OBJECT ORIENTED PROGRAMMING WITH JAVA LAB

B.Com., COMPUTER APPLICATION/BA. COMPUTER APPLICATION

Semester – IV

Lesson Writers

Dr. G. Neelima
Asst. professor
Dept. of Computer Science
Acharya Nagarjuna University

Dr. U. Surya Kameswari
Asst. professor
Dept. of Computer Science
Acharya Nagarjuna University

Editor

Dr. KAMPA LAVANYA
Asst. Professor
Dept. of Computer Science
Acharya Nagarjuna University

DirectorI/c
Prof. V.VENKATESWARLU
MA., M.P.S., M.S.W., M.Phil., Ph.D

CENTRE FOR DISTANCE EDUCATION
ACHARAYANAGARJUNAUNIVERSITY
NAGARJUNANAGAR – 522510
Ph:0863-2346222,2346208,
0863-2346259(Study Material)
Website: www.anucde.info
e-mail:anucdedirector@gmail.com

B.Com., COMPUTER APPLICATION/BA. COMPUTER APPLICATION

Semester – IV

410BCO21: OBJECT ORIENTED PROGRAMMING WITH JAVA LAB

SYLLABUS

1. Write a program to implement command line arguments.
2. Write a program to read Student Name, Reg.No, Marks and calculate Total, Percentage, and Result. Display all the details of students .
3. Write a program to perform String Operations.
4. Java program to implement Addition of two N X N matrices.
5. Java program to implement bubble sort.
6. Java program to demonstrate the use of Constructor.
7. Calculate area of the following shapes using method overloading. a.Rectangle b. Circle c. Square
8. Implement multilevel inheritance
9. Java program for to display Serial Number from 1 to 5 by creating two Threads
10. Java program to demonstrate the following exception handlings a. Divided by Zero b. Array Index Out of Bound c. Arithmetic Exception

CONTENTS

OBJECT ORIENTED PROGRAMMING WITH JAVA LAB

OBJECTIVES:

The objective of this lab is to master the OOP concepts with java programming and to learn how to work with real time applications using java programming. These programs are widely used in most real-time scenario. After the end of lab, students will be able know the complete practical exposure on oop using java

STRUCTURE

1. Program to implement command line arguments
2. Program to read Student Name, Reg.No , Marks and calculate total percentage , and Results. Display all the details of students.
3. Program to perform string operations
4. Program to implement addition of two N*N matrices
5. Program to implement bubble sort
6. program to demonstrate the use of Constructor
7. Calculate area of following shapes using method overloading

a. Rectangle b. Circle c.Square

8. Program to implement multilevel inheritance
9. Program to display serial number from 1 to 5 by creating two threads
10. Program to demonstrate the following exceptions

a. Divide by zero b. Array index out of bound c. Arithmetic Exception

1. PROGRAM TO IMPLEMENT COMMAND LINE ARGUMENTS

CODE:

```
public class CommandLineExample {
    public static void main(String[] args) {
        // Check if any command-line arguments are provided
        if (args.length == 0) {
System.out.println("No command-line arguments were provided.");
            return;
        }

        // Print each command-line argument along with its length
System.out.println("Command-line arguments provided:");
        for (int i = 0; i<args.length; i++) {
System.out.println("Argument " + (i + 1) + ": " + args[i] + " (Length: " + args[i].length() +
")");
        }
    }
}
```

OUTPUT:

```
Command-line arguments provided:
Argument 1: Hello (Length: 5)
Argument 2: World (Length: 5)
Argument 3: Java (Length: 4)
Argument 4: Programming (Length: 11)
```

EXPLANATION:

- The program prints each word (argument) passed to it from the command line, along with the position of the argument and its length.
- For example, "Hello" is the first argument, and its length is 5 characters.

-Dr. G. NEELIMA

2. PROGRAM TO READ STUDENT NAME, REG.NO , MARKS AND CALCULATE TOTAL PERCENTAGE , AND RESULTS. DISPLAY ALL THE DETAILS OF STUDENTS.

