3

Some Integer Functions

A Pair of Fundamental Integer Functions

The integer function that is the heart of this section is the *modulo* function. However, before getting to it, let us look at some very simple functions. The first (and most important) of these is the *floor* function. We denote this function by $\lfloor x \rfloor$ although it can be denoted by floor(x) It is also widely known as the *greatest integer* function and is found in some computer languages as the *integer* function (sometimes denoted INT(x)). The floor of x is the greatest integer less-than-or-equal-to x. For example, $\lfloor 4 \rfloor = 4$, $\lfloor 2.5 \rfloor = 2$, $\lfloor -2 \rfloor = -2$, $\lfloor -2.5 \rfloor = -3$, $\lfloor \pi \rfloor = 3$, $\lfloor -\pi \rfloor = -4$. (Beware of using the *truncate* function found in computer languages as it will only be correct for positive numbers). Note that the floor function is an integer function in that it always returns an integer, although it is applied to all real numbers.

The *ceiling* function complements the floor function. It is denoted by $\lceil x \rceil$ or ceiling(x). The ceiling of x is the least integer greater-than-or-equal-to x. For example, $\lceil 4 \rceil = 4$, $\lceil 2.5 \rceil = 3$, $\lceil -2 \rceil = -2$, $\lceil -2.5 \rceil = -2$, $\lceil \pi \rceil = 4$, $\lceil -\pi \rceil = -3$.[1]

> Informally, the floor function $\lfloor x \rfloor$ rounds x down. The ceiling function $\lceil x \rceil$ rounds x up. If x is an integer, $\lfloor x \rfloor = \lceil x \rceil = x$. $\lfloor x \rfloor = \lceil x \rceil$ if and only if x is an integer. $\lceil x \rceil - \lfloor x \rfloor = 0$ if x is an integer; otherwise it is 1.

---

[1]A rather exotic example of the ceiling function is as follows: the following function for integers greater than 4 evaluates to 1 if n is a prime otherwise it is 0.

$f(n) = \left\lceil \left\lceil \dfrac{(n-1)!}{n} \right\rceil - \dfrac{(n-1)!}{n} \right\rceil$. This is based upon Wilson's Theorem which is given in a later section.

Basic Applications of the Floor Function

If we want to round the number x to the nearest integer, rounding up (as usual) when the number is exactly an integer plus a half then $\lfloor x + .5 \rfloor$ does the trick (proving this is an exercise). *Integer division*, which is sometime denoted by \ means division without remainder. For example, $8 \setminus 3 = 2$, $6 \setminus 2 = 3$, $7 \setminus 2 = 3$. $x \setminus y$ is equivalent to $\left\lfloor \dfrac{x}{y} \right\rfloor$. To round a number to exactly two digits, for example to round 3.14159 to 3.14, we can use the formula $\dfrac{\lfloor 100x \rfloor}{100}$.

Suppose we want to divide the interval of real numbers $0 \leq x \leq 1$ into twenty subinterval of length .05. Given a number, y, such that $0 \leq y \leq 1$, we might ask which sub-interval does y fall into, and the answer is $\left\lfloor \dfrac{y}{.05} \right\rfloor + 1$. For example the number .07 falls in the second sub-interval because $\left\lfloor \dfrac{.07}{.05} \right\rfloor + 1 = 2$.

> The next two exercises more or less call for proofs. However, this book is not just for math, science, or engineering majors. If you are uncomfortable with proofs, or if you are merely reading this book, try to use examples to convince yourself that the statements of the exercises are true. For these particular exercises, examples will not prove the statements, but they can convince you that the statements are true and that is the most important thing.

☐ **Exercise 1**   Almost every computer language has the floor function or some equivalent. Show that you can define the ceiling function in terms of the floor function by $\lceil x \rceil = -\lfloor -x \rfloor$ (in other words, show that this identity is true).

☐ **Exercise 2**   Show that the function round(x), which rounds x to the nearest integer, can be defined by round(x) = $\lfloor x+.5 \rfloor$.

