OJ 10077

The Stern-Brocot Number System

Prof. Edson Alves – UnB/FGA

Problema

The Stern-Brocot tree is a 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 $\frac{1}{0}$,

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

and the next gives two more:

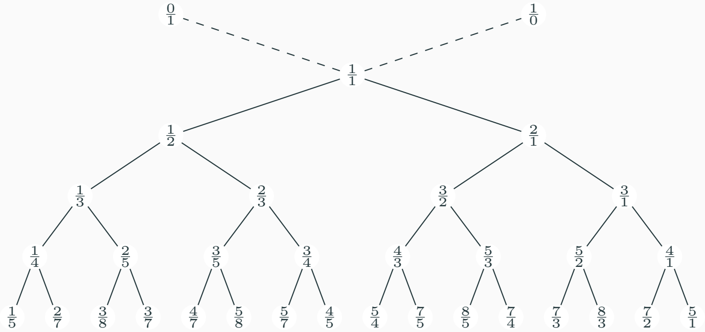
$$\frac{0}{1}, \frac{1}{2}, \frac{1}{1}, \frac{2}{1}, \frac{1}{0}$$

Problema

The next gives four more,

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

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



Problema

The construction preserves order, and we couldn't 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. Let's use the letters '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; then a string of L's and R's uniquely identifies a place in the tree. For example, LRRL means that we go left from $\frac{1}{1}$ down to $\frac{1}{2}$, then right to $\frac{2}{3}$, then right to $\frac{3}{4}$, then left to $\frac{5}{7}$. We can consider LRRL to be a representation of $\frac{5}{7}$. Every positive fraction gets represented in this way as a unique string of L's and R's.

Problema

Well, actually there's a slight problem: The fraction $\frac{1}{1}$ corresponds to the empty string, and we need a notation for that. Let's agree to call it `I`, because that looks something like 1 and it stands for "identity".

In this problem, given a positive rational fraction, you are expected to represent it in *Stern-Brocot number system*.

Entrada e saída

Input

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

Output

For each test case in the input file output a line containing the representation of the given fraction in the *Stern-Brocot number system*.

Exemplo de entradas e saídas

Exemplo de Entrada

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

Exemplo de Saída

```
LRRL
RRLRRLRLLLLRLRRR
```

Solução

- A árvore apresentada no exemplo e as sequências iniciais permitem a observação que a árvore com raiz $\frac{1}{1}$ é uma árvore binária de busca
- Este fato permite o uso da busca binária para determinar, a cada elemento, se é preciso seguir à esquerda ou à direita
- A busca se encerra quando o número a ser representado é atingido
- Deste modo, a solução tem complexidade $O(H)$, onde H é o número de elementos que compõem a representam da fração n/m no sistema apresentado

Solução $O(H)$

```cpp
#include <bits/stdc++.h>

using namespace std;
using ll = long long;
using ii = pair<ll, ll>;

bool operator<(const ii& m, const ii& n)
{
    return m.first * n.second < m.second * n.first;
}

ll gcd(ll a, ll b)
{
    return b ? gcd(b, a % b) : a;
}

string stern_brocot(const ii& n)
{
    ii m(1, 1), L(0, 1), R(1, 0);
    ostringstream os;
```

Solução $O(H)$

```cpp
    while (m != n)
    {
        if (n < m)
        {
            os << "L";
            R = m;
        } else
        {
            os << "R";
            L = m;
        }

        m = ii(L.first + R.first, L.second + R.second);
        ll d = gcd(m.first, m.second);
        m.first /= d;
        m.second /= d;
    }

    return os.str();
}
```

Solução $O(H)$

```cpp
int main()
{
    int p, q;

    while (cin >> p >> q)
    {
        if (p == 1 and q == 1)
            break;

        auto ans = stern_brocot(ii(p, q));

        cout << ans << '\n';
    }

    return 0;
}
```