CODE:

```
import java.util.Scanner;

public class StudentDetails {
   public static void main(String[] args) {
       Scanner scanner = new Scanner(System.in);

       // Reading student details
System.out.print("Enter Student Name: ");
       String name = scanner.nextLine();

System.out.print("Enter Registration Number: ");
       String regNo = scanner.nextLine();

System.out.print("Enter marks for Subject 1: ");
       double marks1 = scanner.nextDouble();

System.out.print("Enter marks for Subject 2: ");
       double marks2 = scanner.nextDouble();

System.out.print("Enter marks for Subject 3: ");
       double marks3 = scanner.nextDouble();

       // Calculating total marks, percentage, and determining the result
       double totalMarks = marks1 + marks2 + marks3;
       double percentage = (totalMarks / 300) * 100;
       String result = (marks1 >= 40 && marks2 >= 40 && marks3 >= 40) ? "Pass" : "Fail";

       // Displaying the student details
System.out.println("\n--- Student Details ---");
System.out.println("Name: " + name);
System.out.println("Registration Number: " + regNo);
System.out.println("Marks:");
System.out.println("  Subject 1: " + marks1);
System.out.println("  Subject 2: " + marks2);
System.out.println("  Subject 3: " + marks3);
System.out.println("Total Marks: " + totalMarks + " / 300");
System.out.println("Percentage: " + String.format("%.2f", percentage) + " %");
System.out.println("Result: " + result);

scanner.close();
   }
}
```

OUTPUT:

```
Enter Student Name: John Doe
Enter Registration Number: 12345
Enter marks for Subject 1: 75
Enter marks for Subject 2: 85
Enter marks for Subject 3: 65

--- Student Details ---
Name: John Doe
Registration Number: 12345
Marks:
  Subject 1: 75.0
  Subject 2: 85.0
  Subject 3: 65.0
Total Marks: 225.0 / 300
Percentage: 75.00 %
Result: Pass
```

EXPLANATION:

- Name: The name of the student entered by the user.
- Registration Number: The registration number entered by the user.
- Marks: The marks obtained by the student in each subject.
- Total Marks: The sum of marks in all three subjects.
- Percentage: The percentage of marks calculated from the total marks.
- Result: Pass or Fail based on whether the student passed all subjects.

-Dr. G. NEELIMA

3. PROGRAM TO PERFORM STRING OPERATIONS

CODE:

```
import java.util.Scanner;
public class StringOperations {
   public static void main(String[] args) {
       Scanner scanner = new Scanner(System.in);

       // Reading strings from the user
System.out.print("Enter the first string: ");
       String str1 = scanner.nextLine();

System.out.print("Enter the second string: ");
       String str2 = scanner.nextLine();

       // 1. Concatenation
       String concatenatedString = str1 + " " + str2;
System.out.println("\nConcatenated String: " + concatenatedString);

       // 2. Length of the strings
System.out.println("Length of the first string: " + str1.length());
System.out.println("Length of the second string: " + str2.length());

       // 3. Convert strings to uppercase and lowercase
System.out.println("First string in uppercase: " + str1.toUpperCase());
System.out.println("Second string in lowercase: " + str2.toLowerCase());

       // 4. Extracting substring
       if (str1.length() >= 3) {
System.out.println("Substring of the first string (first 3 characters): " + str1.substring(0, 3));
       } else {
System.out.println("First string is too short to extract a 3-character substring.");
       }

       if (str2.length() >= 3) {
System.out.println("Substring of the second string (first 3 characters): " + str2.substring(0, 3));
       } else {
System.out.println("Second string is too short to extract a 3-character substring.");
       }

       // 5. Comparing the two strings
       if (str1.equals(str2)) {
System.out.println("Both strings are equal.");
       } else {
System.out.println("The strings are not equal.");
       }

       // 6. Comparing strings ignoring case
       if (str1.equalsIgnoreCase(str2)) {
System.out.println("Both strings are equal (case ignored).");
```

```
        } else {
System.out.println("The strings are not equal (case ignored).");
        }

scanner.close();
    }
}
```

OUTPUT:

```
Enter the first string: Hello
Enter the second string: World
Concatenated String: Hello World
Length of the first string: 5
Length of the second string: 5
First string in uppercase: HELLO
Second string in lowercase: world
Substring of the first string (first 3 characters): Hel
Substring of the second string (first 3 characters): Wor
The strings are not equal.
The strings are not equal (case ignored).
```

EXPLANATION:

- Concatenated String: Combines the two input strings with a space in between.
- Length: Displays the number of characters in each string.
- Uppercase and Lowercase: Converts and displays the first string in all uppercase letters and the second string in all lowercase letters.
- Substring: Extracts and displays the first three characters of each string, if possible.
- String Comparison: Checks whether the two strings are equal, both with and without considering case sensitivity.