An Application of the Floor Function to Random Numbers

Section 11 covers computer generation of random numbers. Typically random number generators return uniform variates. These are numbers that are intended to be between 0 and 1 and such that all numbers are equally likely to be chosen.[1] Suppose, for some reason we want to take uniform variates from two generators and combine them. For instance, the first generator gives us u and the second generator gives us v. We add them together to get a new variate $w = u + v$. The range of w is from 0 to 2, so if we divide w by 2, our new variate $w/2$ lies between 0 and 1. However, $w/2$ is not uniformly distributed. It is triangular; it is much more likely to be close to .5 than to either .0 or .1. However, if we define w by $w = u + v - \lfloor u + v \rfloor$
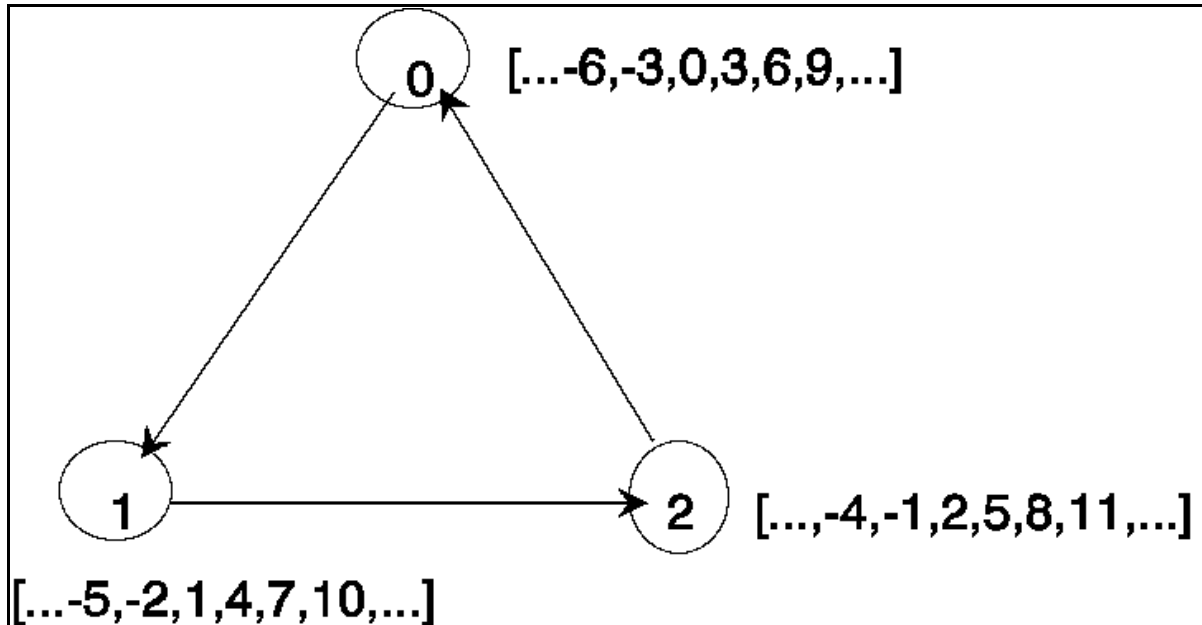
---

[1]Many computer programs return random numbers in a variety of schemes. However, numbers randomly distributed from 0 to 1 are to be preferred. It is almost always possible in a relatively simple way to transform these numbers into the form you want, whether integers uniformly distributed from 1 to 100 or Normal variates with a given mean and standard deviation.

it is uniformly distributed between 0 and 1. To understand why it is only necessary to look at the graph of the distribution of u + v.

> When we take a uniform variate u distributed from 0 to 1, if we create the new variable v = au + b, v is uniformly distributed from b to b + a. In particular v = au is uniformly distributed from 0 to a.

> Let u be a variate uniformly distributed from 0 to 1. Let n be a positive integer. $\lfloor n{\cdot}u \rfloor +$ 1 gives the integers 1, 2, 3 through n (uniformly distributed). If a is any integer $\lfloor n{\cdot}u \rfloor + a$ gives the integers a, a+1, a+2 through a+n$-$1.

The Modulo Function

Figure 1 The Integers Modulo 3.

There are two incarnations of *modulo*: one is the modulo *function* used by computer scientists and the other is the modulo *relation* used by mathematicians. We will explore both versions starting with the modulo relation.[1] Consider the graph in **Figure 1**. The vertices are numbered 0, 1, and 2. The brackets next to each vertex count the number of arcs to travel from Vertex 0 to the vertex in question. The brackets next to Vertex 0 contains the integer 0 because we choose to agree that one can go from Vertex 0 to Vertex 0 in zero steps. By following the arcs from Vertex 0 to Vertex 1 to Vertex 2 and back to Vertex 0, we have transversed 3 arcs. If we travel the same path in the opposite order, that is against the directions of the arcs, we say we have transversed -3 arcs. Similarly we can travel from Vertex 0 to Vertex 0 in -6 arcs, or 9 arcs, or 102 arcs. Clearly there are an infinite number of possibilities as indicated by the ellipses (...). The brackets next to Vertex 1 indicate that it is possible to go from Vertex 0 to Vertex 1 in 1 step or 4 steps or -2 steps, and so on. Notice that the numbers in the three brackets partition

---

[1]This strategy enables me to maximize the all-important confusion function that is so important to text writers everywhere.

all of the integers into three classes. Let us denote these classes next to the vertices as follows: *[0]* will denote the class [..., -6, -3, 0, 3, 6, 9, ...] and the other classes are denoted *[1]* and *[2]* respectively.

