Comp 150/170 Lab

Title: Fraction Calculator Due: start of next class period

In this lab you will complete a program which implements a simple fraction calculator. When the program is completed, it should run according to the sample dialogue given here:

```
Add, subtract, multiply & divide - positive fractions only
Enter '0/0 + 0/0' to quit.

> 1/2 + 3/4
2/4 + 3/4 = 5/4

> 2/16 + 29/32
4/32 + 29/32 = 33/32

> 1/7+1/5
5/35 + 7/35 = 12/35

> 1/2 - 1/3
3/6 - 2/6 = 1/6

> 200/100 * 25/50
2/1 * 1/2 = 1/1

> 1/2 * 3/4
1/2 * 3/4 = 3/8

> 1/2 / 3/4
1/2 / 3/4 = 2/3

> 0/0 + 0/0
```

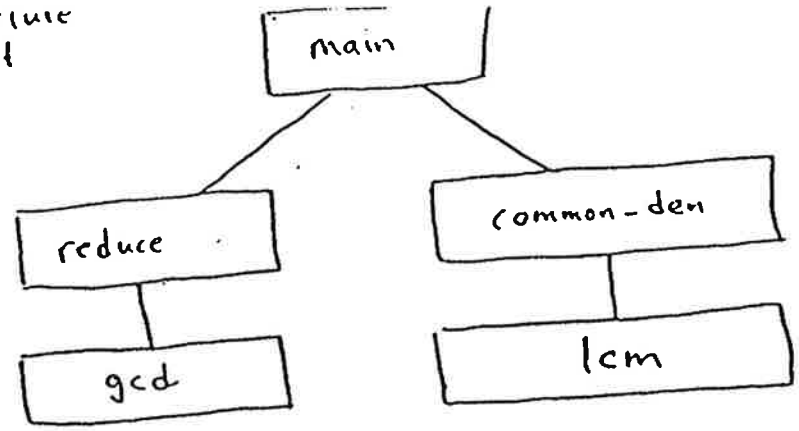
Note that the program will add, subtract, multiply and divide positive fractions only. (For subtraction, this means the larger fraction must be given first so that the answer will be positive also.) All answers will be fractions in lowest terms. For purposes of simplifying this assignment, a whole number such as 5 will be written as 5/1.

On the second page is a "hierarchical structure chart" which gives the desired structure for functions which complete the program. Also given are some diagrams which illustrate what each of the four missing functions is to do. On the third page is a brief description for each function which restates in English what the diagrams are saying. Pages 4 and 5 contain flowcharts you have seen earlier in the semester for the greatest common divisor (gcd) and least common multiple (lcm) algorithms. Page 6 contains the code for the main function which is available on the network so that you need not retype it.

(1) Follow your instructor's instructions for copying FRACTION.CPP from the network to your system.

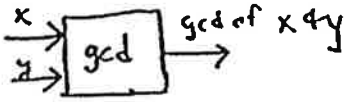
(2) Complete the four functions and turn in FRACTION.CPP using the EASE system