-Dr. G. NEELIMA

4. PROGRAM TO IMPLEMENT ADDITION OF TWO N*N MATRICES

CODE:

```
import java.util.Scanner;

public class MatrixAddition {
   public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Reading the size of the matrix
System.out.print("Enter the size of the matrix (N x N): ");
        int n = scanner.nextInt();

        // Initializing the matrices
int[][] matrix1 = new int[n][n];
int[][] matrix2 = new int[n][n];
int[][] sumMatrix = new int[n][n];

        // Reading the first matrix
System.out.println("Enter the elements of the first matrix:");
        for (int i = 0; i< n; i++) {
            for (int j = 0; j < n; j++) {
System.out.print("Element [" + i + "][" + j + "]: ");
                matrix1[i][j] = scanner.nextInt();
            }
        }

        // Reading the second matrix
System.out.println("Enter the elements of the second matrix:");
        for (int i = 0; i< n; i++) {
            for (int j = 0; j < n; j++) {
System.out.print("Element [" + i + "][" + j + "]: ");
                matrix2[i][j] = scanner.nextInt();
            }
        }

        // Adding the two matrices
        for (int i = 0; i< n; i++) {
            for (int j = 0; j < n; j++) {
sumMatrix[i][j] = matrix1[i][j] + matrix2[i][j];
            }
        }

        // Displaying the resulting matrix
System.out.println("The resulting matrix after addition is:");
        for (int i = 0; i< n; i++) {
            for (int j = 0; j < n; j++) {
System.out.print(sumMatrix[i][j] + " ");
            }
System.out.println();
```

```
        }

scanner.close();
    }
}
```

OUTPUT:

```
Enter the size of the matrix (N x N): 2
Enter the elements of the first matrix:
Element [0][0]: 1
Element [0][1]: 2
Element [1][0]: 3
Element [1][1]: 4
Enter the elements of the second matrix:
Element [0][0]: 5
Element [0][1]: 6
Element [1][0]: 7
Element [1][1]: 8
The resulting matrix after addition is:
6 8
10 12
```

EXPLANATION:

This program handles matrix addition by adding corresponding elements and is flexible for any N×NN \times NN×N matrix size as defined by the user.

-Dr. G. NEELIMA

5. PROGRAM TO IMPLEMENT BUBBLE SORT

CODE:

```
import java.util.Scanner;

public class BubbleSort {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Reading the size of the array
System.out.print("Enter the number of elements: ");
        int n = scanner.nextInt();

        // Initializing the array
int[] array = new int[n];

        // Reading the elements of the array
System.out.println("Enter the elements of the array:");
        for (int i = 0; i< n; i++) {
System.out.print("Element " + (i + 1) + ": ");
            array[i] = scanner.nextInt();
        }

        // Performing bubble sort
bubbleSort(array);

        // Displaying the sorted array
System.out.println("Sorted array:");
        for (int i = 0; i< n; i++) {
System.out.print(array[i] + " ");
        }

scanner.close();
    }

    // Bubble sort function
    public static void bubbleSort(int[] array) {
        int n = array.length;
        for (int i = 0; i< n - 1; i++) {
            for (int j = 0; j < n - i - 1; j++) {
                // Swap if the current element is greater than the next element
                if (array[j] >array[j + 1]) {
                    int temp = array[j];
                    array[j] = array[j + 1];
array[j + 1] = temp;
                }
            }
        }
    }
}
```

OUTPUT:

```
Enter the number of elements: 5
Enter the elements of the array:
Element 1: 64
Element 2: 34
Element 3: 25
Element 4: 12
Element 5: 22
Sorted array:
12 22 25 34 64
```

EXPLANATION:

- The Bubble Sort algorithm sorts the array in ascending order.
- After the sorting process, the smallest element moves to the front, and the largest element moves to the end of the array.

