+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

****

OBJECT ORIENTED PROGRAMMING

LABORATORY

21SC1204-R

STUDENT ID: 2100040024 ACADEMIC YEAR: 2023-24

STUDENT NAME:M.H.G.Subhang

Table of Contents

1. **Session 01: Introductory SessionNA**
2. **Session 02: Basic Control Structure #12**
3. **Session 03: Loop Statements #212**

**4. Session 04: Programs on Recursion#3....23**

**5. Session 05: Programs On Arrays #434**

**6. Session 06: Constructors #546**

**7. Session 07: Method Overloading #658**

**8. Session 08: String Buffer Classes #770**

**9. Session 09: Inheritance #880**

**10 Session 10: Packages #992**

**11 Session 11: Exception Handling#10103**

**12 Session 12: Multi-Threading #11114**

**13 Session 13: swing package-based event driven programming#12125**

**A.Y. 2023-24 LAB/SKILL CONTINUOUS EVALUATION**

| **S.No** | **Date** | **Experiment Name** | **Pre-Lab (10M)** | **In-Lab (25M)** | | | **Post-Lab**  **(10M)** | **Viva Voce (5M)** | **Total**  **(50M)** | **Faculty Signature** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Program/ Procedure (5M)** | **Data and Results (10M)** | **Analysis & Inference (10M)** |
| **1.** |  | **Introductory Session** | **-NA-** | | | | | | | |
| **2.** |  | **Basic Control Structure** |  |  |  |  |  |  |  |  |
| **3.** |  | **Loop Statements** |  |  |  |  |  |  |  |  |
| **4.** |  | **Programs On Recursion** |  |  |  |  |  |  |  |  |
| **5.** |  | **Programs On Arrays** |  |  |  |  |  |  |  |  |
| **6.** |  | **Constructors** |  |  |  |  |  |  |  |  |
| **7.** |  | **Method Overloading** |  |  |  |  |  |  |  |  |
| **8.** |  | **String Buffer Classes** |  |  |  |  |  |  |  |  |
| **9.** |  | **Inheritance** |  |  |  |  |  |  |  |  |
| **10.** |  | **Packages** |  |  |  |  |  |  |  |  |
| **11.** |  | **Exception Handling** |  |  |  |  |  |  |  |  |
| **12** |  | **Multi-Threading** |  |  |  |  |  |  |  |  |
| **13.** |  | **Swing package-based event driven programming** |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

**Following is a step by step guide to download and install Eclipse IDE:**

Step 1) Installing Eclipse.

Step 2) Click on “Download” button.

Step 3) Click on “Download 64 bit” button.

Step 4) Click on “Download” button.

Step 4) Install Eclipse.

Step 5) Click on Run button.

Step 6) Click on “Eclipse IDE for Java Developers”

**Step 7)** Click on “INSTALL” button

**Step 8)** Click on “LAUNCH” button.

**Step 9)** Click on “Launch” button.

**Step 10)** Click on “Create a new Java project” link.

Step 11) Create a new Java Project

Write project name.

Click on “Finish button”.

Organization of the STUDENT LAB WORKBOOK

The laboratory framework includes a creative element but shifts the time-intensive aspects outside of the Two-Hour closed laboratory period. Within this structure, each laboratory includes three parts: Prelab, In-lab, and Post-lab.

1. **Pre-Lab**

The Prelab exercise is a homework assignment that links the lecture with the laboratory period - typically takes 2 hours to complete. The goal is to synthesize the information they learn in lecture with material from their textbook to produce a working piece of software. Prelab Students attending a two-hour closed laboratory are expected to make a good- faith effort to complete the Prelab exercise before coming to the lab. Their work need not be perfect, but their effort must be real (roughly 80 percent correct).

1. **In-Lab**

The In-lab section takes place during the actual laboratory period. The First hour of the laboratory period can be used to resolve any problems the students might have experienced in completing the Prelab exercises. The intent is to give constructive feedback so that students leave the lab with working Prelab software - a significant accomplishment on their part. During the second hour, students complete the In-lab exercise to reinforce the concepts learned in the Prelab. Students leave the lab having received feedback on their Prelab and In-lab work.

1. **Post-Lab**

The last phase of each laboratory is a homework assignment that is done following the laboratory period. In the Post-lab, students analyze the efficiency or utility of a given system call. Each Post-lab exercise should take roughly 120 minutes to complete.

