Due before:	11:59 PM (before Midnight) on Thursday June 6, 2019

CSE-278: Introduction to Systems Programming

Homework #2

Due: Thursday June 6 2019 before 11:59 PM

Email-based help Cutoff: 5:00 PM on Wed, June 5 2019

Maximum Points: 50

Submission Instructions

This homework assignment must be turned-in electronically via Canvas. For this part of the homework you should supply answers to each question in this document. Prior to commencing work on this part, rename this MS-Word document using the naming convention `MUid_hw2.docx` (example: ahmede_`hw2.docx`). Once you have completed answering the questions save this document as a PDF file (don't just rename the document; that is not the correct way to save as PDF) and upload it to Canvas with C++ source. **Do not submit archive files such as: zip,7zip,rar,tar,tar.gz,rpm,deb etc.**

Copy pasting from online resources is plagiarism. Instead you should understand concepts and explain them using your own words!

Objective

The objective of this part of the homework is to:

1. Review the basic C++ concepts
2. Practice developing C++ programs similar to questions that will appear on job interviews & exams.

Name: Ce Zhang

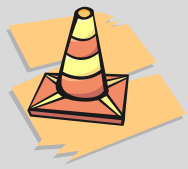
PART 1

Required reading

Prior to answering the questions in this homework briefly review the following chapters from the E-textbook (link in Syllabus on Canvas) "C++ How to Program" (all students have free access to the electronic book):

- Chapter 1 (Introduction)
- Chapter 3.1, 3.2 (String & methods)
- Chapter 5.1 – 5.10 (loops & switch)
- Chapter 2 (Expressions & I/O)
- Chapter 4 (if-statements)
- Chapter 6 (functions/methods)

Due before:	**11:59 PM (before Midnight) on Thursday June 6, 2019**



Although the Safari E-books are available to all students there are only a limited number of concurrent licenses to access the books. Consequently, do not procrastinate working on this homework or you may not be able to access the E-books due to other users using them.

1. Briefly (1 sentence each) state 3 advantages of using the C++ standard library (as inferred from Chapter 1.10)? [**1 points**]

   1.Support of Generic programming

   2.Deliver fast, efficient, and robust code

   3.Easy to use

2. Briefly (about 2 sentences each) discuss 2 key philosophical differences between C++ and Java [**2 points**]

   1. C++ is platform-dependent.
      Java is platform-independent.

   2. Free unused memory
      Java has garbage collection. It is thread which can remove unused objects from the memory and which is automatically generated by JVM. So memory is automatically freed when not used any more.

      C++ has no garbage collection. Thus, all unused reserved memory needs to be freed when no longer used to avoid memory leaks.

3. Briefly (1 to 2 sentences for each phase) describe the 6 phases involved in creating and running a C++ program (see Chapter 1.9) [**3 points**]

   Phase 1: Creating a Program： editing a file with an editor program.
   Phase 2: Preprocessing: give the command to compile the program
   Phase 3: Compiling a C++ Program: the compiler translates the C++ program into machine-language code.

Phase 4 linking: C++ programs typically contain references to functions and data defined elsewhere.
Phase 5 loading: before a program can be executed, it must first be placed in memory
Phase 6: Execution: the computer, under the control of its CPU

4. What is the difference between the Internet and the World Wide Web (WWW) as described in Chapter 1.12 [**1 point**]?

   The internet is bigger than WWW. And it is actually a network of networks, connecting a lot of computers around the world ... The World Wide Web uses Hypertext Transfer Protocol (HTTP) which is a language used to transmit data - to access information which lives in different networks.

5. Briefly (2 or 3 sentences) describe code "refactoring" and provide an example (see Chapter 1.13) [**1 point**]?

   Refactoring is one process of changing a software system in a specific way that it does not alter these external behaviors of the code. But it improve the internet structure. Example is extract method.

6. What is the significance of the `main` function returning zero? Can it return some other value (Chapter 2.2.)? [**1 point**]

   Return 0 means it is a successful termination, while a non-zero return value means that a failure or unexpected termination.

7. What is the use of the `using namespace std;` declaration in a C++ program (Chapter 2.7)? [**1 point**]

   Using namespace std is a form of scope in C++ which help the cpp file holds its own definitions for variables, functions.

8. What is an unsigned numeric data type? State 2 purposes for using unsigned numbers? [**2 point**]

   An unsigned variable type of int include the zero and positive numbers.
   1. It is the simplest, and most used interpretation internally to the CPU because addresses.
   2. Unsigned can hold a larger positive value.

9. What is the `size_type` or `size_t` data type? [**1 point**]

   size_type as a typedef size_type (Allocator is a template parameter), which for std::allocator<T>::size_type is typically defined to be size_t.

10. What is pseudocode? Where/why would you use pseudocode within (or as part of) the source code of a C++ program? [**1 point**]

    Pseudocode is is a detailed yet readable description of what a computer program or algorithm will do. It is a good example to show the users and communicate with each other.

11. Review the source code in Figure 5.11 (Chapter 5.9). Briefly describe what input(s) will be needed to terminate the `while`-loop on line #24 under Linux? [**1 point**]

    Ctrl+d (Control+d) to terminate the while loop, It will gives the input as EOF which means End Of File.