-Dr. G. NEELIMA

6. PROGRAM TO DEMONSTRATE THE USE OF CONSTRUCTOR

CODE:

```
class Student {
    // Instance variables
    String name;
    int regNo;
    double marks1;
    double marks2;
    double marks3;

    // Default constructor
    public Student() {
        // Initializing with default values
        name = "Unknown";
regNo = 0;
        marks1 = 0.0;
        marks2 = 0.0;
        marks3 = 0.0;
    }

    // Parameterized constructor
    public Student(String name, int regNo, double marks1, double marks2, double marks3) {
        // Initializing with provided values
        this.name = name;
this.regNo = regNo;
this.marks1 = marks1;
this.marks2 = marks2;
this.marks3 = marks3;
    }

    // Method to calculate total marks
    public double calculateTotal() {
        return marks1 + marks2 + marks3;
    }

    // Method to display student details
    public void displayDetails() {
System.out.println("Student Name: " + name);
System.out.println("Registration Number: " + regNo);
System.out.println("Marks 1: " + marks1);
System.out.println("Marks 2: " + marks2);
System.out.println("Marks 3: " + marks3);
System.out.println("Total Marks: " + calculateTotal());
    }
}

public class ConstructorDemo {
    public static void main(String[] args) {
        // Creating an object using the default constructor
        Student student1 = new Student();
System.out.println("Student 1 Details (Using Default Constructor):");
        student1.displayDetails();

System.out.println();

        // Creating an object using the parameterized constructor
        Student student2 = new Student("John Doe", 12345, 85.0, 90.5, 78.0);
System.out.println("Student 2 Details (Using Parameterized Constructor):");
        student2.displayDetails();
    }
}
```

OUTPUT:

```
Student 1 Details (Using Default Constructor):
Student Name: Unknown
Registration Number: 0
Marks 1: 0.0
Marks 2: 0.0
Marks 3: 0.0
Total Marks: 0.0

Student 2 Details (Using Parameterized Constructor):
Student Name: John Doe
Registration Number: 12345
Marks 1: 85.0
Marks 2: 90.5
Marks 3: 78.0
Total Marks: 253.5
```

EXPLANATION:

Student 1:

Created using the default constructor, so the details are initialized to default values: "Unknown" for the name, 0 for the registration number, and 0.0 for marks.

Student 2:

Created using the parameterized constructor with specific values provided during object creation.

-Dr. U.SURYA KAMESWARI

7. CALCULATE AREA OF FOLLOWING SHAPES USING METHOD OVERLOADING

a. Rectangle b. Circle c.Square

CODE:

```
public class AreaCalculator {

   // Method to calculate the area of a circle
   public double calculateArea(double radius) {
      return Math.PI * radius * radius;
   }

   // Method to calculate the area of a rectangle
   public double calculateArea(double length, double breadth) {
      return length * breadth;
   }

   // Method to calculate the area of a triangle
   public double calculateArea(double base, double height, booleanisTriangle) {
      if (isTriangle) {
         return 0.5 * base * height;
      }
      return 0; // Default return if not a triangle (though, this condition is just illustrative)
   }

   public static void main(String[] args) {
AreaCalculator calculator = new AreaCalculator();

      // Calculate and display the area of a circle
      double circleRadius = 5.0;
      double circleArea = calculator.calculateArea(circleRadius);
System.out.println("Area of the circle with radius " + circleRadius + " is: " + circleArea);

      // Calculate and display the area of a rectangle
      double rectangleLength = 10.0;
      double rectangleBreadth = 5.0;
      double rectangleArea = calculator.calculateArea(rectangleLength, rectangleBreadth);
System.out.println("Area of the rectangle with length " + rectangleLength + " and breadth " +
rectangleBreadth + " is: " + rectangleArea);

      // Calculate and display the area of a triangle
      double triangleBase = 8.0;
      double triangleHeight = 6.0;
      double triangleArea = calculator.calculateArea(triangleBase, triangleHeight, true);
System.out.println("Area of the triangle with base " + triangleBase + " and height " +
triangleHeight + " is: " + triangleArea);
   }
}
```

OUTPUT:

```
Area of the circle with radius 5.0 is: 78.53981633974483
Area of the rectangle with length 10.0 and breadth 5.0 is: 50.0
Area of the triangle with base 8.0 and height 6.0 is: 24.0
```

EXPLANATION:

- The program calculates and displays the area for each shape using the appropriate overloaded method.
- Circle: With a radius of 5.0, the area is approximately 78.54.
- Rectangle: With a length of 10.0 and a breadth of 5.0, the area is 50.0.
- Triangle: With a base of 8.0 and a height of 6.0, the area is 24.0.

