LAB # 1

INTRODUCTION TO OOP

OBJECTIVE:

To understand OOP(Java Environment),Data Types and Mixed Arithmetic Expression. To get familiarity with the Scanner Class for reading and taking inputs in java.

LAB TASK

A. Write a Java program that reads a number in inches, converts it to meters. Note: One inch is 0.0254 meter.

```java
package com.mycompany.lab1;

import java.util.Scanner;

public class Lab1{
    public static void main(String[] args) {
        // Create a Scanner object for input
        Scanner input = new Scanner(System.in);

        // Prompt user for input
        System.out.print("Enter a value in inches: ");
        double inches = input.nextDouble();

        // Convert inches to meters
        double meters = inches * 0.0254;

        // Display the result
        System.out.println(inches + " inches is equal to " + meters + " meters.");


    }
}
```

```
--- exec:3.1.0:exec (default-cli) @ Lab1 ---
Enter a value in inches: 45
45.0 inches is equal to 1.143 meters.
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  15.554 s
Finished at: 2025-03-18T15:45:18+05:00
------------------------------------------------------------------------
```

B. Write a Java program to convert days into number of months and days.

```java
package com.mycompany.lab1;

import java.util.Scanner;

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

        // Taking input from the user
        System.out.print("Enter the number of days: ");
        int totalDays = scanner.nextInt();

        // Assuming 1 month = 30 days
        int months = totalDays / 30;
        int remainingDays = totalDays % 30;

        // Displaying the result
        System.out.println(totalDays + " days is approximately " + months + " months and " + remainingDays + " days.");


    }
}

```

```
--- exec:3.1.0:exec (default-cli) @ Lab1 ---
Enter the number of days: 764
764 days is approximately 25 months and 14 days.
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  9.105 s
Finished at: 2025-03-18T16:01:39+05:00
------------------------------------------------------------------------
```

C. Write a Java program to compute body mass index (BMI).

Note:

weight in kilogram = weight * 0.45359237.

Height in feet= inches * 0.0254.

BMI= weight in kilogram/(hight in fee)^2.

```java
package com.mycompany.lab1;
import java.util.Scanner;
public class BMICalculator {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Taking user input for weight and height
        System.out.print("Enter your weight in pounds: ");
        double weightInPounds = scanner.nextDouble();

        System.out.print("Enter your height in inches: ");
        double heightInInches = scanner.nextDouble();

        // Converting weight to kilograms
        double weightInKg = weightInPounds * 0.45359237;

        // Converting height to feet
        double heightInFeet = heightInInches * 0.0254;

        // Calculating BMI
        double bmi = weightInKg / (heightInFeet * heightInFeet);

        // Displaying the result
        System.out.printf("Your BMI is: %.2f\n", bmi);
    }
}
```

```
--- exec:3.1.0:exec (default-cli) @ Lab1 ---
Enter your weight in pounds: 30
Enter your height in inches: 5.9
Your BMI is: 605.92
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  29.288 s
Finished at: 2025-03-18T16:32:57+05:00
------------------------------------------------------------------------
```

D. Write a program to make BIO_DATA and take several user inputs (i.e. name, age, gender, department, email id, father name) and print the result along with the greeting message.

```
package com.mycompany.lab1;
import java.util.Scanner;

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

        // Taking user input for Bio Data
        System.out.print("Enter your Name: ");
        String name = scanner.nextLine();

        System.out.print("Enter your Age: ");
        int age = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        System.out.print("Enter your Gender: ");
        String gender = scanner.nextLine();

        System.out.print("Enter your Department: ");
        String department = scanner.nextLine();

        System.out.print("Enter your Email ID: ");
        String email = scanner.nextLine();
```

```
        System.out.print("Enter your Father's Name: ");
        String fatherName = scanner.nextLine();

        // Displaying the Bio Data
        System.out.println("\n***** BIO DATA *****");
        System.out.println("Hello, " + name + "! Welcome!");
        System.out.println("Age: " + age);
        System.out.println("Gender: " + gender);
        System.out.println("Department: " + department);
        System.out.println("Email ID: " + email);
        System.out.println("Father's Name: " + fatherName);

    }
}
```

```
--- exec:3.1.0:exec (default-cli) @ Lab1 ---
Enter your Name: Muhammad Asad Khan
Enter your Age: 19
Enter your Gender: Male
Enter your Department: Software Engineering
Enter your Email ID: asad123@gmail.com
Enter your Father's Name: Muhammad Bashir

***** BIO DATA *****
Hello, Muhammad Asad Khan! Welcome!
Age: 19
Gender: Male
Department: Software Engineering
Email ID: asad123@gmail.com
Father's Name: Muhammad Bashir
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  01:21 min
Finished at: 2025-03-18T16:47:09+05:00
------------------------------------------------------------------------
```