**Experiment Title: Basic Control Structure**

**Aim/Objective:** The aim/objective of basic control structures is to enable the execution of specific sequences of instructions based on certain conditions or criteria.

**Description:**

The student will understand the concept of Basic control structures are fundamental building blocks in programming that allow developers to control the flow of execution in their programs

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

**1) WHAT ARE JAVA TOKENS?**

**Ans:**

**Java tokens are the basic building blocks of a Java program. They are the smallest unit of a program, and they include keywords, identifiers, operators, literals, separators, and comments.**

**2) Explain Data Types?**

**Ans:**

**Primitive data types - includes byte , short , int , long , float , double , boolean and char. Non-primitive data types - such as String , Arrays and Classes.**

**3) What Are Different Operators?**

**Ans:**

Arithmetic Operators , Assignment Operators , Logical Operators, Relational Operators, Unary Operators, Bitwise Operators, Ternary Operators, & Shift Operators.

**4)what are the advantages of oop?**

Ans:

Benefits of OOP language allows to break the program into the bit-sized problems that can be solved easily.

Troubleshooting is easier with the OOP language, Code Reusability, Productivity, Data Redundancy, Code Flexibility, Solving problems, & Security.

**In-Lab:**

**1) To solve the Basic Control Structure**

**.** Java program to read data from user – data types

.Write a Java Program to demonstrate different types of arithmetic operators support by java. Illustrate using an example, the use of arithmetic operators.

.**Procedure/Program:**

**1.** import java.util.Scanner;

public class ReadDataFromUser {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        // Reading data for different data types

        System.out.print("Enter an integer: ");

        int numInt = scanner.nextInt();

        System.out.print("Enter a long integer: ");

        long numLong = scanner.nextLong();

        System.out.print("Enter a floating-point number: ");

        float numFloat = scanner.nextFloat();

        System.out.print("Enter a double-precision number: ");

        double numDouble = scanner.nextDouble();

        scanner.nextLine(); // Consume the newline left by previous nextXxx() methods

        System.out.print("Enter a character: ");

        char charValue = scanner.nextLine().charAt(0); // Read a single character

        System.out.print("Enter a string: ");

        String strValue = scanner.nextLine();

        // Displaying the user input

        System.out.println("Integer: " + numInt);

        System.out.println("Long: " + numLong);

        System.out.println("Floating-point: " + numFloat);

        System.out.println("Double: " + numDouble);

        System.out.println("Character: " + charValue);

        System.out.println("String: " + strValue);

        scanner.close();

    }

}

**2.** import java.util.Scanner;

public class ArithmeticOperatorsDemo {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        // Reading data from the user

        System.out.print("Enter an integer number: ");

        int num1 = scanner.nextInt();

        System.out.print("Enter another integer number: ");

        int num2 = scanner.nextInt();

        System.out.print("Enter a floating-point number: ");

        double num3 = scanner.nextDouble();

        System.out.print("Enter another floating-point number: ");

        double num4 = scanner.nextDouble();

        // Demonstrating arithmetic operators

        int sum = num1 + num2;

        int difference = num1 - num2;

        int product = num1 \* num2;

        int quotient = num1 / num2;

        int remainder = num1 % num2;

        double sumFloat = num3 + num4;

        double differenceFloat = num3 - num4;

        double productFloat = num3 \* num4;

        double quotientFloat = num3 / num4;

        // Displaying the results

        System.out.println("Sum: " + sum);

        System.out.println("Difference: " + difference);

        System.out.println("Product: " + product);

        System.out.println("Quotient: " + quotient);

        System.out.println("Remainder: " + remainder);

        System.out.println("Floating-point Sum: " + sumFloat);

        System.out.println("Floating-point Difference: " + differenceFloat);

        System.out.println("Floating-point Product: " + productFloat);

        System.out.println("Floating-point Quotient: " + quotientFloat);

        scanner.close();

    }

}

* **Data and Results:**

**19-07-2023**

**1.**

**A screenshot of a computer

Description automatically generated**

**2.**

**A screenshot of a computer program

Description automatically generated**

* **Analysis and Inferences:**

**Sample VIVA-VOCE Questions (In-Lab):**

**1.What is oop ?**

Object-oriented programming (OOP) is a style of programming characterized by the identification of classes of objects closely linked with the methods (functions) with which they are associated.