-Dr. U.SURYA KAMESWARI

8. PROGRAM TO IMPLEMENT MULTILEVEL INHERITANCE

CODE:

```
// Base class
class Animal {
   String name;

   public void eat() {
System.out.println(name + " is eating.");
   }

   public void sleep() {
System.out.println(name + " is sleeping.");
   }
}

// Derived class from Animal
class Mammal extends Animal {
   public void walk() {
System.out.println(name + " is walking.");
   }
}

// Derived class from Mammal
class Dog extends Mammal {
   public void bark() {
System.out.println(name + " is barking.");
   }
}

public class MultilevelInheritanceDemo {
   public static void main(String[] args) {
      // Creating an instance of Dog class
      Dog dog = new Dog();
      // Setting the name of the dog
      dog.name = "Buddy";

      // Calling methods from all levels of the inheritance hierarchy
dog.eat();   // Inherited from Animal class
dog.sleep(); // Inherited from Animal class
dog.walk();  // Inherited from Mammal class
dog.bark();  // Defined in Dog class
   }
}
```

OUTPUT:

```
Buddy is eating.
Buddy is sleeping.
Buddy is walking.
Buddy is barking.
```

EXPLANATION:

- The Dog class inherits the eat() and sleep() methods from the Animal class, and the walk() method from the Mammal class.
- When the dog object calls these methods, it performs actions appropriate to each method, using the name "Buddy".

-Dr. U.SURYA KAMESWARI

9. PROGRAM TO DISPLAY SERIAL NUMBER FROM 1 TO 5 BY CREATING TWO THREADS

CODE:

```
class SerialNumberPrinter implements Runnable {
   private String threadName;

   // Constructor to assign a name to the thread
   public SerialNumberPrinter(String name) {
this.threadName = name;
   }

   @Override
   public void run() {
      // Printing numbers from 1 to 5
      for (int i = 1; i<= 5; i++) {
System.out.println(threadName + ": " + i);
         try {
            // Introducing a small delay to simulate real-time thread execution
Thread.sleep(100); // 100 milliseconds delay
         } catch (InterruptedException e) {
System.out.println(threadName + " interrupted.");
         }
      }
   }
}

public class SerialNumberThreads {
   public static void main(String[] args) {
      // Creating two instances of the Runnable
SerialNumberPrinter printer1 = new SerialNumberPrinter("Thread 1");
SerialNumberPrinter printer2 = new SerialNumberPrinter("Thread 2");

      // Creating Thread objects
      Thread thread1 = new Thread(printer1);
      Thread thread2 = new Thread(printer2);

      // Starting the threads
      thread1.start();
      thread2.start();

      // Waiting for both threads to finish
      try {
         thread1.join();
         thread2.join();
      } catch (InterruptedException e) {
System.out.println("Main thread interrupted.");
      }

System.out.println("Both threads have finished execution.");
```

```
    }
}
```

OUTPUT:

```
Thread 1: 1
Thread 2: 1
Thread 1: 2
Thread 2: 2
Thread 1: 3
Thread 2: 3
Thread 1: 4
Thread 2: 4
Thread 1: 5
Thread 2: 5
Both threads have finished execution.
```

EXPLANATION:

- The two threads execute concurrently, each printing numbers from 1 to 5.
- Due to the small delay introduced by Thread.sleep(100), the threads alternate their output, but the exact sequence may vary depending on the thread scheduler.

-Dr. U.SURYA KAMESWARI

10. PROGRAM TO DEMONSTRATE THE FOLLOWING EXCEPTIONS a.Divide by zero b. Array index out of bound c. Arithmetic Exception

CODE:

```
public class ExceptionHandlingDemo {
   public static void main(String[] args) {
      // 1. Demonstrating Divide by Zero (ArithmeticException)
      try {
         int a = 10;
         int b = 0;
         int result = a / b;
System.out.println("Result of division: " + result);
      } catch (ArithmeticException e) {
System.out.println("Exception caught: " + e.getMessage());
      }

      // 2. Demonstrating Array Index Out of Bounds
      try {
int[] array = {1, 2, 3, 4, 5};
System.out.println("Accessing element at index 10: " + array[10]);
      } catch (ArrayIndexOutOfBoundsException e) {
System.out.println("Exception caught: " + e.getMessage());
      }

      // 3. Demonstrating Arithmetic Exception (general)
      try {
         int number = Integer.parseInt("NaN"); // This will throw NumberFormatException, a
subclass of ArithmeticException
System.out.println("Parsed number: " + number);
      } catch (NumberFormatException e) {
System.out.println("Exception caught: " + e.getMessage());
      }
   }
}
```

OUTPUT:

```
Exception caught: / by zero
Exception caught: Index 10 out of bounds for length 5
Exception caught: For input string: "NaN"
```

EXPLANATION:

- / by zero: This message indicates that the program tried to divide by zero, which is not allowed.
- Index 10 out of bounds for length 5: This message indicates that the program tried to access an array element at an index that doesn't exist.
- For input string: "NaN": This message indicates that the program tried to parse a non-numeric string as an integer, which is invalid.

-Dr. U.SURYA KAMESWARI