There are several observations that can be made about these classes of numbers. Class [0] has the property that it contains the differences of every pair of numbers in the class. For example, since 18 and 54 belong to the class so do 18 - 54 = -36 and 54 - 18 = 36. We define a *module* as any class of numbers containing at least two numbers and containing the differences of every pair of numbers in the class. Hence, we say that [0] is a module but [1] and [2] are not. If we take any of the three classes, the difference in any two numbers belongs to [0]. More concisely, the difference between any two numbers in any class is a multiple of three—the number of nodes in the graph.

Let us pick any two of the three classes or better yet, you the reader, pick any two of the three classes (you can choose the same class twice). Now choose two numbers—one from each class. First add the numbers and find out what class the sum is in. Now do the same experiment again and again but with the same two classes. You will find that the sum always falls in the same class it did the first time. Try this for multiplication and you will get the same phenomenon. For example, every time we add a number from [1] to a number from [1] we get a number in [2] regardless of which numbers we choose. Similarly, every time we multiply a number in [2] times a number in [2], we get a number in [1]. Hence, we can save time by operating on just the classes [0], [1], [2].

☐ **Exercise 3**   Write addition and multiplication tables for the classes [0], [1], [2]. As per the preceding discussion, to build the table you simply pick a sample number from each class. In fact the simplest choices are the numbers 0, 1, 2.   Notice that we have in effect discovered an algebra consisting of three numbers,[1] since we have partitioned the integers into three classes. Belonging to the same class is an equivalence relation. In the parlance of mathematics, we indicate that integers A and B belong to the same class by saying *A ≡ B (mod 3)* or in words: *A is congruent to B mod 3*. If you like extra syllables then say A is congruent to B *modulo* 3. In general, we can use any integer greater than one as a modulus. In any case, a graphical interpretation such as in **Figure 1** applies.

▸   Integer division will be important for a while, so we will use special notation for it. We say that **the integer, x, is divided by the integer y, if there is another integer z, such that x = y·z.** We write this as y|x. If the integer x divides both y and z, we write x|y,z. In other words, **A ≡ B (mod n) is equivalent to n|A - B**.

> We can also define modulo arithmetic by saying A ≡ B (mod n) whenever, n divides A – B. This is both quick and practical. This relation partitions the integers into n classes.

It is customary to write parentheses around the words *mod n*. However, I often leave off the parentheses on the grounds that they are redundant. There is no significance to this other than I am inconsistent.

---

[1]This was discovered for the first time by C. F. Gauss while he was still a boy in the 1790's. Those of you who are math, science, or engineering majors may want to figure out how many of the properties this system has as compared to the properties of the real numbers you were given in basic algebra. A hint: you should find virtually all of the familiar properties of addition and multiplication. Division is trickier. You have division in this case, modulo 3 arithmetic, but you do not have it in modulo 4 arithmetic as will be explained later.

Again, when we look at the classes of integers modulo 3, which are the three classes given in **Figure 1**, we can choose any number from a class to represent the class. The *standard* representative of a class is the first non-negative integer in that class.

Example   Consider the integers mod 5. This gives us five classes of integers whose standard representatives are 0, 1, 2, 3, 4. For example, 0 is the standard representative of the class $\{...-10,-5,0,5,10,15...\}$. We write:

$[0] = \{...-10,-5,0,5,10,15...\}$

$[1] = \{...-9,-4,1,6,11,16...\}$

$[2] = \{...-8,-3,2,7,12,17...\}$

$[3] = \{...-7,-2,3,8,13,18...\}$

$[4] = \{...-6,-1,4,9,14,19...\}$.

