|  |
| --- |
| **Computer Engineering Department - ITU** |
| **CE101L: Object Oriented Programming Lab** |

|  |  |
| --- | --- |
| **Course Instructor: Usama Bin Shakeel** | **Dated: 15/06/2022** |
| **Teaching Assistant: Aqsa Khalid** | **Semester: Spring 2022** |
| **Lab Engineer: Nadir Abbas** | **Batch: BSCE2021** |

# **Lab 14A. Problem Based Learning in Java CFP**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Roll number** | **Report**  **(out of 100)** | **Scaled to 10** | **Total**  **(out of 10)** |
| Muhammad Abubakar Saif | BSCE21017 |  |  |  |

Checked on: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## **Objective**

The objective of this lab is to observe the basic knowledge of programming classes in C++.

## **Equipment and Component**

|  |  |  |
| --- | --- | --- |
| **Component Description** | **Value** | **Quantity** |
| Computer | Available in lab | 1 |

## **Conduct of Lab**

1. Students are required to perform this experiment individually.
2. In case the lab experiment is not understood, the students are advised to seek help from the course instructor, lab engineers, assigned teaching assistants (TA) and lab attendants.

## **Theory and Background**

**Java** is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. Java is a popular programming language. Java is used to develop mobile apps, web apps, desktop apps, games and much more.

**Lab Task**

**Task A: [Marks: 20]**

Write a program to find the factorial of a number using loop & recursion in java.

|  |
| --- |
| **package** com.example.main; *//if you are running it on ONLINE COMPILER, please REMOVE this line* **import** java.util.Objects; *//imports the class Objects (Use in the program to check that given objects are equal or not)* **import** java.util.Scanner; *//imports the class Scanner (Use in the program for input & output stream)*  **public class** Main {  **//Factorial of a number using Recursion:**  **static int** factorial(**int** num) { *//definition of factorial function (recursion)* **if** (num == 1) { *//terminating condition* **return** 1;  }  **return** num \* *factorial*(num - 1); *//recursive function call* }  **//Factorial of a number using Loop:**  **static int** factorialLoop(**int** num) { *//definition of factorial function (through loop)* **int** mul = 1;  **for** (**int** i = 0; i <= num; i++) { *//loop control statement* mul = num \* mul; *//multiply the number with the products of previous number until iteration will become equal to user-given number* num--; *//decrement in "num"* }  **return** mul; *//returns the "mul" to the MAIN Function* }  **//Main Function:**  **public static void** main(String[] args) { *//MAIN FUNCTION / DRIVER CODE* String rep; *//declares the string type variable* Scanner obj = **new** Scanner(System.***in***); *//declares the instance for Scanner class for input stream* **int** choice; *//declares the string type variable* **do** {  System.***out***.println(**"1. Factorial through Recursion"**); *//prints the given statement* System.***out***.println(**"2. Factorial through Loop"**); *//prints the given statement* System.***out***.println(**"Please Enter Your Choice: "**); *//prints the given statement* choice = obj.nextInt();  **switch** (choice) { *//switch the cases based upon user's selection* **case** 1:  System.***out***.println(**"Enter the value for you want to calculate Factorial: "**); *//prints the given statement* **int** num = obj.nextInt(); *//takes input from user* **int** fact = *factorial*(num); *//calls the function to calculate factorial* System.***out***.print(**"Your Required Factorial is: "**); *//prints the given statement* System.***out***.println(fact); *//prints the factorial of the given number* **break**; *//breaks the case* **case** 2:  System.***out***.println(**"Enter the value for you want to calculate Factorial: "**); *//prints the given statement* **int** num1 = obj.nextInt(); *//takes input from user* **int** fact1 = *factorialLoop*(num1); *//calls the function to calculate factorial* System.***out***.print(**"Your Required Factorial is: "**); *//prints the given statement* System.***out***.println(fact1); *//prints the factorial of the given number* **break**; *//breaks the case*  }  System.***out***.print(**"Do you want to use this program again? (Y/N): "**); *//prints the given statement* rep = obj.next(); *//takes input from user* } **while** (Objects.*equals*(rep, **"Y"**) || Objects.*equals*(rep, **"y"**) || Objects.*equals*(rep, **"yes"**) || Objects.*equals*(rep, **"Yes"**) || Objects.*equals*(rep, **"YES"**) || Objects.*equals*(rep, **"yeS"**));  } } |

**Output:**

**Factorial through Recursion:**

**Graphical user interface, text, application

Description automatically generated**

**Factorial through Loop:**

**Text

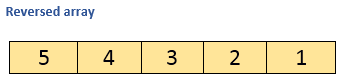
Description automatically generated**

**Task B: [Marks: 20]**

Write a program to reverse the elements of the array in java.