E. Write a program which takes two numbers as input from user and calculate sum and difference of both numbers.

```java
package com.mycompany.lab1;
import java.util.Scanner;

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

        // Taking user input for two numbers
        System.out.print("Enter the first number: ");
        double num1 = scanner.nextDouble();

        System.out.print("Enter the second number: ");
        double num2 = scanner.nextDouble();

        // Calculating sum and difference
        double sum = num1 + num2;
        double difference = num1 - num2;

        // Displaying the results
        System.out.println("\nThe Sum of " + num1 + " and " + num2 + " is: " + sum);
        System.out.println("The Difference of " + num1 + " and " + num2 + " is: " + difference);

    }
}
```

```
--- exec:3.1.0:exec (default-cli) @ Lab1 ---
Enter the first number: 45
Enter the second number: 30

The Sum of 45.0 and 30.0 is: 75.0
The Difference of 45.0 and 30.0 is: 15.0
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  02:53 min
Finished at: 2025-03-19T13:59:04+05:00
------------------------------------------------------------------------
```

F. Take a three-digit number from user, separate each digit of the number and sum them.

```java
package com.mycompany.lab1;

import java.util.Scanner;

public class SumDifferenceCalculator2{
    public static void main(String[] args) {
        // Create a Scanner object to take input from the user
        Scanner scanner = new Scanner(System.in);
        // Prompt the user to enter a three-digit number
        System.out.print("Enter a three-digit number: ");
        int number = scanner.nextInt();
        // Extract digits
        int digit1 = number / 100;
        int digit2 = (number / 10) % 10;
        int digit3 = number % 10;
        // Calculate sum of digits
        int sumOfDigits = digit1 + digit2 + digit3;
        // Display the results
        System.out.println("Digits: " + digit1 + ", " + digit2 + ", " + digit3);
        System.out.println("Sum of digits: " + sumOfDigits);

    }
}
```

```
--- exec:3.1.0:exec (default-cli) @ Lab1 ---
Enter a three-digit number: 268
Digits: 2, 6, 8
Sum of digits: 16
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  25.159 s
Finished at: 2025-03-19T14:02:01+05:00
------------------------------------------------------------------------
```

LAB # 2

Implementation of various operators in java & Construct of Escape sequence.

OBJECTIVE:

To get familiarity with the various operators in java and the escape sequence.

LAB TASK

A. Create a Java application for a basic calculator with the following capabilities:

Addition and Subtraction

Division, Exponentiation, and Multiplication

The user should be able to select the desired operation and enter two numbers into the program.

```java
package javaapplication3;

import java.util.Scanner;

public class BasicCalculator {

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

        System.out.println("=== Basic Calculator ===");
        System.out.println("Choose an operation:");
        System.out.println("+ for Addition");
        System.out.println("- for Subtraction");
        System.out.println("* for Multiplication");
        System.out.println("/ for Division");
        System.out.println("^ for Exponentiation");

        System.out.print("Enter the operation: ");
        String operation = scanner.next();

        System.out.print("Enter first number: ");
        double num1 = scanner.nextDouble();

        System.out.print("Enter second number: ");
        double num2 = scanner.nextDouble();

        double result;
```

```
28
29          if (operation.equals("+")) {
30              result = num1 + num2;
31              System.out.println("Result: " + result);
32          } else if (operation.equals("-")) {
33              result = num1 - num2;
34              System.out.println("Result: " + result);
35          } else if (operation.equals("*")) {
36              result = num1 * num2;
37              System.out.println("Result: " + result);
38          } else if (operation.equals("/")) {
39              if (num2 != 0) {
40                  result = num1 / num2;
41                  System.out.println("Result: " + result);
42              } else {
43                  System.out.println("Error: Division by zero is not allowed!");
44              }
45          } else if (operation.equals("^")) {
46              result = Math.pow(num1, num2);
47              System.out.println("Result: " + result);
48          } else {
49              System.out.println("Invalid operation. Please try again.");
50          }
51
52
53      }
54  }
```

```
run:
=== Basic Calculator ===
Choose an operation:
+ for Addition
- for Subtraction
* for Multiplication
/ for Division
^ for Exponentiation
Enter the operation: -
Enter first number: 12
Enter second number: 6
Result: 6.0
BUILD SUCCESSFUL (total time: 19 seconds)
```

B. To convert temperature from Celsis to Fahrenheit, write a Java program <u>Guidelines:</u>

Asking for the temperature in Celsius will prompt the user.

Check the temperature that the user has entered.