Given an integer, say 10, what is the standard representative of its class mod 3? Since 10 is in the class [1], the answer is 1, and we write $1 \equiv 10 \bmod 3$. Clearly this definition works for any modulus (by which we mean a positive integer greater than one). From **Figure 1** we can see that -5 mod 3 is 1, that is, $1 = -5 \bmod 3$. However, a word of warning! Do not trust a compiler to give you the correct answer when taking the modulus of a negative number. The general implementation of the mod function, which is basically as we will use it, is that mod is a function of two numbers sometimes written *mod(x, y)* and in most implementations written *x mod y*, and y is a positive integer and x is a non-negative integer with the function x mod y defined to be the remainder of x divided by y. For example: 7 mod 2 = 1, 25 mod 7 = 4, 5 mod 6 = 5, 0 mod 8 = 0.

Ordinary twelve-hour clock time is done in modulo arithmetic. If a job starts at 10 o'clock and lasts 4 hours then it is finished at 2 o'clock. The only difference between clock time and arithmetic mod 12 is that in mod 12 we say usually 0 instead of 12.

8

☐ **Exercise 4**   A very nice application of the mod function is to formatting computer output. Try writing a program that will print the first n positive integers in m columns where you give n and m as inputs.

Another application of the mod function is to monitoring a program. If you have a large number of iterations of i and you print each i, you lose an enormous amount of time writing to the screen and the i's change too fast to read. Instead use a statement like: If i MOD 100 = 0 Then Print i. In that case, only multiples of 100 are printed.

k mod n = h means that h is the remainder of k divided by n.

k mod n ≡ h means that n|h – k (that is: n divides the difference of h and k).

☐ **Exercise 5**   The difference between the mod function and the mod relation is subtle. Show that if A mod n = B that A ≡ B (mod n), but that the implication does not work in the other direction.

☐ **Exercise 6**   Suppose that you have a compiler which does the mod function for positive integers. Write an extension of that function so that it works for negative integers. In other words, suppose that your computer will return x mod y when x is non-negative and y is positive. Write a function definition, say FNMOD(x, y) so that FNMOD works whether x is negative as well as non-negative (y is always positive).

Example   We are writing a program to simulate dealing from a deck of cards. We want to deal one card from the deck with each card having equal probability of being chosen. First, we use the built-in random number generator to give us a real

9

number, rand, with $0 <$ rand < 1. We then compute the floor of rand·52: $\lfloor$rand·52$\rfloor$. This gives the integers 0, 1, 2, through 51, each with equal probability of occurrence. Let n = $\lfloor$rand·52$\rfloor$. To find the value of the card we use $\lfloor$rand·52$\rfloor$ mod 13 + 1. For example, 9 mod 13 + 1 = 10; 51 mod 13 + 1 = 13 is a King; 26 mod 13 + 1 = 1 is an Ace. If we take $\lfloor$rand·52$\rfloor$ mod 4, that tells us the suit. 9 mod 4 = 1 is a diamond; 51 mod 4 = 3 is a spade; 2 mod 4 = 2 is a heart; 8 mod 4 = 0 is a club.

The following definition is always correct. It works even for negative u (v must be positive) and it is frequently useful.

$$u \bmod v \;=\; u - \left\lfloor \frac{u}{v} \right\rfloor v$$

Another Definition for the Mod Function

Modulo Arithmetic

One of the cute aspects of modulo numbers is that they almost obey the usual rules we know for algebra and arithmetic.

▸ Let us review for a moment: Going back to our first example, we built the integers mod 3 as three classes of numbers [0], [1], [2]. We realized the classes gave us an arithmetic because we can use any representative of a class to represent the class in arithmetic and we always get the same result. In general, we choose the standard representatives to be the least non-negative element of its class. In the case of arithmetic mod n, these numbers are 0, 1, 2, 3, ... , n − 1. They give us an arithmetical system and mathematicians often denote this system of integers mod n by Z_n. Within Z_n we no longer write $A \equiv B \pmod{n}$, but we write $A = B$. If, for example, in Z_8 you have a number, X, outside the domain 0,1,...,7 you should view it as equivalent to X mod 8. For example, in mod 8 we write 23 = 7, 16 = 0, -3 = 5.

Within the integers mod n or within Z_n (we can use these interchangeably) addition subtraction and multiplication work like usual. In particular:

> # If X ≡ Y (mod n) then for any integer Z
>
> ▷   **X + Z ≡ Y + Z (mod n)**
>
> ▷   **X - Z ≡ Y − Z (mod n)**
>
> ▷   **X · Z ≡ Y · Z (mod n)**

All the above rules are as in basic algebra. However, it is not an accident that nothing is said about division. Division is covered in Chapter 11.

Example   We want to solve x + 8 = 5 (mod 10). By adding 2 to both sides we get
x + 10 = x = 7 (mod 10). Remember, in (Mod 10) or Z_{10}, 10 = 0.

