$ and let $g' = \frac{|ab|}{\ell}$. To complete the proof, it suffices to show that $\ell = \ell'$. Our plan is to show that $\ell \leq \ell'$ and that $g' \leq g$. This is enough because, assuming we have established these two inequalities, we will have:

$$\begin{aligned} g' \leq g \ &\Rightarrow \ \frac{|ab|}{\ell} \leq \frac{|ab|}{\ell'} \\ &\Rightarrow \ \frac{1}{\ell} \leq \frac{1}{\ell'} \\ &\Rightarrow \ \ell' \leq \ell \end{aligned}$$

In other words, we will have both $\ell \leq \ell'$ and $\ell' \leq \ell$.

Claim 1. $\ell \leq \ell'$

Proof of Claim 1. We will show that ℓ' is a common multiple of a, b; it will then follow that $\ell \leq \ell'$. Notice that

$$\ell' = \frac{mn}{g} = m \cdot \frac{n}{g} = n \cdot \frac{m}{g}.$$

Since $\frac{m}{g}$ and $\frac{n}{g}$ are integers (by definition of g), the displayed equations show that ℓ' is a multiple of both m and n, as required.

Claim 2. $g' \leq g$.

Proof of Claim 2. By the previous Example, g' is a common divisor of a, b. Since g is the *greatest* common divisor of a, b, it follows that $g' \leq g$.

Review of Recursion. (See the file Recursion.pdf)

Appendix: More On Mathematical Induction

In this Appendix, we'll perform an induction argument in detail, and address some of the issues that sometimes make this mathematical technique difficult to understand at first. We'll use Mathematical Induction to prove that for all n, $n < 2^n$. We are required to demonstrate that $P(n)$ holds for all n, where $P(n)$ is simply the statement $n < 2^n$. We need to establish

(1) the Basis Step, that $P(0)$ holds, and
(2) the Induction Step, that if $P(n)$ holds, then $P(n+1)$ also holds.

Part (1) asserts that $0 < 2^0$ — and certainly this is true, since $2^0 = 1$. To prove (2), the Induction Step, we let n denote any whole number whatsoever, and assume for the moment that $n < 2^n$ is true; we are not claiming yet that we know for sure that $n < 2^n$—we are just *assuming* for the moment that it is true. The reason for making this assumption is that we wish to show that, whenever this assumption is made, it necessarily implies that $n + 1 < 2^{n+1}$. Now, assuming $n < 2^n$, here's how we can reason: Certainly if we add 1 to each side of this inequality, it remains true:

$$n + 1 < 2^n + 1.$$

To proceed, we make one important observation: Whenever you double a natural number m (that is, a *positive* whole number m), the number that you get is at least as big as $m + 1$. Certainly $2 \cdot 1 \geq 1 + 1; 2 \cdot 2 \geq 2 + 1; 2 \cdot 3 \geq 3 + 1$, and so forth. (To prove this properly, we actually need to use Mathematical Induction again — I will return to this point below.) Using our observation, notice that, no matter what n is, $2^n \geq 1$. Therefore, doubling 2^n produces a number at least as big as $2^n + 1$. This lets us extend our inequality:

$$n + 1 < 2^n + 1 \leq 2 \cdot 2^n.$$

Finally, using properties of exponents, you can perform the computation $2 \cdot 2^n = 2^{n+1}$, and we therefore have

$$n + 1 < 2^n + 1 \leq 2 \cdot 2^n = 2^{n+1}.$$

This chain of inequalities and equalities demonstrates that $n + 1 < 2^{n+1}$. This is exactly what we needed to prove to handle part (2). By Mathematical Induction, we have therefore demonstrated that for all whole numbers n, $n < 2^n$.

On first encountering Mathematical Induction, students often find the Induction Step puzzling, particularly in the way that it is applied, as in the proof we gave in the last paragraph. Here is the question:

> How can we *assume* that $n < 2^n$ when this is precisely what we are trying to *prove*?

Before answering the question, let me describe an analogous situation in real life, in the form of a simple game, that will make the technical point more clear. Suppose four people are sitting side-by-side in a room:

Jim, Lisa, Carey, Anne.

They are going to play a simple game. Jim puts on a pair of headphones, attached to a tape recorder. What he hears on the recording is just the single word "hello". Jim thereupon whispers this word to Lisa, who whispers it to Carey, who in turn whispers it to Anne. Assuming that everyone says the word, and hears the word, correctly, it is certain that everyone in the group hears the word "hello".

Now let's include more people in the game. Instead of having four people sitting side-by-side, let's assume there are one thousand people sitting side-by-side along the coast of Florida. The first person, whose name is Jim again, listens to a tape recording in which the single word "hello" is (again) uttered. Consider the following question: What has to happen to guarantee that every one of the one thousand people will hear this word by the time the game has ended? I can give you a long list of conditions that will give such a guarantee in the following way: Rather than inventing names for all these people, let's say that Jim is Person #0; the person seated next to him is Person #1; the next person is Person #2; and so forth, all the way to Person #999 (there are 1000 numbers from 0 through 999). Here is the list of conditions:

(0) Person #0 accurately hears the word "hello"

(1) If Person #0 accurately hears the word "hello", then Person #1 also accurately hears the word "hello"

(2) If Person #1 accurately hears the word "hello", then Person #2 also accurately hears the word "hello"

(3) If Person #2 accurately hears the word "hello", then Person #3 also accurately hears the word "hello"

.

.

.