**2.What are the differences between C++ and Java?**

C++ code can be called into C, C++ libraries, or API of operating systems. On the other hand, Java code is only ideal for Java-based libraries. In addition, C++ interacts with hardware more effectively than Java due to its low-level nature and lack of automatic memory management.

**3.Will the program run if we write static public void main?**

If you write static public void instead of public static void then it is perfectly OK. Your Java program will compile and run successfully. It doesn't really make any difference as long as method name comes last and return type of method comes second last

**4.Can Java be said to be the complete object-oriented programming language**

Java is not a fully object-oriented language as it supports primitive data types like int, byte, long, short, etc., which are not objects. Hence these data types like int, float, double, etc., are not object-oriented. That's why Java is not 100% object-oriented.

**5.Can you implement pointers in a Java Program?**

Pointers are a feature associated with the C programming language. They are responsible for providing the memory address in a situation where the programmer directly stores the data. But unfortunately, you won't find any pointers in Java. The closest thing to pointers are references in Java.

**Post-Lab:**

. Java Program to find greatest of three number

. Java program to convert Fahrenheit To Celsius degrees and display messages depends on the temperature

* **Procedure/Program:**

**1.** public class GreatestOfThreeNumbers {

    public static void main(String[] args) {

        java.util.Scanner scanner = new java.util.Scanner(System.in);

        System.out.println("Enter three numbers:");

        int num1 = scanner.nextInt();

        int num2 = scanner.nextInt();

        int num3 = scanner.nextInt();

        int greatest = num1;

        if (num2 > greatest) {

            greatest = num2;

        }

        if (num3 > greatest) {

            greatest = num3;

        }

        System.out.println("The greatest number among " + num1 + ", " + num2 + ", and " + num3 + " is: " + greatest);

        scanner.close();

    }

}

**2.** public class FahrenheitToCelsius {

    public static void main(String[] args) {

        java.util.Scanner scanner = new java.util.Scanner(System.in);

        System.out.print("Enter temperature in Fahrenheit: ");

        double fahrenheit = scanner.nextDouble();

        // Convert Fahrenheit to Celsius

        double celsius = (fahrenheit - 32) \* 5 / 9;

        System.out.println("Temperature in Celsius: " + celsius);

        // Display messages based on the temperature

        if (celsius < 0) {

            System.out.println("It's freezing cold!");

        } else if (celsius >= 0 && celsius < 20) {

            System.out.println("It's cool.");

        } else if (celsius >= 20 && celsius < 30) {

            System.out.println("It's warm.");

        } else {

            System.out.println("It's hot!");

        }

        scanner.close();

    }

}

* **Data and Results:**

**19-07-2023**

**1.**

**A screen shot of a computer

Description automatically generated**

**2.**

**A screen shot of a computer program

Description automatically generated**

* **Analysis and Inferences:**
* The first Java program is designed to find the greatest of three numbers provided by the user. It uses an if-else statement to compare the numbers and determine the maximum. By defining a separate method, the program encapsulates the comparison logic, making it modular and easy to understand.
* The second Java program converts a temperature given in Fahrenheit to Celsius using a simple formula. It then displays the converted Celsius temperature and provides different messages based on the temperature range. This program showcases the use of basic arithmetic operations and conditional statements to handle temperature conversion and temperature-based messaging.
* Both programs demonstrate fundamental concepts in Java programming, including user input handling, conditional statements, and method definition. They are simple yet effective examples of how Java can be used to perform basic calculations and decision-making tasks.

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**

**Experiment Title: Loop Statements**

**Aim/Objective:** The aim of loop statements is to allow programmers to repeat a block of code multiple times. Loop statements provide a way to execute a set of instructions iteratively until a specific condition is met.