☐ **Exercise 7**   Solve the following problems in Z_7 for X:

X + 3 = 6

X + 5 = 3

X − 4 = 5

X + 1 = 6.

The Two Queens Problem (Which Uses both Floor and Modulo Functions)

The queen in chess moves (and captures) along rows, columns and diagonals. Suppose we number the squares of the board, starting from the left of the top row and moving from left to right until ending at the lower right square, and we number the squares from 0 to 63. Then we pick two different numbers in the range from 0 to 63 (inclusive). Let the positions of the queens be given by the two different integers. x and y. How, numerically can we decide whether either, moving as a queen can take the other? It is not hard to show that t he following rules will

work: The row numbers of the two positions are respectively $\left\lfloor \dfrac{x}{8} \right\rfloor$ and $\left\lfloor \dfrac{y}{8} \right\rfloor$ (giving numbers

0 through 7). The columns are given by x mod 8 and y mod 8, which numbers the columns from 0 to 7. The queens are on the same row or same column if they have the same row or column numbers respectively. They are on the same diagonal if the absolute difference of the row numbers equals the absolute difference of the column numbers. For example, suppose our position numbers are x = 20 and y = 41. Then the row numbers are respectively 2 (the third row) and 5 (the sixth row). The column numbers are 4 and 9. The queens are on the same diagonal if $|2-5| = |4-9|$, that is if $3 = 5$. Since this is false, they are on different diagonals.

1.   First, if you haven't done so already, read the box immediately preceding this problem. Secondly, the statement is clearly true if x is an integer. If x is an integer then $\lceil x \rceil = \lfloor x \rfloor$. Hence, if x is an integer, $-\lfloor -x \rfloor = -(-x) = x = \lceil x \rceil$. (I did the case where x is an integer first because the following proof only works when x is not an integer; note that this identity is only interesting if x is not an integer.) Assume now that x is not an integer. We can define the fractional part of x, by $frac(x) = x - \lfloor x \rfloor$. The key observation to this proof is that $frac(-x) = 1 - frac(x)$. Note also that $\lfloor x \rfloor = x - frac(x)$. We have that $-\lfloor -x \rfloor$ $= -(-(x) - frac(-x)) = (x + frac(-x)) = x + 1 - frac(x) = \lfloor x \rfloor + 1 = \lceil x \rceil$. It is this last equality that is true if and only if x is not an integer.

2.   Notice that if x is closer to $\lfloor x \rfloor$ than $\lceil x \rceil$, then $round(x) = \lfloor x \rfloor$ otherwise $round(x) = \lceil x \rceil$. If it is equally close to both of them the usual convention is that $round(x) = \lceil x \rceil$. It can easily be seen that the statement is true if x is an integer. Assume x is not an integer. Then $round(x) = \lfloor x \rfloor$ if and only if $frac(x) < .5$ ($frac(x)$ is defined in the solution to the previous problem). However, $frac(x) < .5$ if and only if $frac(x + .5) < 1$ in which case $\lfloor x+.5 \rfloor = \lfloor x \rfloor$. Otherwise, $frac(x + .5) \geq 1$ and $\lfloor x+.5 \rfloor = \lceil x \rceil$.

3.

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

×	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

4.   Input n, m
       $j \leftarrow 0$ {$\leftarrow$ is the *assignment* operator: See box in 3 section 4.}
       For $i \leftarrow 1$ to n
       print i
       move to the next column to the right
       $j \leftarrow j + 1$ (mod m)
       if j = 0 go to first column of next line

13

5.   If A mod n = B then B is the remainder from division of A by n. That is there must exist some integer k such that n·k + B = A (with $0 \leq B < n$). Hence, A − B = n·k or equivalently $A \equiv B$ (mod n). Going the other way however, we have $20 \equiv 30$ (mod 5) but it is **not** true that 20 mod 5 = 30 (since 20 mod 5 = 0, and 0 is the unique answer).

6.   Define FNMOD(x, y)
       If $y \leq 0$ then return "error"
       If $x \geq 0$ then return x mod y
       If $x < 0$ then z ← (−x) mod y, return y − z.

7.   Arithmetic Z_7 is equivalent to arithmetic mod 7. x + 3 = 6. Add 4 to both sides and you get x + 7 = 10. But in mod 7, 7 is 0 and 10 is 3. Thus we have **x = 3**. x + 5 = 3. Add 2 to both sides to get **x = 5**. x − 4 = 5. Add 4 to both sides to get **x = 2**. x + 1 = 6 implies x + 7 = 12 which implies **x = 5**.