(999) If Person #998 accurately hears the word "hello", then Person #999 also accurately hears the word "hello"

Let's see why these 1000 conditions provide us with the necessary guarantee. Condition (0) tells us that Person #0 — Jim — hears the word accurately. Condition (1) tells us that, *if* Jim hears the word accurately, Person #1 also hears it accurately. Since Condition (0) tells us that Jim *does* hear the word accurately, we can conclude that Person #1 also hears the word accurately. Here's the logic:

> Jim hears the word accurately
> If Jim hears the word accurately, Person #1 hears the word accurately.
>
> ---
>
> Person #1 hears the word accurately.

(The horizontal line is read "therefore".) Now that we know that Person #1 hears the word accurately, we can combine this fact with Conditon (2) (which says that if Person #1 hears the word accurately, then so does Person #2) to conclude that Person #2 hears the word accurately. Combining conditions in this way, we successively can verify that Person #3, Person #4, . . ., Person #999 each hears the word accurately.

Our long list of conditions works, but we have written them down in a long and laborious way. A more compact way of describing these conditions is the following:

(1′) Jim accurately hears the word "hello"

(2′) For any n from 0 through 998, if Person $\#n$ hears the word "hello" correctly, then the next person—Person $\#n+1$—also hears the word "hello" correctly.

Notice that condition (2′) is a compact way of stating conditions (1)–(999) listed above: When $n = 0$, (2′) is the same as (1); when n=1, (2′) is the same as (2); when $n = 2$, (2′) is the same as (3); and so forth, all the way to $n = 998$, in which case (2′) is the same as (999). (2′) simply says that *assuming* the nth person hears the word correctly, the $n + 1$st person must hear it accurately too. Certainly, if we are somehow guaranteed that (1′) and (2′) are both true, then we are absolutely guaranteed that every one of the thousand people in the game has heard the word "hello" accurately.

We can use this game example to better understand Mathematical Induction. Mathematical Induction—when applied to the problem of showing that for all n, $n < 2^n$—can be viewed as saying the following: If you can prove all of the following conditions, then it will follow that for all n, $n < 2^n$:

- $0 < 2^0$
- If $0 < 2^0$, then $1 < 2^1$
- If $1 < 2^1$, then $2 < 2^2$
- If $2 < 2^2$, then $3 < 2^3$

.

.

.

- If $n < 2^n$, then $n + 1 < 2^{n+1}$

.

.

.

The first of these conditions says $0 < 2^0$; assume we can prove this. Assume we can also prove the next condition, that if $0 < 2^0$, then $1 < 2^1$. It would then follow that in fact $1 < 2^1$. Again, assume we can prove the next one, that if $1 < 2^1$ then $2 < 2^2$. Then since $1 < 2^1$, we may conclude that $2 < 2^2$. Continuing in this way, we can see that for each n, $n < 2^n$.

Of course, listing infinitely many conditions in this way is laborious. As in the Florida coast game, we can reduce this list to two, as follows:

(1) $0 < 2^0$

(2) For any whole number n, if $n < 2^n$, then $n + 1 < 2^{n+1}$.

These are the two conditions we originally stated in describing Mathematical Induction; this principle asserts that, if you can prove (1) and (2), then you may conclude that, for all n, $n < 2^n$. Therefore, again, a proof by Mathematical Induction that $n < 2^n$ for all n involves proving (1) and (2).

We can now address the question that was raised earlier about our proof using induction. The proof of (1) was easy: certainly $0 < 2^0$. What must we do to prove (2)? Notice that, again, (2) is a compact way of writing down infinitely many conditional statements of the form "if . . ., then . . .". How can we prove a conditional statement like this?

Consider a simpler example. Suppose you had to prove the following statement: if n is even, then $n + 1$ is odd. The way this is done is to *assume* n is even, and then prove $n + 1$ must be odd (which is true since the next even number after n must be $n + 2$). This example shows how, in order to prove conditional statements, it is necessary to *assume* the hypothesis (the "if" part), and then prove the consequent (the "then" part).

Therefore, to prove (2) in our original example, we must *assume* that $n < 2^n$ and then *prove* $n + 1 < 2^{n+1}$. This is precisely what was done.

As another example of Mathematical Induction, let's prove, as promised earlier, that for all positive whole numbers n, $2 \cdot n \geq n + 1$. Here, the property $P(n)$ is simply $2 \cdot n \geq n + 1$. As usual, there are two things to prove:

(1) $P(1)$

(2) For all $n \geq 1$, if $P(n)$ is true, then $P(n + 1)$ is true.

For the step (1), the Basis Step, we must prove $P(1)$ rather than $P(0)$ in this case because we are trying to establish the result for all *positive* whole numbers.

Proving (1) is easy: $P(1)$ asserts that $2 \cdot 1 \geq 1 + 1$, and this is certainly true. For (2), the Induction Step, we assume that $P(n)$ is true and prove $P(n+1)$ is true. $P(n)$ says that $2n \geq n+1$; let us assume this for the moment. We must prove $P(n+1)$, which says that $2(n+1) \geq (n+1)+1$. We can demonstrate $P(n + 1)$, using $P(n)$, with a chain of equalities and inequalities; first notice that since $2n \geq n + 1$, it follows that $2n + 1 \geq (n + 1) + 1$. Therefore,

$$2(n + 1) = 2n + 2 > 2n + 1 \geq (n + 1) + 1.$$

This shows $2(n + 1) \geq (n + 1)$, and we have proven (2). By Mathematical Induction, we may conclude that for all positive whole numbers n, $2n \geq n + 1$.