    If it is not terminated, try this one more time.

12. What is a function prototype and when is it needed (Chapter 6.4)? [**1 point**]

    It is is a declaration of a function that specifies the function's name and type signature. When the function tell what it exactly did.

13. What is a function-call stack? How can you observe the call stack in the IDE used for your course? (Chapter 6.11)? [**1 point**]

    The Call Stack window shows the order in which methods and functions are getting called.

It can help me to let me know where I am now.

14. Given a constant string object `str` (`const std::string str`) what methods can be called on `str`? Give an example of 1 method that can and 1 method that cannot be called on the object `str`? [**2 points**]

All methods except modifiers can be called on constant string object. Because constant value will not be altered.

Examples :

size , length , substr can be called on constant string object.

Assignment (=) , modifiers like insert ,erase , replace and append methods what will not work on constant string.

15. What is the advantage of using constant reference parameters in functions (Chapter 6.13)? [**1 point**]

when you use use 'normal' parameter, will pass the parameter by value and hence creating a copy of that one if you are using const reference, you pass it by reference and the original data will be not copied.

16. What is a common programming error that occurs when returning reference to a local variable (Chapter 6.13)? [**1 point**]

We should never return a reference or a to a local variable. It will be destroyed right when the function returns.
when the function ends, the storage space for the local variables goes away

It will appear to work in some cases because the stack may not be written twice.

17. When is it safe/appropriate to return a reference from a method? [**1 point**]

it's okay to return a reference if the lifetime of the object won't end after the call.

PART 2A: File I/O

Consider the following incomplete C++ program:

```
#include <iostream>
int main()
{
 .
 .
 .
}
```

a. Write a statement that includes the header files *fstream*, *string* and *iomanip* in this program.
ANS:

b. Write statements that declare **inFile** to be an ifstream variable and **outFile** to be an ofstream variable.
ANS:

c. The program will read data from the file **inData.txt** and write output to the file **outData.txt**. Write statements to open both these files, associate inFile *with* inData.txt and associate outFile *with* outData.txt
ANS:

d. Suppose that the file **inData.txt** contains the following data:

```
3   4  5
15.6
Mark Taylor  28
18500 3.5
B
```

The numbers in the first line represent the Side A, Side B and Side C of a Triangle. The number in the second line represents the radius of a circle (Assume that $\pi = 3.1416$). The third line contains the first name, last name, and the age of a person. The first number in the fourth line is the savings account balance at the beginning of the month and the second number is the interest rate per year. The fifth line contains an uppercase letter between B and Z (inclusive). Write statements so that after the program executes, the contents of the outData.txt are shown as below. If necessary, declare additional variables. Your statements should be general enough so that if the content of the input file changes and the program is run again (without editing and recompiling), it outputs the appropriate results.

```
Triangle:
Side A= 3,  Side B  = 4,  Side C = 5, area = 6, perimeter = 12
```

Circle:
Radius = 15.60, area = 764.54, circumference = 98.02

Name: Mark Taylor, age: 28
Beginning balance = $18500.00, interest rate = 3.50
Balance at the end of the month = $18553.96

The character that comes before B in the ASCII set is A

e. Write statements that close the input and output files.
ANS:

f. Write a complete C++ program that tests the statements in parts **a** through **e.**

Hints for Part 2A:

• You should include the math header file and call appropriate routine

• You should calculate the area of Triangle using *Heron*'s formula, defined as follows:

s = (a+b+c)/2 where a, b and c are the side lengths of the triangle

area = $\sqrt{(s(s-a)(s-b)(s-c))}$

• Financial compound interest computed as follows: $F = P(1 + i)^n$ Where F is the future value of money P is the present value of money *i* is the interest rate in percentage (i.e., rate /100), represent as decimal number *n* is the time period

PART 2B File I/O using file streams

When working on programs, you should be periodically saving and compiling your source code to ensure it compiles successfully! In general, when coding you should save and compile your C++ program after every 4 to 5 lines of change.

Exercise/Program requirement: Develop a C++ program that prints the last line of each paragraph in a given text file specified as command-line argument. Paragraphs are separated by one or more blank (i.e., empty) lines.

Procedure: Work on this exercise in the following manner:

1. Use the NetBeans project created in the earlier step.
2. Clean-up the starter code and ensure it compiles and runs successfully
3. In order gain practice working with generic I/O streams, add the following method that works with abstract streams to your starter code:

`void printLastLine(std::istream& is, std::ostream& os)`

The above method is the one that should process lines from input stream is (use std::getline to read lines) and print last line of each paragraph to output stream os.

4. Call the `printLastLine` method (from main) with a suitable std::ifstream to process data from a given text file specified as a command-line argument. Recollect that the file name will be in `argv[1]` in the main method. Use std::cout as the output stream.

Part #3: Submission

- No late assignments will be accepted!
- This work is to be done individually
- The submission file will be saved with the name ***HW2Part2a_yourMUID.cpp***
- The submission file will be saved with the name ***HW2Part2b_yourMUID.cpp***
- Assignment is due Thursday June 6, 2019 before Midnight
- On or before the due time, drop the *electronic copy* of your work in the *canvas*
  Don't forget to Turn in the files! HW2_yourMUID.pdf & HW2_yourMUID*.cpp