Due before:	11:59 PM (before Midnight) on Mon/Tues February 24/25, 2020

Introduction to Systems Programming (Systems I)

Lab #3

Due: Tuesday September 24 2019 before 11:59 PM

Maximum Points: 50

Submission Instructions

This part of the homework assignment must be turned-in electronically via Canvas. Ensure you name this document `MUid_lab3.docx`, where `MUid` is your Miami University unique ID. Complete the method shown for each problem. For each method, you can develop and test them in `NetBeans` and just copy-paste your solutions into this document.

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

General Note: Upload each file associated with homework (or lab exercises) individually to Canvas. Do not upload archive file formats such as zip/tar/gz/7zip/rar etc.

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

Name: Ben Hilger

The objective of this PART A of the homework is to review and recollect the necessary background information and terminology from CSE-174 and CSE-271 (the prerequisite courses). The terminology will be often used in the course.

PART A

1. **What is source code (1 or 2 sentences)?**

Source code is the human-readable version of commands and instructions that can be compiled into an executable computer program.

2. **What is the difference between source code, pseudo code, and an algorithm (answer in 3 separate sentences)**

Source code is able to be compiled into an executable computer program. Whereas pseudocode is used in designing a program and is simply notation regarding the commands and instructions that the executable will run and isn't compliable. And an algorithm is a set of instructions to follow that will complete a certain task such as addition, which is typically then written into source code.

3. **What is a syntax error? How do you detect them? How do you fix syntax errors? (Answer in 3 separate sentences)**

A syntax error is a simple error that comes from mistyping a variable, method or reserved word or putting identifiers in the wrong order. It's very easy to detect them because the compiler will be unable to compile the code and it should return an error stating on which line the syntax error lies. To fix syntax errors once you locate them your simple rewrite or reorder the mistake until the compiler is able to understand what you are trying to do.

4. **What is a semantic error? How do you detect them? How do you fix semantic errors? (Answer in 3 separate sentences)**

A semantic error is a logical error where the code will still compile and execute but the end result or a certain instruction won't make logical sense. This can be detected by analyzing what the program outputs against what it should output if working perfectly to see if there are any discrepancies. To fix a semantic error, you can use a debugger to make breakpoints and analyze where any unnecessary/unwanted changes are taking effect.

5. **What is a debugger? What is a breakpoint? (answer in 2 separate sentences)**

A debugger is a program that tests and corrects errors found in computer programs. A breakpoint is an intentional way of stopping the program under controlled conditions to analyze the state of the program at that point in the program's execution.

6. **Briefly (1 to 2 sentences each) state the 4 key principles of object-oriented programming**

The first principle of object-oriented programming is encapsulation, which deals with the interaction between objects. It allows the programmer to hide and reveal certain methods and implementations to improve reliability of the code.

The second principle of object-oriented programming is abstraction, which deals with the idea that one class doesn't need to know the implementation of other classes methods in order to use it. Instead, it should deal with an abstract or an interface of the class, which means that the concept/idea is not associated with any particular instance.

The third principle of object-oriented programming is inheritance, which is the principle that allows a class to take behaviors or characteristics from another, typically more generic, class and use it as its own.

The fourth principle of object-oriented programming is Polymorphism, which is the idea that one method can come in many forms. It can come in the form of overloading a method or even a subclass overriding the method of a superclass.

7. **Briefly (2 to 3 sentences each) discuss pass-by-value versus pass-by-reference mechanisms for passing parameters to methods. Which approach is preferred based on data type of parameters in C++?**

Pass-by-value is a term used when the parameter in the method is merely a copy of the original value sent. This means that any change made during the method isn't passed to the original value, with the value instead being copied and then sent to the method.

Whereas pass-by-reference actually passes the memory reference of the variable sent as a parameter. This means that the value can be changed by any operation within the method and it would affect the variable. This can save time from copying large values but also runs the risk of making unwanted changes to the original value.