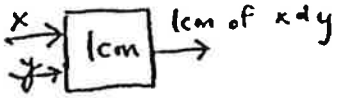
Structure Chart



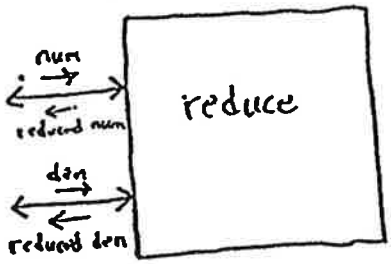
Function Diagrams

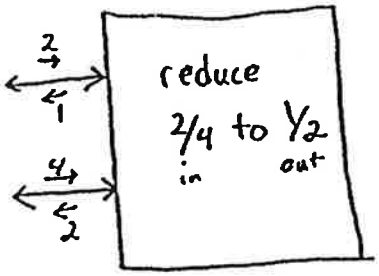
Function

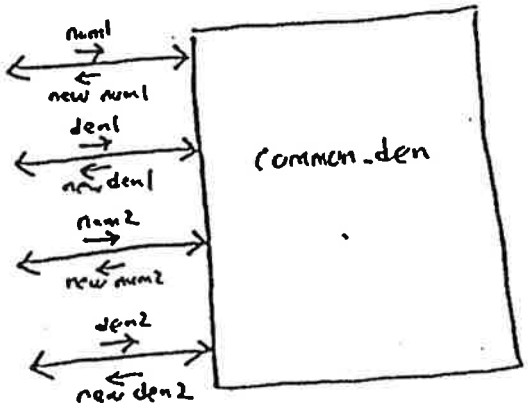


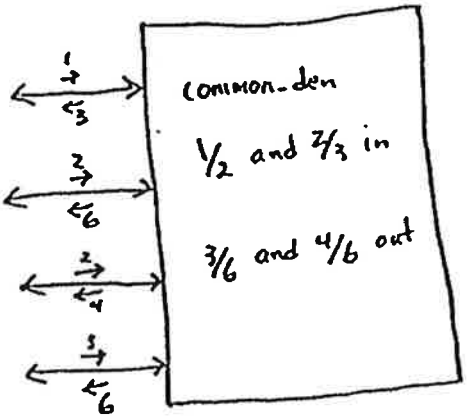


Example









Function Descriptions:

The gcd function receives two integers from the calling program and returns a single integer which is their greatest common divisor. For example:

```
x = gcd(30,42);
```

will result in a value of 6 for x. It is assumed that all integers are positive and are within the normal range for the int data type.

The lcm function receives two integers from the calling program and returns a single integer which is their least common multiple. For example:

```
x = lcm(10,15);
```

will result in a value of 30 for x. It is assumed that all integers are positive and are within the normal range for the int data type.

The reduce function receives two integers from the calling program and it is assumed that the first integer is the numerator and that the second integer is the denominator of a fraction. The reduce function calls the gcd function and uses the information returned to alter both the numerator and denominator so that the values sent back to the calling program are an equivalent fraction to the one received, but in lowest terms. For example:

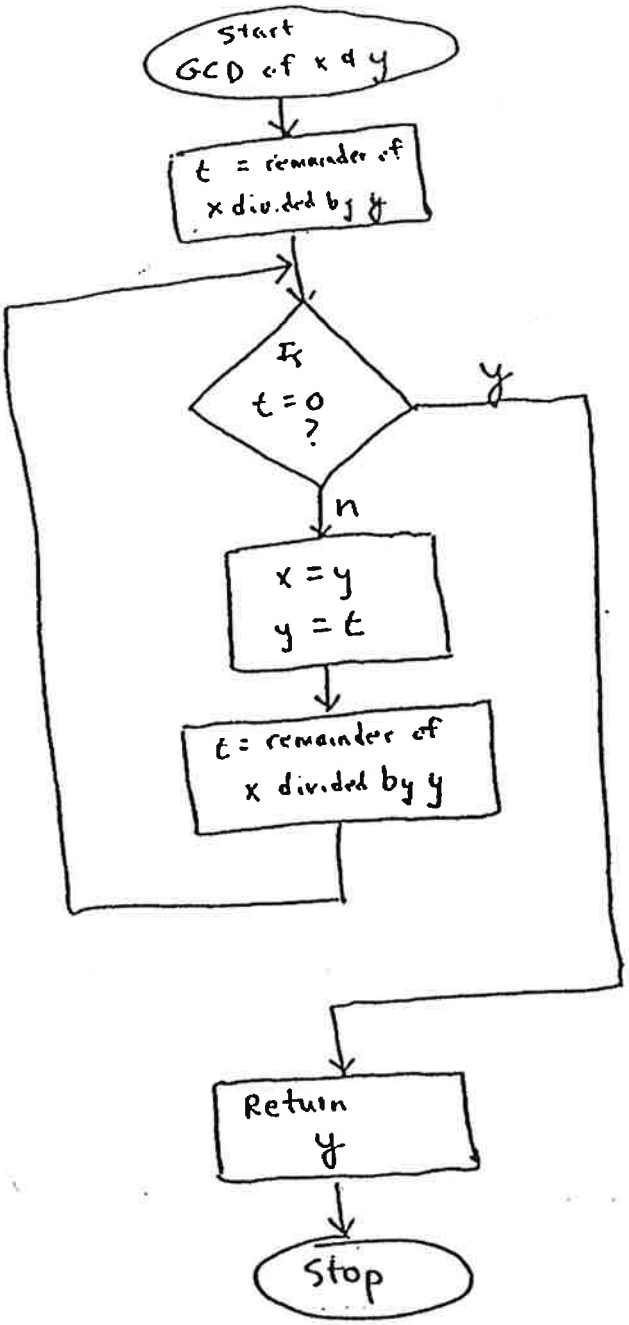
```
x = 2; y = 4;
reduce(x,y);
```

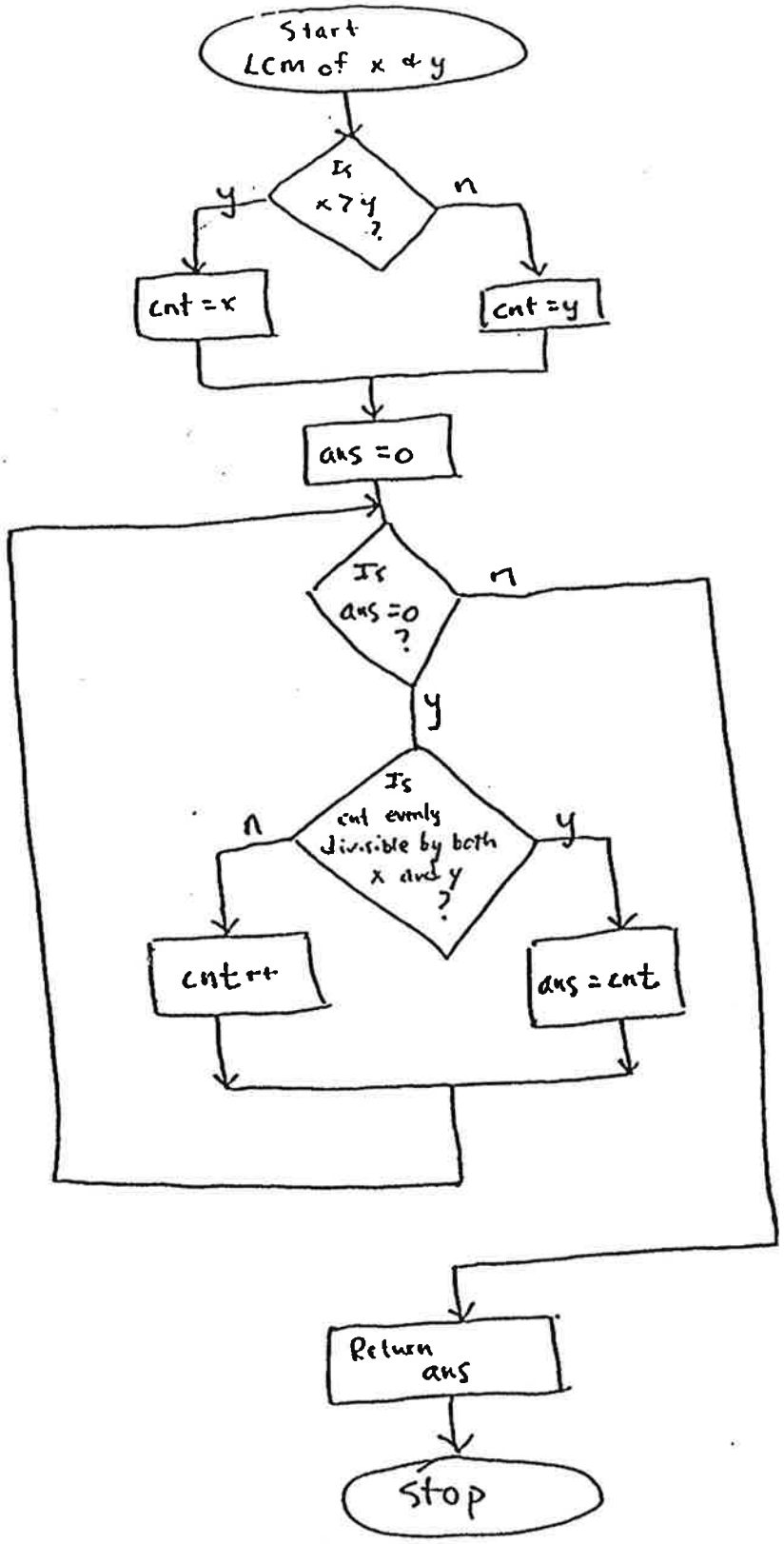
will result in x being changed to 1 and y being changed to 2 since the original data represents 2/4 and the lowest terms version of 2/4 is 1/2.