**Description:**

Loop statements execute a set of instructions until a specific condition is met or until a certain number of iterations are completed. Here is a description of the three common loop statements

**Pre-Requisites:**

1. Java Development Kit (JDK)

2. Text Editor

3. Integrated Development Environment (IDE)

**Pre-Lab:**

**1) Explain Switch Case With Syntax?**

**Ans:**

Switch Case is a control flow statement that allows you to choose between multiple code blocks based on the value of an expression. The syntax is as follows:

switch (expression) {

  case value1:

    // Code block executed when expression matches value1

    break;

  case value2:

    // Code block executed when expression matches value2

    break;

  // Add more cases as needed

  default:

    // Code block executed if none of the cases match

}

**2) Explain jump and break?**

**Ans:**

**Jump and Break:**

"Jump" refers to changing the flow of program execution to another section, usually achieved with keywords like goto (in some languages). However, excessive use of jumps is discouraged as it can make code hard to read and maintain.

"Break" is used to exit a loop or switch case prematurely. When encountered, the loop stops iterating or the switch case exits immediately, continuing with the code after the loop or switch.

**3) What are pre post exist loop?**

**Ans:**

Pre, Post, and Exit Loops: I'm not familiar with the term "pre, post, and exit loops." It's possible that you meant "pre-test," "post-test," and "exit-controlled" loops, which are loop control structures. Let me explain them briefly:

Pre-Test Loop (while loop):

while (condition) {

  // Code block executed while the condition is true

}

Post-Test Loop (do-while loop):

do {

  // Code block executed while the condition is true

} while (condition);

**In-Lab:**

**1) To solve the Loop Statements**

.Java Program to demonstrate the example of Switch statement  .where we are printing

month name for the given number

. Java program to generate a multiplication table using for- loop

**Procedure/Program:**

import java.util.Scanner;

public class monthselectswitch {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the month number (1-12): ");

        int monthNumber = scanner.nextInt();

        switch (monthNumber) {

            case 1:

                System.out.println("January");

                break;

            case 2:

                System.out.println("February");

                break;

            case 3:

                System.out.println("March");

                break;

            case 4:

                System.out.println("April");

                break;

            case 5:

                System.out.println("May");

                break;

            case 6:

                System.out.println("June");

                break;

            case 7:

                System.out.println("July");

                break;

            case 8:

                System.out.println("August");

                break;

            case 9:

                System.out.println("September");

                break;

            case 10:

                System.out.println("October");

                break;

            case 11:

                System.out.println("November");

                break;

            case 12:

                System.out.println("December");

                break;

            default:

                System.out.println("Invalid month number. Please enter a number between 1 and 12.");

        }

        scanner.close();

    }

}

public class forloopmultiplication {

    public static void main(String[] args) {

        int tableOf = 5;

        int limit = 10;

        System.out.println("Multiplication Table of " + tableOf + ":");

        for (int i = 1; i <= limit; i++) {

            int result = tableOf \* i;

            System.out.println(tableOf + " x " + i + " = " + result);

        }

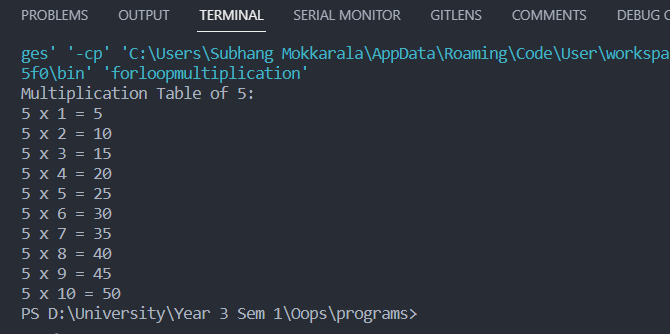
    }

}

* **Data and Results:**

**19-07-2023**

**1.**

****

**2.**

**A screen shot of a computer

Description automatically generated**

* **Analysis and Inferences:**
* **Analysis and Inferences:**

The given Java program uses a for-loop to generate a multiplication table for a user-inputted number. It first prompts the user to enter a number using the Scanner class and stores it in the variable "number." The program then proceeds to print the multiplication table for this number from 1 to 10 using a for-loop. During each iteration of the loop, it calculates the product of the number and the loop index "i," printing the multiplication expression and the result. The program provides a simple and efficient way to generate and display the multiplication table for any given number, helping users understand the basic concept of multiplication and its applications. By entering different numbers, users can quickly generate multiplication tables tailored to their specific needs. Overall, this program is a helpful educational tool and showcases the use of loops and user input in Java programming.

**Sample VIVA-VOCE Questions (In-Lab):**

1. **Write a logic to find the factorial of a given number using for loops.**

To find the factorial of a given number using a for-loop, you can follow this logic:

Initialize a variable factorial to store the result and set it to 1 initially.