8. **List at least 3 unique properties that can be inferred from data type of a variable?**

One unique property that can be inferred from the data type of a variable is the amount of memory that will be allocated to store that value. Another unique property that can be inferred from the data type is the type of data stored, such as int for integers, or double for decimal numbers. Finally, a third unique property that can be inferred from a data type is the range of the variable. Some variables such as objects can only store one object of that type where the primitive types, such as int can hold from -2_{31} to 2_{31} or an unsigned int can hold. 0 to 2_{32}.

Due before:	11:59 PM (before Midnight) on Mon/Tues February 24/25, 2020

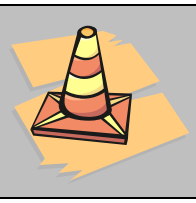
PART B

Objective
The objective of this PART B of the homework is to: • Understand working with `std::vector`. • Practice answering exam style questions

Required reading

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

- Chapter 7.10 (`std::vector`)
- Chapter 14.1 – 14.6 (File I/O)



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. What is quoted text and how do you read quoted text in C++? Explain with a suitable example (other than the one shown in Chapter 14)

Quotes text in C++ is a string that its contents are surrounded by double-quotes (“”). To read the quoted text you do the following:

Suppose you have a name as seen in the file “Names.txt”: “Ben Hilger”
To read this string you code the following:

```
ifstream inFile (“Names.txt”);
string name;
inFile >> quoted(name);
```

2. Assume you have a method called `processLines(std::istream& is, std::ostream& os)` that process line-by-line. Complete the `main` method below to call `processLines` method to process the 3 lines: `"Line #1"`, `"Second Line"`, and `"Last line"`. The output should be written to standard console output stream. (Hint: Use a `std::istringstream`)

```
// Prototype declaration
void processLines(std::istream& is, std::ostream& os);

int main() {

    string lines = "Line #1\nSecond Line\nLast line";
    std::istringstream stream;
    stream.str(lines);

    processLines(stream, cout);

}
```

3. Complete the following method that returns a vector with only even values in the `src` vector. If the `src` vector has values {2, -4, 7, 9, 3, 8} this method should return a vector with values {2, -4, 8}.

```
using IntVec = std::vector<int>;

IntVec evens(const IntVec& src) {


	IntVec evens = {};

	for(int i = 0; i < src.size(); i++) {

		if(src[i] % 2 == 0) {
			evens.push_back (src[i]);
		}
	}

	return evens;
}
```

4. Complete the following method that returns a vector that contains a reverse of the words. For example if `src` is {`"one"`, `"two"`, `"three"`} the method should return a vector with strings {`"three"`, `"two"`, `"one"`}

```
using StrVec = std::vector<std::string>;

StrVec reverse(const StrVec& src) {


	StrVec reversed = {};

	for(int i = src.size()-1; i >= 0; i--) {
		reversed.push_back (src[i]);
	}

	return reversed;



}
```

5. Complete the following method that returns a vector with the first `n` prime numbers. For example, if `n == 7`, this method should return a vector with values {1, 2, 3, 5, 7, 11, 13}.

```
using IntVec = std::vector<int>;
IntVec getPrimes(int n) {

    IntVec primes = {};

    int count = 1;
    while(primes.size() < n) {

        // Checking if prime
        bool isPrime = true;
        for(int i = 2; i <= count/2; i++)  {

            // If there is no remainder and it's not the same
number, then the number isn't prime
            if(count % i == 0 && count != i) {
                isPrime = false;
            }

        }

        // If it is prime then add it to the list
        if(isPrime) {
            primes.push_back(count);
        }

        count++;
    }


    return primes;



}
```

Part C: Submission

- No late assignments will be accepted!
- This work is to be done individually
- The submission file will be saved with the name ***Lab3_yourMUID.pdf***
- The source code will be saved with the name ***Lab3_yourMUID*.cpp***
- Assignment is due Monday/Tuesday February 24/25, 2020 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! Lab3_yourMUID.pdf & Lab3_yourMUID*.cpp