CSE1222 Programming & Algorithms for Engineers **Autumn 2016**

Programming Assignment 3: Loops

1 Setting up the Programming Environment

1. Create a new directory (folder) called `hw3` and move to that directory.

2. Copy the files from the directory `/class/cse1222/9643/hw3` into the current directory by typing:

```
cp /class/cse1222/9643/hw3/hour_glass_template.cpp hour_glass.cpp
cp /class/cse1222/9643/hw3/hour_glass_solution.exe .
cp /class/cse1222/9643/hw3/factors_solution.exe .
```

2 Work by Yourself

All lab and programming assignments are to be done by yourself. You may discuss labs or assignments with other students in the class but DO NOT LOOK AT ANYONE'S CODE OTHER THAN YOUR OWN. Needless to say, you should not share or copy anyone else's code.

3 Program Requirements

Effective commenting and tabbing will affect your grade. The "style" of your program should follow the style of the sample programs in the course notes. Your program should have the file name, your name, creation and last modification dates and a brief description of the program in the comments at the top of the program. The declaration of every variable should have a comment.

Write a program that prints an hour glass using the character '#'. The user will enter the number of '#'s on the top row and then the number of rows from the top to the middle row. For instance, an hour glass with 7 '#'s in the top row and with 3 rows from the top row to the middle row looks like:

```
#######
 #####
  ###
 #####
#######
```

No row is allowed to have less than two '#'s. You will write code to check for this.

Write a program that prompts for an integer greater than one that represents the low end of a range and a second integer greater than or equal to the first integer that represents the high end of a range. The program will print the factors of integers in this range in descending order.

For example, if the user enters 2 for the low end of the range and 6 for the high end of the range, then the program will output:

```
6: 1, 2, 3, 6
5: 1, 5
4: 1, 2, 4
3: 1, 3
2: 1, 2
```

For both problems, write your code incrementally. This means write only enough code to solve a particular step in the algorithms provided below and then compile and test that portion thoroughly before writing your solution to the next algorithmic step. Test your code on small input values first and handle boundary and error cases early in coding your solution.

1. Program `hour_glass.cpp` using the starter template provided:

   (a) Prompt and read in the number of '#'s in the top row;

   (b) If the number of '#'s is less than three, print the error message "Size of the top row must be at least three.", and prompt again for the number of '#'s in the top row. Repeat until the user enters a valid number. (Use a `while` loop.)

   (c) Prompt and read in the number of rows from the top row to the middle row.

   (d) If the number of rows is invalid then print "Invalid number of rows." and prompt again for the number rows. An invalid number of rows would be a number less than 1 or a number that leads to a row in the hour glass with less than two '#'s. Repeat until the user enters a valid number of rows. (Use a `while` loop.)

   (e) Display an empty line.

   (f) Use `for` loops to display the hour glass. Your outer `for` loop will iterate over the number of rows times. For each row use one nested `for` loop to display blanks (the top row contains no blanks) and another nested `for` loop to display the characters '#'.

   First, test your code on a top row size of 3 '#'s and one row. Your program should display only one total row. Second, test your code on a top row size of 4 '#'s and two rows. Continue testing your code for larger input values and combinations.

   Test your code on invalid input to ensure that only valid values are used when displaying the hour glass.

2. Program `factors.cpp` (no starter template provided):

   (a) Prompt and read in an integer value for the low end of the range. This value must be greater than one. Print the message "Number must be greater than 1." if an invalid value is input. Repeat this until a valid value is input.

   (b) Prompt and read in an integer value for the high end of the range. This value cannot be lower than the low end value. For example, if the user enters 5 for the low end value and 2 for the high end value then print the message "Number must be greater or equal to 5.". Repeat this until a valid value is input.

   (c) Use nested `for` loops to display all the factors of each integer from the high end to the low end in increasing order. The outer `for` loop will iterate from the high end to the low end integer and the nested `for` loop will display all the factors for a particular integer. An integer x is a factor of y if y mod x equals 0. Note that 1 and y are always (trivial) factors of y.

4 Sample Program Interaction

Test your programs against the program `hour_glass_solution.exe` and `factors_solution.exe`. The input and output of your program should match the input and output of the solution. Test your programs thoroughly with many input values.

5 Program Submission

Submit your files `hour_glass.cpp` and `factors.cpp` in the hw3 drop box on Carmen. DO NOT submit the file `a.out`.

If you do not submit your program, you will receive zero credit for the homework.

If your program does not compile and run you will receive zero credit for the homework.