Take the input number for which you want to find the factorial.

Run a for-loop from 1 to the input number.

In each iteration, multiply the current value of factorial with the loop index.

Update the value of factorial with the new result in each iteration.

After the loop completes, the value of factorial will be the factorial of the given number.

1. **How many times 'Hello' is printed?**

public class CppBuzz {

public static void main(String[] args){

for(int i = 0; i<5; i++)

{

System.out.println("Hello");

i+=2;

}

}

}

Out put:

Hello

Hello

Hello

1. **Is their any difference in while and do while?**

while loop:

The while loop is a pre-test loop, meaning that the loop condition is checked before each iteration. If the condition is true, the loop body is executed, and if the condition is false initially, the loop body is never executed.

do-while loop:

The do-while loop is a post-test loop, meaning that the loop body is executed at least once before the loop condition is checked. After executing the loop body, the loop condition is evaluated. If the condition is true, the loop continues, and if it is false, the loop terminates.

1. **Difference between break and continue?**

break: Terminates the loop immediately when a condition is met.

continue: Skips the rest of the current iteration and moves to the next iteration when a condition is met.

1. **What is pre-checking Loop**

A pre-checking loop checks the loop condition before executing the loop body. If the condition is false initially, the loop body is not executed. It is often implemented using the while loop.

**Post-Lab:**

. **Java program to illustrate the use of switch statements with strings**

. Java Program to Print Hollow Square Star Pattern

* **Procedure/Program:**

**1.)** public class StringSwitchExample {

    public static void main(String[] args) {

        String dayOfWeek = "Monday";

        switch (dayOfWeek) {

            case "Monday":

                System.out.println("It's the start of the week!");

                break;

            case "Tuesday":

            case "Wednesday":

            case "Thursday":

                System.out.println("It's a workday.");

                break;

            case "Friday":

                System.out.println("TGIF! It's Friday!");

                break;

            case "Saturday":

            case "Sunday":

                System.out.println("It's the weekend. Time to relax!");

                break;

            default:

                System.out.println("Invalid day of the week.");

        }

    }

}

**2.**

public class HollowSquarePattern {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of rows: ");

        int rows = scanner.nextInt();

        for (int i = 1; i <= rows; i++) {

            for (int j = 1; j <= rows; j++) {

                if (i == 1 || i == rows || j == 1 || j == rows) {

                    System.out.print("\* ");

                } else {

                    System.out.print("  ");

                }

            }

            System.out.println();

        }

    }

}

* **Data and Results:**

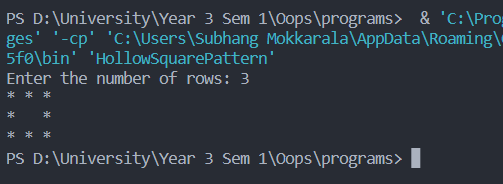
**26-07-2023**

**1.)**

**A screenshot of a computer screen

Description automatically generated**

**2.)**

****

* **Analysis and Inferences:**
* Loop statements in programming enable the repetition of a code block until a certain condition is satisfied. They save coding effort, enhance efficiency, and are crucial for automating repetitive tasks.

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**

**Experiment Title: Programs on Recursion**

**Aim/Objective:** The aim of recursion in programming is to solve complex problems by breaking them down into smaller, simpler instances of the same problem

**Description:**

The recursive case describes how the function calls itself with a smaller instance of the problem. By solving the smaller subproblem, the function makes progress towards solving the original problem. The recursive case reduces the problem size and brings it closer to the base case.

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

**1.What Is Recursion?**

**2. Explain Call by Value?**

**3.Explain Call by Reference?**

**4.Explain Nested If?**

**5.What Are Buzz Words?**

**In-Lab:**

Write a program to Print Fibonacci Series in Java using Recursion  
  
 Write a program to reverse String in Java using Recursion

**Procedure/Program:**

* **Data and Results:**
* **Analysis and Inferences:**

**Sample VIVA-VOCE Questions (In-Lab):**

**1. What Are The Main Features Of Oops?**

**2. What Is Pure Virtual Function?**

**3.What Is Coupling?**

**4.Expalin Scope Of The Variable?**

**5.What You Mean Type Casting?**

**Post-Lab:**

.Write a program to calculate Greatest Common Division  GCD using Euclid's algorithm  
  
.Write a program to calculate the sum of digits using recursion in Java

**Procedure/Program:**

* **Data and Results:**

* **Analysis and Inferences:**

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**

**Experiment Title: Programs On Arrays**

**Aim/Objective:** The aim of arrays in programming is to store and organize multiple values of the same data type under a single variable name

**Description:** Arrays provide a convenient way to store and manipulate collections of elements. They are widely used in programming for tasks like data storage, sorting, searching, and more.

