Object-Oriented Programming (Lab Exercise)
Spring 2024
Software Engineering Department, NED University of Engineering and Technology

Week #3

Objective: The objective of this lab is to provide students with an opportunity to practice and reinforce their understanding of repetition and conditional structures in C++

Introduction: In this lab session, students will dive into the world of programming logic and decision-making.
Repetition and conditional structures are fundamental concepts in programming that allow them to control the flow of their code and execute certain blocks of instructions repeatedly or conditionally based on specific criteria.
Throughout this lab, the students will explore various techniques for implementing repetition using for loops, while loops, and conditional structures such as if, if-else, and if-else-if statements. These constructs are essential tools in a programmer's toolkit and are used extensively in real-world applications to automate tasks and solve complex problems.

By practicing these concepts through hands-on exercises and challenges, the students will enhance their problem-solving skills and develop a deeper understanding of how to write efficient and effective C++ code. So, let's get started and embark on this journey to master the art of programming logic!

Conditional Structures:
In OOP, conditional structures like if-else statements and switch-case statements are used to control the flow of execution based on certain conditions. These structures allow you to make decisions within your code and execute specific blocks of code based on whether certain conditions are true or false.

If-Else Statements: An if-else statement evaluates a condition and executes a block of code if the condition is true, and another block of code if the condition is false. This structure allows for branching based on a single condition.

```
if (condition) {
  // Code to execute if condition is true
} else {
  // Code to execute if condition is false
}
```

Switch-Case Statements: A switch-case statement evaluates an expression and compares it to multiple values (cases). It executes the block of code associated with the first case that matches the expression. Switch-case statements are often used when there are multiple possible conditions to check.

```
switch (expression) {
  case value1:
    // Code to execute if expression is equal to value1
    break;
  case value2:
    // Code to execute if expression is equal to value2
    break;
  default:
    // Code to execute if expression doesn't match any case
}
```

Loops: Loops are used to execute a block of code repeatedly as long as a specified condition is true. In OOP, loops such as for, while, and do-while are commonly used to iterate over collections, perform repetitive tasks, and control program flow.
For Loops: A for loop is used to iterate over a range of values or elements in a collection. It typically consists of an initialization step, a condition for continuing the loop, and an update step to modify the loop variable.

```
for (initialization; condition; update) {
  // Code to execute repeatedly as long as condition is true
}
```

While Loops: A while loop executes a block of code as long as a specified condition is true. It continuously evaluates the condition before each iteration.

```
while (condition) {
  // Code to execute repeatedly as long as condition is true
}
```

Do-While Loops: A do-while loop is similar to a while loop, but it executes the block of code at least once before checking the condition for subsequent iterations. This ensures that the block of code is executed at least once, regardless of whether the condition is initially true or false.

```
do {
  // Code to execute at least once, then repeatedly as long as
condition is true
} while (condition);
```

In OOP, conditional structures and loops are fundamental tools for controlling the flow of execution, making decisions, and performing repetitive tasks within classes and objects. They help in writing flexible, efficient, and expressive code that can adapt to various scenarios and conditions.

Q.1) Any character is entered by the user; write a program to determine whether the character entered is a capital letter, a small case letter, a digit or a special symbol. The following table shows the range of ASCII values for various characters.

Characters	**ASCII Values**
A – Z	65 – 90
a – z	97 – 122
0 – 9	48 – 57
special symbols	0 - 47, 58 - 64, 91 - 96, 123 – 127

Q.2) Write a program to calculate the monthly telephone bills as per the following rule:

Minimum Rs. 200 for upto 100 calls.
Plus Rs. 0.60 per call for next 50 calls.
Plus Rs. 0.50 per call for next 50 calls.
Plus Rs. 0.40 per call for any call beyond 200 calls.

Q3) Write a program to check the strength of a password entered by the user. The strength of the password is determined based on the following criteria:

- Minimum length of 8 characters.
- Contains at least one uppercase letter, one lowercase letter, one digit, and one special character.

