McMaster University

COMPENG 2SH4
Principles of Programming
LAB 1 (Sept. 16 – 27)
Instructor: Mohamed Hassan
Fall 2019

This lab is worth 5% of the course mark.
The *Bonus Question* is worth additional 1% of the course mark.
The number in square brackets at the beginning of each question shows the number of points the question is worth. The total number of points is 50 (+ 10 bonus).

You will receive a mark only for the codes that are demonstrated in front of a TA by the end of the lab. With that in mind, please complete as many problems as you can before you come to lab. ***Work not completed by the end of the lab will not be marked.*** *TAs will leave the lab at 5:20pm sharp.*

Additionally, you have to submit on Avenue a text file (with extension ***txt****) for each exercise, containing the source code for that exercise. Naming instructions for the files are provided on Avenue. The online submission has to be done by the end of the lab session.*

General Requirements on Your Programs

A. Your programs should be written in a good programming style, including instructive comments and well-formatted, well-indented code. Use self-explanatory names for variables as much as possible. (~5% of the mark)
B. When outputting the results, include in the output an explanatory message. When inputting values display a message prompting the user to input the appropriate values (~10% of the mark)

Lab Questions

For this lab you are allowed to use from the C standard library only functions for input and output *(e.g. printf(), scanf())*

1. [5] Write a program to compute the sum of all numbers that are multiples of 4, between 30 and 1000, in three different ways: with a for loop, a while loop and a do-while loop. After each loop print the value on the screen.
2. [5] Write a program which reads n integers from the standard input (i.e., keyboard), and prints the smallest number. Prompt the user first to input the value of n, which has to be positive. Check if the

input is valid and in case it is not, keep asking the user for a correct input. Then prompt the user to input each number.

3. [5] For this exercise you should be able to write a logical expression (i.e., with logical operators) which checks if some integer x consists of exactly 5 digits. Ex: 30498 and -14004 are 5-digit numbers, while 1098, -1 and 34 are not.

   Write a program that reads integers input by the user and outputs those integers of 5 digits. The program should end when 0 is the input. The following is an example of the execution of your program ("↓" is Enter or the Return key)

   ```
   Please input an integer (0 to end): 90807↓
   Your input is a five-digit number: 90807
   Please input an integer (0 to end): -19253↓
   Your input is a five-digit number: -19253
   Please input an integer (0 to end): 98↓
   Please input an integer (0 to end): 0↓
   Thank you for using my software. Bye!
   ```

4. [5] (Adapted from Textbook, Ex. 30, pp. 217) Write a program that takes a student's average as an input, which is a floating point value, and prints 4 if the average is in the range 90-100, 3 if it is in the range 80-89, 2 if it is in the range 70-79, 1 if it is in the range 60-69 and 0 if the average is between 0 and 59. If the average is not in the range 0-100, the program should print a message to signal an invalid input.

5. [7] Calculate the value of π from the infinite series

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \ldots$$

   Write a program which reads in a positive integer n (i.e., n>=1) and calculates the value of π by adding up the first n terms of the above series.

6. [7] (Pythagorean Triples – from Textbook, ex. 27, pp. 161) A right triangle can have sides that are all integers. The set of three integer values for the sides of a right triangle is called a Pythagorean triple. These three sides must satisfy the relationship that the sum of the squares of two of the sides is equal to the square of the hypotenuse. Find all Pythagorean triples for side1, side2 and the hypotenuse all no larger than 400, with side1<= side2. Use a triple-nested for loop that simply tries all possibilities. This is an example of the "brute-force" approach. Your program should print each triple on a separate line, and the total number of triples at the end.

7. [8] (Adapted from Textbook, Ex. 26, pp. 217) A positive integer number is said to be a ***perfect number*** if its positive factors, including 1 (but not the number itself), sum to the number. For example, 6 is a perfect number because 6=1+2+3. Write a program that prints all perfect numbers smaller than or equal to some integer m (m > 1) input by the user.

   Note: Assume that x and y are two positive integers. Then x is a **factor** of y if the remainder of the division of y by x is 0. For instance, 5 is a factor of 15, but not of 36.

8. [8] **a)** Write a program that prints backwards a seven-digit positive integer. For example, if the integer is 9806593, the program should print 3956089. You are not allowed to use any function of C standard library other than scanf() and printf(). You are not allowed to use arrays either. For the case when the integer ends with 0, the number printed cannot have leading 0's (Eg: input 3412400; output 42143). **Hint:** Use the division and remainder operators to separate the number into its individual digits. **b)** Modify your program so it prints backwards any positive integer, not necessarily a six-digit one.

9. **Bonus Question [10]**: A binary sequence (consisting of only 0's and 1's) which contains long runs of 0's and of 1's can be efficiently compressed using run-length coding. This method represents the binary sequence as a sequence of integers containing the sizes of the runs of 0's and 1's in the binary sequence. More specifically, the integer sequence is determined as follows. The first integer represents the number of 0's at the beginning of the sequence until the first 1. The second integer represents the number of 1's until the following 0. The third integer represents the number of subsequent 0's until the next 1, and so on.

   Ex1: 00000110000000111 leads to 5,2,7,3

   Ex2: 11111000011100 leads to 0,5,4,3,2

   Finally, this sequence of integers will be converted into a binary stream (for instance by using the binary representation of each integer). In this exercise you are asked to write two programs:

   a) the first one to convert a binary sequence to the corresponding sequence of integers;

   b) the second one to do the reversed conversion from a sequence of positive integers into a binary stream.

   Each program has to read its input from a text file (named, for instance, inputfile1.txt, respectively inputfile2.txt) and has to write the result into another text file

   The first input file contains a positive integer representing the total number of digits (actually bits) in the binary sequence. Then the digits in the sequence follow, with consecutive digits separated by one blank space. The output file should contain the corresponding sequence of integers with consecutive integers separated by one blank space.

   inputfile1 for Ex. 1: 17 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 1 1

   outputfile1 for Ex. 1: 5 2 7 3

   The second input file starts with a positive number representing the number of integers in the sequence followed by a blank. Then the sequence of integers follows, with consecutive integers separated by one blank space. The second output file must have the same format as inputfile1.

   inputfile2 for Ex. 1: 4 5 2 7 3

   To create an input file complying to the above specified format use simple text editors such as Notepad or Textpad. When using Netbeans you have to include the input file into the corresponding project by doing the following. First save the file in the directory containing the project. In the

Netbeans Navigator select your project and right-click on “Important Files”. Then right-click on “Add item to Important Files” and select the input file.

For instructions on reading/writing text files see the document entitled “Notes for LAB 1 Bonus Question”.

Note that you do not need to use arrays in order to write this program.