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

1.What is a class?

2. What is an object?

3. What is encapsulation?

4. What is Polymorphism?

5. What is Abstraction?

**In-Lab:**

. Write a program to sort the given array in an ascending order

. Write a Java program to find the index of an array element

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

**Sample VIVA-VOCE Questions (In-Lab):**

1. What will happen if you do not initialize an Array?

2. What is the default value of Array in Java?

3. Can you declare an array without assigning the size of an array?

4. Difference between Array and Object.

**5.**Where is an Array stored in JVM memory?

**Post-Lab:**

. Write a Java program for an array of strings dates[], the task is to sort these dates in ascending order. Note: Each date is of the form dd mmm yyyy where:

Write a Java program to create an array of its anti-diagonals from a given square matrix.

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**

**Experiment Title: Constructors**

**Aim/Objective:** Constructors are special methods that are automatically invoked when an object is created. They allow for the initialization of object properties or variables, setting them to initial values or performing necessary setup tasks.

**Description:** Constructors can accept parameters, allowing for the initialization of object properties based on specific values provided during object creation. Parameterized constructors provide flexibility in customizing object initialization.

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

**1.What is a**[**Constructor**?](https://www.geeksforgeeks.org/constructors-in-java/)

**2.Do we have**[**Copy Constructor in Java**](https://www.geeksforgeeks.org/copy-constructor-in-java/)?

**3.What is No-arg constructor?**

**4.What are**[**private constructors**](https://www.geeksforgeeks.org/private-constructors-and-singleton-classes-in-java/) **and where are they used**

**5.Do we have destructors in Java?**

**In-Lab:**

**.** Java program on constructor over loading

**.** Java program with constructor

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

**Sample VIVA-VOCE Questions (In-Lab):**

1**.Can we have private constructors?**

**2.How a no – argument constructor is different from**[**default Constructor**](https://www.geeksforgeeks.org/g-fact-50/)**?**

**3 What happens if you keep a return type for a constructor?**

**4.Can we call sub class constructor from super class constructor?**

**5.Why Constructors are not inherited in Java?**

**Post-Lab:**

**.** Write a Java program to create a class named 'StudentData' with data members stuID,stuName,stuAge; and get(), set() methods to access and assign values. The class has two constructors which are: a) no parameter b) three parameter, and assign the values by using 'this' keyword. Now create objects of the 'StudentData' class having none and two parameter constructors, and print the result

.java Program to demonstrate the use of the parameterized constructor.

**Procedure/Program:**

* **Data and Results:**
* **Analysis and Inferences:**

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**

**Experiment Title: Method Overloading**

**Aim/Objective:** The aim/objective of method overloading in programming is to provide multiple methods within a class that have the same name but different parameter lists

**Description:**

Method overloading is a feature in object-oriented programming languages that enables the definition of multiple methods with the same name but different parameters. These methods can have different parameter types, different numbers of parameters, or both.

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

**1.What is method overloading?**

**2.What is method signature? What are the things it consist of?**

**3. Can we declare one overloaded method as static and another one as non-static?**

**4 How do compiler differentiate overloaded methods from duplicate methods?**

**5.Is it possible to have two methods in a class with same method signature but different return types?**

**In-Lab:**

**. write java program on Overloading by changing the number of parameters**

**. write java program on Method Overloading by changing the data type of parameters**

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

**Sample VIVA-VOCE Questions (In-Lab):**

**1 Can overloaded method be overrided?**

**2 Overloading is the best example of dynamic binding. True or false?**

**3 Can we overload main() method?**

**4 Can Overloaded Methods Be Synchronized?**

**5 Can we declare overloaded methods as final?**

**Post-Lab:**

. Java program on inheritance and method overriding

.Java program on inheritance and final class – method overeriding

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**

**Experiment Title: String Buffer Classes**

**Aim/Objective:**

String buffer classes offer a convenient way to modify strings without creating new string objects, making them suitable for scenarios that involve frequent string modifications.