The common_den function receives four integers from the calling program and it is assumed that the first two integers represent the numerator and denominator of one fraction and that the second two integers represent the numerator and denominator of a second fraction. This function will call the lcm function to find a common denominator and then alter both fractions so that the values returned to the calling program are equivalent fractions with a common denominator. For example:

```
n1 = 1; d1 = 2;
n2 = 2; d2 = 3;
common_den(n1,d1,n2,d2);
```

will result in n1 being changed to 3 and d1 being changed to 6. It will also result in n2 being changed to 4 and d2 being changed to 6. This is due to the fact that the original data represents 1/2 and 2/3 and the commonly denominated equivalents are 3/6 and 4/6.


Start
GCD of x & y
t = remainder of x divided by y
If t = 0 ?
y
n
x = y
y = t
t = remainder of x divided by y
Return y
Stop


Start
LCM of x & y
Is x > y ?
y
n
cnt = x
cnt = y
ans = 0
Is ans = 0 ?
n
y
Is cnt evenly divisible by both x and y ?
n
y
cnt++
ans = cnt
Return ans
Stop

```
#include <iostream>
using namespace std;

//  Missing functions go here!

void main()
{
  int num1, den1, num2, den2, newnum, newden;
  char slash1, slash2, op;

  cout << "\nFraction Calculator\n\n";
  cout << "Add, subtract, multiply & divide - positive fractions only\n";
  cout << "Enter '0/0 + 0/0' to quit.\n";

  // Input will be assumed to be in correct form for simplification
  // Input data before loop in case they want to exit right away

  cout << "\n> ";
  cin >> num1 >> slash1 >> den1 >> op >> num2 >> slash2 >> den2;
  while (num1+den1+num2+den2 > 0)
  {
     // Reduce both fractions to keep integers as small as possible
     reduce(num1,den1);
     reduce(num2,den2);
     switch (op) {
       case '+':
      // Find common denominator and add
      common_den(num1,den1,num2,den2);
      newnum = num1 + num2;
      newden = den1;
      break;
       case '-':
      // Find common denominator and subtract
      common_den(num1,den1,num2,den2);
      newnum = num1 - num2;
      newden = den1;
      break;
       case '*':
      // Multiply numerators and multiply denominators
      newnum = num1 * num2;
      newden = den1 * den2;
      break;
       case '/':
      // Invert and multiply
      newnum = num1 * den2;
      newden = den1 * num2;
     }
     // Reduce the answer to lowest terms
     reduce(newnum,newden);

     // Output the results
     cout << num1 << "/" << den1 << " " << op << " ";
     cout << num2 << "/" << den2 << " = ";
     cout << newnum << "/" << newden << endl;

     // Input data for next iteration of loop
     cout << "\n> ";
     cin >> num1 >> slash1 >> den1 >> op >> num2 >> slash2 >> den2;
  }
}
```