CSCI 1730 - Programming Assignment 2 - 50 pts.
Due Date: Thursday, February 19, 2015

What you need to turn in:

- **Code listing:** A printed copy of your C++ code solution for each problem. Remember to include your name on every page you turn in. Be sure to follow the "Code Style Guidelines" specified in the "Assignment Information and Guidelines" handout that was distributed on the first day of class (these guidelines are also available at the class web page).
- **Code files:** E-mail me a copy of your C++ code. Copy/paste all code into one e-mail message – please, do not send attachments. **Enter your name in the subject line of your email.**
- **Working in Pairs:** If you want to work with one other student in our class on this assignment, this is acceptable provided that both members of the pair make a contribution to the solution. If you decide to work in a pair, turn in only one copy of the solution – clearly identify the name of each pair member on everything that you turn in.
- **Late Assignments:** Assignments are due **by the end of class** on the specified due date (both the paper copies and the e-mail copies). Assignments turned in late will be assessed a 20% penalty per day late.

1. (10 pts.) Recall that a quadratic equation is one of the form $ax^2 + bx + c = 0$, where the coefficients *a*, *b*, and *c* are real numbers and $a \neq 0$. The solutions to a quadratic equation can found using the quadratic formula:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Write a program that will prompt for and read the values of the coefficients *a*, *b*, and *c*, and then calculate and display the solutions of the quadratic equation having those coefficients.

Program requirements:

- Use floating point variables to hold the values of *a*, *b*, and *c*.
- Your program should generate an error message and force reentry if the user enters a value of zero for *a*.
- Your program must use the predefined `sqrt` function – remember to add the include directive for the `cmath` library.
- Your program should work whether the quadratic equation has two real number roots, one real number root, or two complex number roots. **Hint:** Calculate the value of the discriminant (the expression inside the square root) and store it in a separate variable. Then, generate the appropriate output display based on that variable having a positive, or zero, or negative value.

Here are examples of a user's interaction with the program (user input in **bold**):
Run 1:

```
Quadratic equation solver:
Enter a: 0
a cannot be zero - reenter
Enter a: 1
Enter b: 3
Enter c: 2
There are two real solutions: -1 and -2
```

Run 2:

```
Quadratic equation solver:
Enter a: 1
Enter b: -6
Enter c: 9
There is one real solution: 3
```

Run 3:

```
Quadratic equation solver
Enter a: 1
Enter b: 0
Enter c: -2
There are two real solutions: 1.41421 and -1.41421
```

Run 4:

```
Quadratic equation solver:
Enter a: 1
Enter b: 2
Enter c: 10
There are two complex solutions: -1+3i and -1-3i
```

2. (10 pts.) Euclid's method for finding the greatest common divisor (GCD) of two positive integers is given by the following algorithm:

a) Divide the larger number by the smaller and retain the remainder.
b) Divide the smaller number by the remainder, again retaining the remainder.
c) Continue dividing the prior remainder by the current remainder until the remainder is zero, at which point the last nonzero remainder is the greatest common divisor.

For example, find the GCD of 72 and 114:

114/72 = 1 with remainder 42
72/42 = 1 with remainder 30
42/30 = 1 with remainder 12
30/12 = 2 with remainder 6
12/6 = 2 with remainder 0

So, the GCD of 72 and 114 is 6, the last nonzero remainder.

Using Euclid's method, write a C++ function, `gcd`, that will take in two positive integers, determine the GCD of the integers, and return the result.

Then, modify the fraction calculator problem from the last assignment (problem 3) to include your `gcd` function and use it to produce reduced fraction results. That is, after you calculate the numerator and denominator of a fraction arithmetic operation, find their GCD and then divide each by the GCD to obtain the reduced fraction numerator and denominator.

Here is an example of a user's interaction with the new fraction calculator program that includes use of the `gcd` function to reduce the fraction results (user input in **bold**):

```
Enter first fraction: 3/4
Enter operation: -
Enter second fraction: 1/6
Difference = 7/12
Continue (y or n)? y
Enter first fraction: 2/3
```

```
Enter operation: *
Enter second fraction: 5/6
Product = 5/9
Continue (y or n)? y
Enter first fraction: 6/7
Enter operation: /
Enter second fraction: 8/9
Quotient = 27/28
Continue (y or n)? n
```

3. (10 pts.) Write a C++ program that lets the user play the game of Rock, Paper, Scissors against the computer. The program should work as follows.

a) The program first determines the computer's play in the game, as follows: A random number in the range of 1 through 3 is generated. If the number is 1, then the computer has chosen "paper". If the number is 2, then the computer has chosen "rock". If the number is 3, then the computer has chosen "scissors".
b) The program then prompts the user to enter her "rock", "paper", or "scissors" choice and then reads it, using the same number scheme as above.
c) The program then displays both choices (in words) and the result of the play. A winner is selected according to the following game rules:
    - If one player chooses rock and the other player chooses scissors, then rock wins. (The rock smashes the scissors.)
    - If one player chooses scissors and the other player chooses paper, then scissors wins. (Scissors cuts paper.)
    - If one player chooses paper and the other player chooses rock, then paper wins. (Paper covers rock.)
    - If both players make the same choice, that play is a tie.

The program should repeatedly ask the user if she wants to play again and continues play (i.e., repeats steps 1-3, above) until she selects to end the game. After the user decides to end play, the program should display the overall results of the game – that is, how many times the user won, how many times the computer won, and how many ties there were.

