CSCI 310 – 02 (Fall 2019)

Programming Foundations

Lab #16: A Top–Down Approach to the Stern–Brocot Number System

DUE: Fri, Oct 18, 11:59pm (`turnin` time)

Specifications

The *Stern–Brocot tree* is a mathematically beautiful way for constructing the set of all nonnegative fractions $\frac{m}{n}$ where m and n are relatively prime. The idea is to start with two fractions $(\frac{0}{1}, \frac{1}{0})$ and then repeat the following operations as many times as desired:

- Insert $\frac{m+m'}{n+n'}$ between two adjacent fractions $\frac{m}{n}$ and $\frac{m'}{n'}$

For example, the first step gives us one new entry between $\frac{0}{1}$ and $\frac{1}{0}$:

$$\frac{0}{1}, \mathbf{\frac{1}{1}}, \frac{1}{0}$$

the next step gives us two more:

$$\frac{0}{1}, \mathbf{\frac{1}{2}}, \frac{1}{1}, \mathbf{\frac{2}{1}}, \frac{1}{0}$$

and the next gives us four more:

$$\frac{0}{1}, \mathbf{\frac{1}{3}}, \frac{1}{2}, \mathbf{\frac{2}{3}}, \frac{1}{1}, \mathbf{\frac{3}{2}}, \frac{2}{1}, \mathbf{\frac{3}{1}}, \frac{1}{0}.$$

If we continue with this process, then we will get 8, 16, and so on. The entire list can be regarded as an infinite binary tree structure whose top levels look like this:

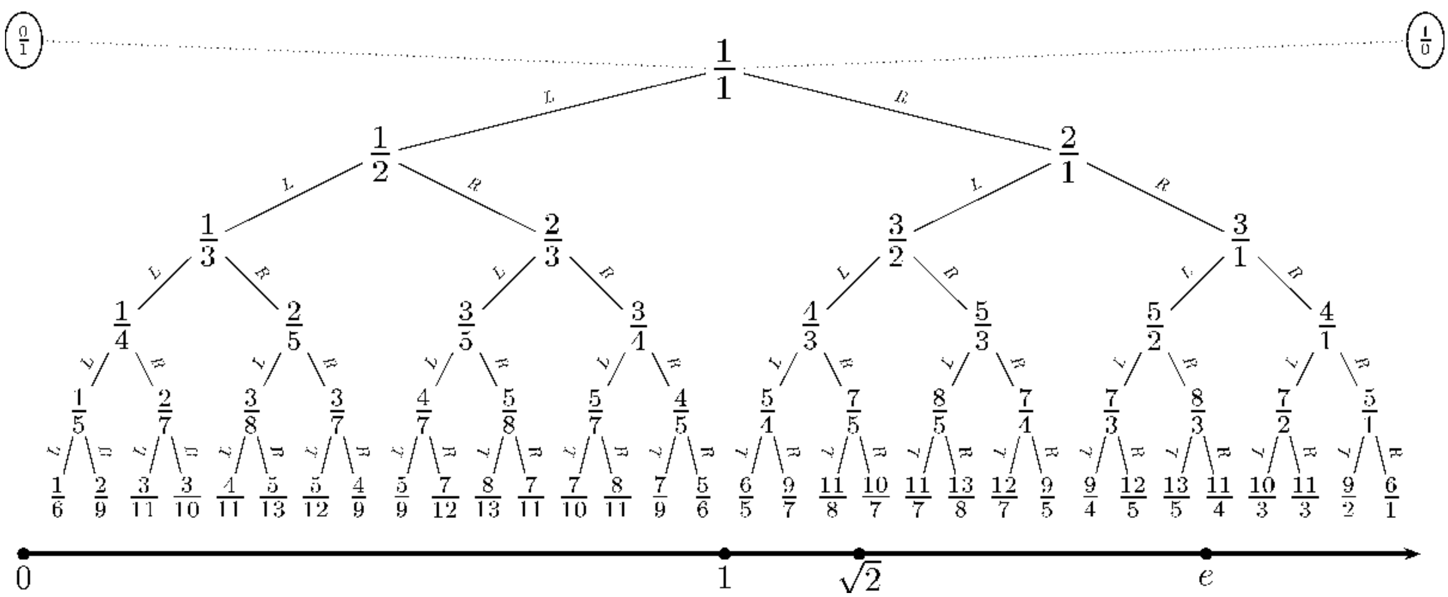

Figure 1: A portion of the *Stern–Brocot tree*

The construction procedure preserves order, and we could not possibly get the same fraction in two different places. We can, in fact, regard the *Stern–Brocot tree* as a number system for representing rational numbers, because each positive, reduced fraction occurs exactly once. Using the characters `'L'` and `'R'` to stand for going down to the left or right branch as we proceed from the root of the tree to a particular fraction, a string of `L`'s and `R`'s uniquely identifies a place in the tree. For example, the string `LRRL` means that starting at the root with $\frac{1}{1}$ we go `L`eft down to $\frac{1}{2}$, then `R`ight to $\frac{2}{3}$, then `R`ight to $\frac{3}{4}$, then `L`eft to $\frac{5}{7}$. Thus, we can consider the string `LRRL` to be a representation of the rational number $\frac{5}{7}$. Every positive fraction gets represented in this way as a unique string of `L`s and `R`s.

There is one small problem: The fraction $\frac{1}{1}$ corresponds to the empty string, and we need a notation for that. We will denote this I, because that looks like the number 1 and it will stand for "identity."

Write a driver program that takes a positive rational fraction and determines its representation in the Stern–Brocot number system.

Input

The input contains multiple test cases. Each test case consists of a line containing two positive integers m and n where m and n are relatively prime. The input terminates with a test case containing two 1s for m and n, representing the "identity," and this case must not be processed.

Sample Input

```
5 7
878 323
1 1
```

Output

For each test case from the input, output a line containing the representation of the given fraction in the Stern–Brocot number system.

Sample output, corresponding to the sample input above, is provided below.

Sample Output

```
LRRL
RRLRRLRLLLLRLRRR
```

Other requirements

There are a number of solutions to this problem online and most of them use recursion in a bottom–up approach (start with a leaf, use recursion to get to the root, then generate the notation as you recurse back. To prevent you from just using those, for any credit you are required to solve this problem like a tree search in *top–down* fashion that starts at the root. Make sure you understand how this works before you start coding!

Submission

Your submission will consist of the following file(s), submitted using the **turnin** facility.

- `lab16.cpp` – code with the `main()` driver function