**Description:**

String buffer classes are designed to handle mutable strings in programming languages. Unlike regular string objects that are immutable (cannot be changed once created), string buffers allow for efficient modification of strings by providing methods to append, insert, delete, or replace characters or substrings within the string.

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

**1.What is String in Java?**

**2 Is String a class or data type in Java?**

**3 How many ways can we create string object in Java?**

**4What is String Constant Pool in Java?**

**5 What is the superclass of string class in Java?**

**In-Lab:**

**.** Write a Java program to demonstrate String class and its methods. Use length(), indexOf(), toUpperCase(), toLowerCase(), equals(), equalsIgnoreCase(), compareTo(), charAt(), replace(), trim() methods.

. Write a Java program to demonstrate String Buffer capacity() Method

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

**Sample VIVA-VOCE Questions (In-Lab):**

1. When to use String and StringBuffer?

2. What is difference between StringBuffer and StringBuilder?

3. What is wrapper class in java?

4.Which class does not override the equals() and hashCode() methods, inheriting them directly from class Object?

5. Which of the two "StringBuilder" or "StringBuffer" is faster and Why ?,

**Post-Lab:**

Write a Java program to demonstrate StringBuffer ensureCapacity() method

Write a Java program to demonstrate StringBuffer delete() Method

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment**

**Experiment Title: Inheritance**

**Aim/Objective:**

Inheritance enables the creation of new classes based on existing classes, inheriting their attributes and behaviors while allowing customization or extension as needed.

**Description:**

Inheritance is a fundamental concept in object-oriented programming (OOP) that allows classes to inherit properties and behaviors from other classes. It establishes an "is-a" relationship between classes, where a derived class (subclass or child class) inherits characteristics from a base class (superclass or parent class)

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

**1. What is Inheritance in Java?**

**2. Why do we need to use inheritance?**

**3. What is Is-A relationship in Java?**

**4. What is super class and subclass?**

**5. Write the syntax for creating the subclass of a class?**

**In-Lab:**

**.** write java program on Method overriding in Java Inheritance

**.** write java program by using super Keyword in Inheritance

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

**VIVA-VOCE Questions (In-Lab):**

**1. Which class in Java is superclass of every other class?**

Ans:

**2. How will you prove that the features of Superclass are inherited in Subclass?**

Ans:

**3. Can a class extend itself?**

Ans:

**4. Can we assign superclass to subclass?**

Ans:

**5. Can a class extend more than one class?**

Ans:

**Post-Lab:**

**.** Write java program by using protected Members in Inheritance

. Write simple program based on multilevel inheritance in java.

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**

**Experiment Title: Packages**

**Aim/Objective:** The aim/objective of packages in programming is to organize and manage classes, interfaces, and other code entities into meaningful and logical groups

**Description**: Packages are a mechanism in programming languages, particularly in object-oriented languages like Java, to organize code into modular units. A package is a collection of related classes, interfaces, enumerations, and other code entities grouped together

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

**1.Do I need to import java.lang package any time? Why?**

**2.Can I import same package/class twice? Will the JVM load the package twice at runtime?**

**3.What does the default access specifier means ?**

**4.What is the difference between public class and class ?**

**In-Lab:**

**.** Write a Java program to create a package for Book details giving bookName, authorName, price and yearOfPublish, use that package in another program to perform operation on the data.

. Name of the package must be same as the directory under which this file is saved

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

**VIVA-VOCE Questions (In-Lab):**

**1.What is java classpath?**

**2.Can a class in java be private?**

**3. Can we re-assign a value to a field of interfaces?**

**4. Can an interface extend a class?**

**5.Can interfaces have static methods?**

**Post-Lab:**

. Java program with two interfaces

.Java program to Demonstrate List Interface

**Procedure/Program:**

* **Data and Results:**
* **Analysis and Inferences:**

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**

**Experiment Title: Exception Handling**

**Aim/Objective**: The aim/objective of exception handling in programming is to effectively manage and handle unexpected or exceptional situations that may occur during program execution.