Program requirements:
Your program should be modular and must contain the following functions:

- Function `main` will be the program driver – it will repeatedly call the following functions to get the user's play selection, randomly generate the computer's play selection, determine the winner of a play, and tally the results. The repetition should continue until the user selects to exit the program, at which time it will call a function to display the final results.
- `getcompselection` – this function will use the `rand` function to determine the computer's play selection and return it.
- `getuserselection` – this function will prompt for and read the user's play selection and return it.
- `playresults` – this function will take the computer's and user's play selections, as well as counters for determining the total play results, and will determine the play result and display it, and will also increment the counters appropriately.
- `finalresults` – this function will take the total play results counters and generate the final display.

Extra code needed:

a) You need to add these to your `#includes` list:

```
#include <cstdlib>
#include <ctime>
```

b) You need to include this call to the `srand` function after your variable declarations in `main` in order to "seed" the `rand` function:

```
srand(time(NULL));
```

Here is the output from a sample run of the program (user input in **bold**):

```
It's time to play ROCK, PAPER, SCISSORS!
What is your play?
Paper (1), Rock (2), Scissors (3): 1
You played paper and I played scissors
I win.

Play again? (y or n): y
What is your play?
Paper (1), Rock (2), Scissors (3): 3
You played scissors and I played scissors
The play was a tie.

Play again? (y or n): y
What is your play?
Paper (1), Rock (2), Scissors (3): 2
You played rock and I played scissors
You win.

Play again? (y or n): y
What is your play?
Paper (1), Rock (2), Scissors (3): 2
You played rock and I played rock
The play was a tie.

Play again? (y or n): n
Final results of our match:
You won 1 time.
I won 1 time.
There were 2 ties.
Bye, bye...
```

4. (10 pts.) Write a C++ program that can be used to calculate and display the area and the perimeter of three different two-dimensional geometric shapes: a circle, a rectangle, and a triangle.

Program requirements:
Your program should be modular and must contain the following functions:

- Function `main` will be the program driver – it will repeatedly display a menu of shapes to the user asking the user to select one and then call the appropriate functions to prompt the user for input of dimension(s), read the dimension(s), calculate the area and perimeter, and display the results. The repetition should continue until the user selects to exit the program.
- Three overloaded `getdim` functions that will ask the user to enter the needed shape dimension(s) from the user and return the dimension(s) to `main`.
- Three overloaded `area` functions that will take in the shape dimension(s) and calculate and return the shape area.
- Three overloaded `perimeter` functions that will take in the shape dimension(s) and calculate and return the shape perimeter.
- One `display` function that will take a shape indicator value and the shape area and perimeter values and display the labeled results.

Note: Your program should use Heron's formula (given below) for finding the area of a triangle (the formula uses the triangle's side lengths instead of the base and the height to find the area).

Note: Your program should also include a global named constant `PI` initialized to 3.141592654 that will be needed by your circle area and perimeter functions.

Area and Perimeter Formulas:

Circle: $A = \pi r^2$ $P = 2\pi r$ where *r* is the circle radius.

Rectangle: $A = lw$ $P = 2l + 2w$ where *l* and *w* are the rectangle length and width.

Triangle: $A = \sqrt{s(s-a)(s-b)(s-c)}$, where $s = \frac{a+b+c}{2}$ $P = a + b + c$

where *a*, *b*, and *c* are the lengths of the sides of the triangle.

Here is the output from a sample run of the program (user input in **bold**):

```
AREA/PERIMETER CALCULATOR
Select a shape:
Circle (1)  Rectangle (2)  Triangle (3) Exit (4)
Enter selection => 1
Enter circle radius: 10
Circle area = 314.159 and perimeter = 62.8319

AREA/PERIMETER CALCULATOR
Select a shape:
Circle (1)  Rectangle (2)  Triangle (3) Exit (4)
Enter selection => 2
Enter rectangle length: 3.5
Enter rectangle width: 4.5
Rectangle area = 15.75 and perimeter = 16

AREA/PERIMETER CALCULATOR
Select a shape:
```

```
Circle (1)  Rectangle (2)  Triangle (3) Exit (4)
Enter selection => 3
Enter 1st side of triangle: 3
Enter 2nd side of triangle: 4
Enter 3rd side of triangle: 5
Triangle area = 6 and perimeter = 12

AREA/PERIMETER CALCULATOR
Select a shape:
Circle (1)  Rectangle (2)  Triangle (3) Exit (4)
Enter selection => 4
Bye...
```

5. (10 pts.) The Fibonacci sequence is 0, 1, 1, 2, 3, 5, 8, 13, 21, … where the first two terms are 0 and 1, and each term thereafter is the sum of the two preceding terms. Write a C++ program that repeatedly prompts for and reads a positive value *n*, calls a **recursive** function `fibonacci` to calculate the n^{th} number in the Fibonacci sequence, and then displays the number. For example, if *n* = 9, then the program would display 21.

Here is the output from a sample run of the program (user input in **bold**):

```
Fibonacci number generator
Which one do you want (0 to exit)? 10
Fibonacci number #10 is 34
Which one do you want (0 to exit)? 20
Fibonacci number #20 is 4181
Which one do you want (0 to exit)? 40
Fibonacci number #40 is 63245986
Which one do you want (0 to exit)? 0
```