CSC / CIS 175

Problem Solving and Programming - I

University of Michigan-Flint
Department of Computer Science, Engineering, and Physics (CSEP)

Fall 2013

October 27, 2013

Homework 7

(100 points)

due by November 6, Wednesday 8:00am

Remarks:

- No emailed homeworks will be accepted.
- Only submission is via the BB system.
- No late submissions will be accepted.
- Individual assignment. No collaboration or group work is permitted.

Questions for the deliverable:

1. Write a program to display the area of a rectangle after accepting its length and width from the user by means of the following functions:
   `getLength` – ask user to enter the length and return it as double
   `getWidth` – ask user to enter the width and return it as double
   `getArea` – pass width and length, compute area and return area
   `displayArea` – pass area and display it in this function.

   Expected Output:

```
 (i)
Enter the length: 29
Enter the width: 34
```

```
Rectangle Data
--------------
Length: 29
Width: 34
Area: 986

  (ii)
Enter the length: 5.98
Enter the width: 29.345

Rectangle Data
--------------
Length: 5.98
Width: 29.345
Area: 175.483
```

2. A common application in computer graphics, as well as in many different subfields in science and engineering, is to compute the distance between two points which involves Pythagorean theorem and hence a square root calculation. However, as accuracy of the square root increases the time to compute it also increases. Often, a very coarse granularity approximation of the square root is good enough. Use Babylonian method (special case of Newton's method – Newton's method is a method in calculus for determining a zero of a function) and Halley approximation to find crude estimates of square root of numbers. You may need to do some research about these algorithms.

   (a) **Babylonian Method:**

      i. Start with an initial estimate. A rough estimate is to use the number of digits of the number to the left of the decimal point and raise 3 to the power of that number of digits. For example, if the user enters 45346.984, the number of digits to the left of the decimal point is 5 and the initial estimate is $x_0 = 3^5 = 243$,

      ii. Iterate through the following series until a desired level of accuracy is achieved:

$$x_n = \frac{x_{n-1} + \left(\frac{number}{x_{n-1}}\right)}{2} \tag{1}$$

      where *number* is the user-entered value for the square-root calculation, and n is the current iteration number of the approximation process.

   (b) **Halley Method:** (a) Start with an initial estimate. A rough estimate is to use the number of digits of the number to the left of the decimal point and raise 3 to the power of that number of digits. For example, if the user enters 45346.984, the number of digits to the left of the decimal point is 5 and the initial estimate is $x_0 = 3^5 = 243$, (b) Iterate through the following series until a desired level of accuracy is achieved:

$$x_n = \frac{x_{n-1} * \left(x_{n-1}^2 + 3 * number\right)}{3 * x_{n-1}^2 + number} \tag{2}$$

Babylonian method is simpler but converges to the real square root slower (quadratic). Halley method is more involved but converges to the square root faster (cubical). In other words, with the same number of iterations, the accuracy of Halley method will be better.

Write a program that will have three functions:
`babylonianSquareRoot`,
`halleySquareRoot`, and
`wholeNumberOfDigits`
that will find the square root of a number entered by the user. The latter function will find the number of digits of a number passed to it to the left of the decimal point.

Expected Output:

```
(i)
Enter a number to find the square root : 81

Square root approximation by using the Babylonian Method is
9.0000000000000000000

Square root approximation by using Halley Method is
9.0000000000000000000

Square root by cmath function sqrt is
9.0000000000000000000
(ii)
Enter a number to find the square root : 9922338833

Square root approximation by using the Babylonian Method is
99610.93731614013086073101

Square root approximation by using Halley Method is
99610.93731614015996456146

Square root by cmath function sqrt is
99610.93731614014541264623

(iii)
Enter a number to find the square root : 12345678.434523565

Square root approximation by using the Babylonian Method is
3513.64176240600318124052

Square root approximation by using Halley Method is
3513.64176240600272649317

Square root by cmath function sqrt is
```

```
3513.641762406003181240 52
```

3. A company pays its employees as managers (who receive a fixed weekly salary), hourly workers (who receive a fixed hourly wage for up to the first 40 hours they work and time-and-a-half, i.e. 1.5 times their hourly wage, for overtime hours worked), commission workers (who receive $250 plus 5% of their gross weekly sales), or pieceworkers (who receive a fixed amount of money for each item they produce-each pieceworker in the company works only on one type of item). Write a program to compute the weekly pay for each employee. You do not know the number of employees in advance. Each type of employee has its own pay code: Managers have code 1, hourly workers have code 2, commission workers 3 and pieceworkers 4. Use a switch to compute each employee's pay according to his/her paycode. Within the switch, prompt the user to enter the appropriate facts your program needs to calculate each employee's pay according to the paycode.

   Sample Output:

```
Enter paycode (-1 to end): 100
Invalid number! Please try again.

Enter paycode (-1 to end): 1
Manager selected.
Enter weekly salary: 244
The manager's pay is $244.00
Enter the next paycode (-1 to end): 2
Hourly worker selected.
Enter the hourly salary: 12.30
Enter the total hours worked: 39
Worker's pay is $479.70
Enter the next paycode (-1 to end): 3
Commission worker selected.
Enter gross weekly sales: 8000
Commission worker's pay is $650.00
Enter the next paycode (-1 to end): 4
Pieceworker selected.
Enter number of pieces: 2345
Enter wage per piece: 1.75
Pieceworker's pay is $4103.75
Enter the next paycode (-1 to end): -1

Good-bye!
```

4. A K-12 science and engineering competition has 5 judges, each of whom awards a score between 0 and 10 to each project. Fractional scores, such as 8.3, are allowed. A projects final score is determined by dropping the highest and lowest score received, then averaging the 3 remaining scores. Write a program that uses this method to calculate a contestants

score. It should include the following functions: a) void getJudgeData() should ask the user for a judges score, store it in a reference parameter variable, and validate it. This function should be called by main once for each of the 5 judges. b) void calcScore() should calculate and display the average of the 3 scores that remain after dropping the highest and lowest scores the performer received. This function should be called just once by main, and should be passed the 5 scores.