**Description**: Exception handling is a mechanism in programming languages that enables the detection, propagation, and handling of errors or exceptional situations during program execution. Exceptions occur when a program encounters unexpected conditions or situations that prevent normal execution

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

**1.How can you handle exceptions in Java?**

**2.Why do we need exception handling in Java?**

**3.Can we just use try instead of finally and catch blocks?**

**4.Name the different types of exceptions in Java**

**5.What is the difference between exception and error in Java?**

**In-Lab:**

**.** Java program on exception handling

**.**Write a Java Program to demonstrate different types of arithmetic exceptions supposed by java. Illustrate using an example, the use of arithmetic exceptions.

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

**VIVA-VOCE Questions (In-Lab):**

**1. Can we throw an exception explicitly or manually?**

**2. Describe OutofMemoryError in exception handling.?**

**3 Is it illegal to keep an empty catch?**

**4.Suppose there is a catch block in tune with a try block with 3 statements - 1, 2, and 3. Now, imagine that the statement is thrown in statement 2. Will there be an execution of statement 3?**

**5. Define unreachable catch block error.**

**Post-Lab:**

. Write a Java program to demonstrate the concept of exception handling mechanism to handle user defined exceptions using throw keyword.

. Write a Java Program To Print The Exception Information Using Getmessage() Method

**Procedure/Program:**

* **Data and Results:**
* **Analysis and Inferences:**

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**

**Experiment Title: Multi Threading**

**Aim/Objective:** The aim/objective of multithreading in programming is to achieve concurrent execution and improve program performance by executing multiple threads simultaneously.

**Description:** Multithreading is a technique in programming that enables the execution of multiple threads within a single program. A thread is a lightweight unit of execution that represents an independent sequence of instructions.

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

**1. What is multithreading?**

**2.What is the purpose of wait() method in Java?**

**3. What is context switching?**

**4. should we interrupt a thread?**

**5.Can Java object be locked down for exclusive use by a given thread?**

**In-Lab:**

. Java program on threads using Thread class

. Java program Multithreading In Java Using The Runnable Interface**.**

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

**VIVA-VOCE Questions (In-Lab):**

**1. What is the purpose of wait() method in Java?**

**2.What is context switching?**

**3.What does join() method?**

**4.When should we interrupt a thread?**

**5.What is the difference between notify() and notifyAll()?**

**Post-Lab:**

. write java program **to demonstrate Join() method of Thread class in java**

. write java Program For Producer Consumer Problem

.**Procedure/Program:**

* **Data and Results:**
* **Analysis and Inferences:**

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**

**Title: Swing package-based event driven programming.**

**Aim/Objective:** The aim/objective of using the Swing package for event-driven programming is to create graphical user interfaces (GUIs) and develop interactive applications

**Description:**

Swing is a GUI toolkit in Java that allows developers to create cross-platform desktop applications with a rich graphical user interface. Swing provides a collection of classes and components that can be used to create windows, dialog boxes, buttons, menus, text fields, and other GUI elements

**Pre-Requisites:**

**1. Java Development Kit (JDK)**

**2. Text Editor**

**3. Integrated Development Environment (IDE)**

**Pre-Lab:**

**1.What are the different ways of threads usage?**

**2.What are the differences between the constructor and method of a class in Java?**

**3.What do you understand about Object Cloning and how do you achieve it in Java?**

**4.Can you explain the Java thread lifecycle?**

**5.What is a framework in Java?**

**In-Lab:**

**. Demonstrate The Event Handling Within The Class**

**. Event Handling By The Other Class**

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

**VIVA-VOCE Questions (In-Lab):**

**1.Does Swing is a thread-safe? What do you mean by swing is not thread-safe?**

**2 Why Swing components are called lightweight components?**

**3.What is difference between Swing and AWT in Java?**

**4.Which method of Swing are thread-safe?**

**5.What is difference between Container and Component ?**

**Post-Lab:**

. write java for Event Handling By The Anonymous Class

. write java for Demonstrate Event Handling In Java

* **Procedure/Program:**
* **Data and Results:**
* **Analysis and Inferences:**

|  |  |
| --- | --- |
| **Evaluator Remark (if Any):** | **Marks Secured: \_\_\_\_\_out of 50** |
| **Signature of the Evaluator with Date** |

**Evaluator MUST ask Viva-voce prior to signing and posting marks for each experiment.**