Q4) Write a program to encrypt and decrypt a text file using a simple encryption algorithm. The encryption algorithm involves shifting each character by a fixed number of positions in the ASCII character set.

Q5) Write a C++ program to generate the multiplication table of a given number.

Q6) Write a C++ program to create a simple menu-driven calculator that performs basic arithmetic operations (addition, subtraction, multiplication, division).

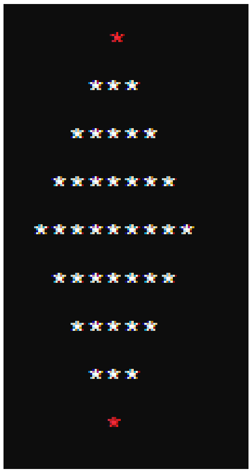
Q7) Write a C++ program to generate the Fibonacci series up to a given number of terms.

Q8) Write a C++ program to implement a number guessing game where the user tries to guess a randomly generated number.

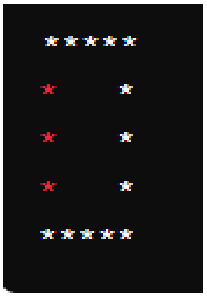
Q9) Write a C++ program to implement a simple rock, paper, scissors game between the user and the computer.

Q10) Write a C++ program to display the name of the day of the week based on the day number entered by the user.

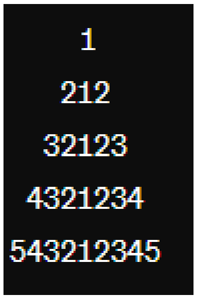
Q11) Write a C++ program to print the following patterns and shapes:



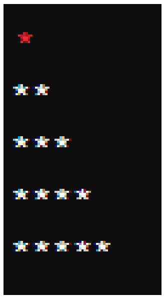
```
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
```



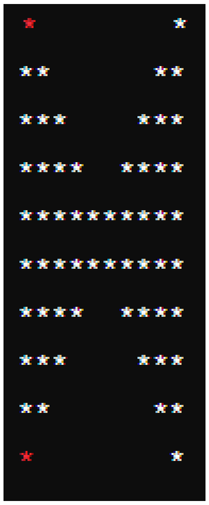
```
*****
*   *
*   *
*   *
*****
```



```
    1
   212
  32123
 4321234
543212345
```



```
*
**
***
****
*****
```



```
*        *
**      **
***    ***
****  ****
**********
**********
****  ****
***    ***
**      **
*        *
```

NED University of Engineering & Technology
Department of Software Engineering
Object Oriented Concepts and Programming

COGNITIVE DOMAIN ASSESSMENT RUBRIC LEVEL C3-PLO3				
SKILL SETS	**EXTENT OF ACHIEVEMENT**			
CRITERIA	**0-1**	**2-3**	**4-5**	**TOTAL**
Understanding of Object-Oriented Concepts	Poor Understanding of Object-Oriented Concepts	Fair Understanding of Object-Oriented Concepts	Good Understanding of Object-Oriented Concepts	
Design of Object-Oriented Solutions	Poor Design of Object-Oriented Solutions	Fair Design of Object-Oriented Solutions	Good Design of Object-Oriented Solutions	
Implementation of Object-Oriented Solutions	Poor Implementation of Object-Oriented Solutions	Fair Implementation of Object-Oriented Solutions	Good Implementation of Object-Oriented Solutions	
Testing and Debugging	Poor Testing and Debugging	Fair Testing and Debugging	Good Testing and Debugging	
Documentation and Comments	Poor Documentation and Comments	Fair Documentation and Comments	Good Documentation and Comments	

Laboratory Session No. ______________ **Date:** ________________________

Weighted CLO (Psychomotor Score)	
Remarks	
Instructor's Signature with Date:	