The last two functions, described below, should be called by calcScore, who uses the returned information to determine which of the scores to drop. c) int findLowest() should find and return the lowest of the 5 scores passed to it. d) int findHighest() should find and return the highest of the 5 scores passed to it.

Input Validation: Do not accept judge scores lower than 0 or higher than 10.

Expected Output:

```
Enter score between 0 and 10: 8.4
Enter score between 0 and 10: 3.9
Enter score between 0 and 10: 5.6
Enter score between 0 and 10: 5.9
Enter score between 0 and 10: 8.0

This contestant's talent score is 6.50
```

5. Calculate the value of π from the infinite series

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \cdots \tag{3}$$

Print a table that shows the approximate value of π for 100, 200, 300, 400, ..., 1500 terms of the series.

Expected Output:

```
pi with   100 terms  = 3.131592903558552
pi with   200 terms  = 3.136592684838814
pi with   300 terms  = 3.138259329515588
pi with   400 terms  = 3.139092657496011
pi with   500 terms  = 3.139592655589782
pi with   600 terms  = 3.139925988080528
pi with   700 terms  = 3.140164082890081
pi with   800 terms  = 3.140342654078072
pi with   900 terms  = 3.140481542821614
pi with  1000 terms  = 3.140592653839790
pi with  1100 terms  = 3.140683562868528
pi with  1200 terms  = 3.140759320401133
pi with  1300 terms  = 3.140823422934350
```

```
pi with  1400 terms  = 3.140878367966611
pi with  1500 terms  = 3.140925986997196
```

6. Write a program that inputs an odd number from 1 to 35 and prints a diamond of asterisks. You may only use output statements that print a single asterisk (*) or a single blank. Maximize the use of repetition with nested for loops and minimize the number of output statements. Check the input to make sure it is not an even number nor it is outside the range. Use do...while to ensure acceptable input value.

Expected Output:

```
(i)
 Enter an odd number for rows (1 to 35) for the diamond of asterisks : 22
22 is not an odd number. Please re-try.
Enter an odd number for rows (1 to 35) for the diamond of asterisks : -4
-4 is outside the range. Please re-try.
Enter an odd number for rows (1 to 35) for the diamond of asterisks : 39
39 is outside the range. Please re-try.
Enter an odd number for rows (1 to 35) for the diamond of asterisks : 5
  *
 ***
*****
 ***
  *
```

(ii)

```
Enter an odd number for rows (1 to 35) for the diamond of asterisks : 23
           *
          ***
         *****
        *******
       *********
      ***********
     *************
    ***************
   *****************
  *******************
 *********************
***********************
 *********************
  *******************
   *****************
    ***************
     *************
      ***********
       *********
        *******
         *****
          ***
           *
```

(iii)

```
Enter an odd number for rows (1 to 35) for the diamond of asterisks : 35
                  *
                 ***
                *****
               *******
              *********
             ***********
            *************
           ***************
          *****************
         *******************
        *********************
       ***********************
      *************************
     ***************************
    *****************************
   *******************************
  *********************************
 ***********************************
*************************************
 ***********************************
  *********************************
   *******************************
    *****************************
     ***************************
      *************************
       ***********************
        *********************
         *******************
          *****************
           ***************
            *************
             ***********
              *********
               *******
                *****
                 ***
                  *
```

7. `Extra credit (5%)`: Redo **hw6-q4** with only

```
cout << "*";
```

   not using any

```
cout << " ";
```

   statements.

8. `Extra credit (5%)`: Redo **hw6-ec** with only

```
cout << "*";
```

   not using any

```
cout << " ";
```

   statements.

Deliverables:

1. Source Code: (**.cpp** file) that must start with a comment block similar to the following:

```
/*************************************************************************
** Author          : Suleyman Uludag
** Program          : hw1, q1
** Date Created     : September 15, 2013
** Date Last Modified : September 16, 2013
** Usage            : No command line arguments
**
** Problem:
Accept the following information from the user (keyboard):
- Hw1, hw2 and hw3 (out of 100)
- Midterm (out of 100)
- Final exam (out of 100)
Calculate the total grade out of 100 based on the following grading scale:
Hws          -->    30% (10% each)
Midterm      -->    30%
Final Exam   -->    40%
** Pseudocode:
** 1)
** 2)
*************************************************************************/
```

2. Executable (.exe file under windows). You must explicitly state the platform of your executable (such as Linux, etc.) if it is not Windows. Please name your file by using the question number: **hw1-q1.exe** (for Windows)

3. Screenshot of your app. For screenshot, you can use the following free program on windows:

   `http://www.wisdom-soft.com/downloads/setupscreenhunterfree.exe`

   For Linux/Unix, there are many alternatives. I personally like shutter.
   File naming convention example:

   `hw1_q1.png` (or .jpg or another graphics format)

4. You must zip all the above three files into ONE .zip file and submit your assignment by the deadline on moodle system. Name your file as Lastname-Firstname-hw#.zip. For example, **Uludag-Suleyman-hw1.zip**

   For generating .zip file, you may use the following free software on Windows:

   `http://www.7-zip.org/download.html`

   Linux/Unix has many built-in.