Lab 2

CSE 165: Object Oriented Programming

Spring 2022

(100 points)

This programming assignment has six tasks, complete each task as instructed. To submit your assignment, please organize your code in the folder "Lab2" by placing your code in its corresponding sub-folder. For example, store your code for task 1 *"If-else statements"* in the following directory "Lab2/1/". Then, submit the compressed version of folder Lab2 to CatCourses. Submissions must arrive by one minute before the lab section of week 4 (2/7 – 2/11).

1. **If-else statements (10 Points)**

   Write a program that reads in an integer and outputs "POSITIVE" if the integer is positive and "NEGATIVE" if it is negative, and "ODD" if the integer is odd and "EVEN" if the integer is even.

2. **While loops (10 Points)**

   Modify the program just you wrote for the previous exercise so that it keeps reading integers and classifying them as ODD or EVEN and POSITIVE or NEGATIVE until the user inputs 0. Then the program ends. No classification should be produced for the 0.

3. **Finding Prime Numbers (15 Points)**

   Write a program that reads in an integer N and prints out all the prime numbers strictly less than N. These should be printed one per line.

4. **Formatted Output (15 Points)**

   Write a program that keeps reading in positive integers and for each integer outputs the corresponding value in hexadecimal and binary format. Stop when -1 is read. No output should be generated for -1.

5. **Pointer math (20 Points)**

   Write a program that a user enters two integers. Create a pointer to each of the numbers. Multiply, add, subtract and divide the pointers together (remember to dereference them) and output the result to the console.

6. **Bit Manipulation (30 Points)**

   Write a program that reads in a decimal number "n". Convert the decimal number into binary format and prints out the number in binary format. Now, Write following functions to get, set and clear bit at the position "index" and display the corresponding output given below. Please note that index 0 is the right most(least significant) bit.

   int getBit(int n, int index)

   Retrieve a bit from a number "n" in binary format at position "index"

Input: number n and position index.

Output: bit at position "index"

int setBit(int n, int index)

Set a bit at position "index"

Input: number "n" and position "index".

Output: Output the binary number after a bit is set at position "index"

int clearBit(int n, int index)

Clear a bit at position "index". If a bit is 1, change it to 0, otherwise keep the bit unchanged.

Input: number "n" and position "index".

Output: Output the binary number after a bit is cleared at position "index"

Example1:

Consider input number $n = 20$ and $index = 0$

Binary representation of 20 is: 10100

Get bit at position 0: 0

Binary representation after setting bit at position 0: 10101

Binary representation after clearing bit at position 0: 10100

Example2:

Consider input number $n = 30$ and $index = 2$

Binary representation of 30 is: 11110

Get bit at position 2: 1

Binary representation after setting bit at position 2: 11110

Binary representation after clearing bit at position 2: 11010

Note: To display a decimal number into 8 bit binary format, use std::bitset$< 8 >(n)$.

Example: std::bitset$< 8 >(4)$ if $n = 4$. The output will be 00000100