Greedy Algorithm for Egyptian Fraction

Every positive fraction can be represented as sum of unique unit fractions. A fraction is unit fraction if numerator is 1 and denominator is a positive integer, for example 1/3 is a unit fraction.Such a representation is called Egyptial Fraction as it was used by ancient Egyptians.

Following are few examples:

```
Egyptian Fraction Representation of 2/3 is 1/2 + 1/6
Egyptian Fraction Representation of 6/14 is 1/3 + 1/11 + 1/231
Egyptian Fraction Representation of 12/13 is 1/2 + 1/3 + 1/12 + 1/156
```

We can generate Egyptian Fractions using Greedy Algorithm. For a given number of the form ‘nr/dr’ where dr > nr, first find the greatest possible unit fraction, then recur for the remaining part. For example, consider 6/14, we first find ceiling of 14/6, i.e., 3. So the first unit fraction becomes 1/3, then recur for (6/14 – 1/3) i.e., 4/42.

Below is C++ implementation of above idea.

```
// C++ program to print a fraction in Egyptian Form usin
// Algorithm
#include <iostream>
using namespace std;

void printEgyptian(int nr, int dr)
{
    // If either numerator or denominator is 0
    if (dr == 0 || nr == 0)
        return;

    // If numerator divides denominator, then simple div
    // makes the fraction in 1/n form
    if (dr%nr == 0)
    {
        cout << "1/" << dr/nr;
        return;
    }

    // If denominator divides numerator, then the given
```

```
    // is not fraction
    if (nr%dr == 0)
    {
        cout << nr/dr;
        return;
    }

    // If numerator is more than denominator
    if (nr > dr)
    {
        cout << nr/dr << " + ";
        printEgyptian(nr%dr, dr);
        return;
    }

    // We reach here dr > nr and dr%nr is non-zero
    // Find ceiling of dr/nr and print it as first
    // fraction
    int n = dr/nr + 1;
    cout << "1/" << n << " + ";

    // Recur for remaining part
    printEgyptian(nr*n-dr, dr*n);
 }

// Driver Program
int main()
{
    int nr = 6, dr = 14;
    cout << "Egyptian Fraction Representation of "
         << nr << "/" << dr << " is\n ";
    printEgyptian(nr, dr);
    return 0;
}
```

Output:

```
Egyptian Fraction Representation of 6/14 is
 1/3 + 1/11 + 1/231
```

The Greedy algorithm works because a fraction is always reduced to a form where denominator is greater than numerator and numerator doesn't divide denominator. For such reduced forms, the highlighted recursive call is made for reduced numerator. So the recursive calls keep on reducing the numerator till it reaches 1.

References:

http://www.maths.surrey.ac.uk/hosted-sites/R.Knott/Fractions/egyptian.html