Table

Description automatically generated with medium confidence



|  |
| --- |
| **package** com.example.main; *//if you are running it on ONLINE COMPILER, please REMOVE this line* **import** java.util.Arrays; *//imports the class Array* **import** java.util.Objects; *//imports the class Objects (Use in the program to check that given objects are equal or not)* **import** java.util.Scanner; *//imports the class Scanner (Use in the program for input & output stream)* **public class** Main {  **public static void** main(String[] args) { *//MAIN FUNCTION / DRIVER CODE* String rep; *//declares the string type variable* Scanner obj = **new** Scanner(System.***in***); *//declares the instance for Scanner class for input stream* **int** choice; *//declares the string type variable* **do** {  **int**[] array; *//declares array of int type* **int**[] temp; *//declares array of int type* System.***out***.println(**"Enter the size of the Array: "**); *//prints the given statement* **int** size = obj.nextInt(); *//takes input from user* array = **new int**[size]; *//initialize the size of the array* temp = **new int**[size]; *//initialize the size of the temp* **for** (**int** i = 0; i < size; i++) { *//loop control statement* System.***out***.print(**"Element "** + (i + 1) + **": "**); *//prints the (user-friendly) element number of the array* array[i] = obj.nextInt(); *//takes input from user* temp[((size - 1) - i)] = array[i]; *//saves the array in reverse order in "temp" array* }  System.***out***.println(**"Array Before Reversing: "** + Arrays.*toString*(array)); *//prints the array elements in the line* System.*arraycopy*(temp, 0, array, 0, size); *//copy the arrays* System.***out***.println(**"Array After Reversing: "** + Arrays.*toString*(array)); *//prints the array elements in the line*  System.***out***.print(**"Do you want to use this program again? (Y/N): "**); /*/prints the given statement* rep = obj.next(); *//takes input from user* } **while** (Objects.*equals*(rep, **"Y"**) || Objects.*equals*(rep, **"y"**) || Objects.*equals*(rep, **"yes"**) || Objects.*equals*(rep, **"Yes"**) || Objects.*equals*(rep, **"YES"**) || Objects.*equals*(rep, **"yeS"**));  } } |

**Output:**

**Text

Description automatically generated**

#### **Assessment Rubric for Lab**

**Method for assessment:**

Lab reports and instructor observation during lab sessions. Outcome assessed:

a. Ability to conduct experiments, as well as to analyze and interpret data (P) b. Ability to function on multi-disciplinary teams (A)

c. Ability to use the techniques, skills, and modern engineering tools necessary for engineering practice (P)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Performance metric** | **Task** | **CLO** | **Description** | **Max marks** | **Exceeds expectation** | **Meets expectation** | **Does not meet expectation** | **Obtained marks** |
| 1. Realization of experiment (a) | 1 | 1 | Functionality | 40 | Executes without errors excellent user prompts, good use of symbols, spacing in output. Through testing has been completed (35-40) | Executes without errors, user prompts are understandable, minimum use of symbols or spacing in output. Some testing has been completed (20-34) | Does not execute due to syntax errors, runtime errors, user prompts are misleading or non-existent. No testing has been completed (0-19) |  |
| 2. Teamwork (b) | 1 | 3 | Group Performance | 5 | Actively engages and cooperates with other group member(s) in effective manner (4-5) | Cooperates with other group member(s) in a reasonable manner but conduct can be improved (2-3) | Distracts or discourages other group members from conducting the experiment (0-1) |  |
| 3. Conducting experiment (a, c) | 1 | 1 | On Spot Changes | 10 | Able to make changes (8-10) | Partially able to make changes (5-7) | Unable to make changes (0-4) |  |
| 1 | 1 | Viva | 10 | Answered all questions (8-10) | Few incorrect answers (5-7) | Unable to answer all questions (0-4) |  |
| 4. Laboratory safety and disciplinary rules (a) | 1 | 3 | Code commenting | 5 | Comments are added and does help the reader to understand the code (4-5) | Comments are added and does not help the reader to understand the code (2-3) | Comments are not added (0-1) |  |
| 5. Data collection (c) | 1 | 3 | Code Structure | 5 | Excellent use of white space, creatively organized work, excellent use of variables and constants, correct identifiers for constants, No line-wrap (4-5) | Includes name, and assignment, white space makes the program fairly easy to read. Title, organized work, good use of variables (2-3) | Poor use of white space (indentation, blank lines) making code hard to read, disorganized and messy (0-1) |  |
| 6. Data analysis (a, c) | 1 | 4 | Algorithm | 20 | Solution is efficient, easy to understand, and maintain (15-20) | A logical solution that is easy to follow but it is not the most efficient (6-14) | A difficult and inefficient solution (0-5) |  |
| 7. Computer use (c) | 1 | 2 | Documentation & GitHub Submissions | 5 | Timely (4-5) | Late (2-3) | Not done (0-1) |  |
|  | Max Marks (total): | | | 100 | Obtained Marks (total): | | |  |

Lab Engineer Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_