Use this formula to convert the temperature from Celsius to Fahrenheit: Degrees Celsius * (9/5) + 32 = Fahrenheit

Print the Fahrenheit temperature conversion.

```java
package lab.pkg2;
import java.util.Scanner;

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

        System.out.print("Enter temperature in Celsius: ");
        if (scanner.hasNextDouble()) {
            double celsius = scanner.nextDouble();
            double fahrenheit = (celsius * 9 / 5) + 32;
            System.out.println("Temperature in Fahrenheit: " + fahrenheit);
        } else {
            System.out.println("Invalid input. Please enter a valid numeric temperature.");
        }


    }
}

```

```
run:
Enter temperature in Celsius: 45
Temperature in Fahrenheit: 113.0
BUILD SUCCESSFUL (total time: 16 seconds)
```

C. Write a program which shows the implementation of increment and decrement operators.

```java
package lab.pkg2;

public class IncrementDecrementDemo {
    public static void main(String[] args) {
        int number = 10;

        System.out.println("Initial value: " + number);

        // Post-increment
        System.out.println("Post-increment (number++): " + (number++));
        System.out.println("Value after post-increment: " + number);

        // Pre-increment
        System.out.println("Pre-increment (++number): " + (++number));

        // Post-decrement
        System.out.println("Post-decrement (number--): " + (number--));
        System.out.println("Value after post-decrement: " + number);

        // Pre-decrement
        System.out.println("Pre-decrement (--number): " + (--number));
    }
}

```

```
run:
Initial value: 10
Post-increment (number++): 10
Value after post-increment: 11
Pre-increment (++number): 12
Post-decrement (number--): 12
Value after post-decrement: 11
Pre-decrement (--number): 10
BUILD SUCCESSFUL (total time: 0 seconds)
```

D. Write a program which shows the implementation of any three bitwise operators.

```java
public class BitwiseOperatorsDemo {
    public static void main(String[] args) {
        int a = 5;  // Binary: 0101
        int b = 3;  // Binary: 0011

        // Bitwise AND
        int andResult = a & b; // 0101 & 0011 = 0001 (1 in decimal)
        System.out.println("Bitwise AND (a & b): " + andResult);

        // Bitwise OR
        int orResult = a | b; // 0101 | 0011 = 0111 (7 in decimal)
        System.out.println("Bitwise OR (a | b): " + orResult);

        // Bitwise XOR
        int xorResult = a ^ b; // 0101 ^ 0011 = 0110 (6 in decimal)
        System.out.println("Bitwise XOR (a ^ b): " + xorResult);
    }
}
```

```
run:
Bitwise AND (a & b): 1
Bitwise OR (a | b): 7
Bitwise XOR (a ^ b): 6
BUILD SUCCESSFUL (total time: 0 seconds)
```

E. Write a program to print the Fibonacci Series without using recursion.

```java
import java.util.Scanner;

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

        System.out.print("Enter the number of terms for Fibonacci series: ");
        int n = scanner.nextInt();

        int first = 0, second = 1;
        System.out.print("Fibonacci Series: " + first + " " + second);

        for (int i = 2; i < n; i++) {
            int next = first + second;
            System.out.print(" " + next);
            first = second;
            second = next;
        }

        System.out.println();

    }
}
```

```
run:
Enter the number of terms for Fibonacci series: 12
Fibonacci Series: 0 1 1 2 3 5 8 13 21 34 55 89
BUILD SUCCESSFUL (total time: 3 seconds)
```

LAB # 03

Implementation of math class with various methods in java & Construct if else statements in java

Objective

To get familiarity with the math class with various methods in java & Implementation of various decisions making constructs.

LAB TASK

A. Take values of length and breadth of a rectangle from user and check if it is square or not.

```java
import java.util.Scanner;

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

        // Taking length and breadth as input
        System.out.print("Enter the length of the rectangle: ");
        double length = scanner.nextDouble();

        System.out.print("Enter the breadth of the rectangle: ");
        double breadth = scanner.nextDouble();

        // Check if it is a square
        if (length == breadth) {
            System.out.println("It is a square.");
        } else {
            System.out.println("It is not a square.");
        }
    }
}
```

```
run:
Enter the length of the rectangle: 31
Enter the breadth of the rectangle: 47
It is not a square.
BUILD SUCCESSFUL (total time: 8 seconds)
```

```
run:
Enter the length of the rectangle: 25
Enter the breadth of the rectangle: 25
It is a square.
BUILD SUCCESSFUL (total time: 17 seconds)
```

B. A school has following rules for grading system:

a.	Below 25	F
b.	25 to 45	E
c.	45 to 50	D
d.	50 to 60	C
e.	60 to 80	B
f.	Above 80	A

Write a program to enter marks and print the corresponding grade.

```java
import java.util.Scanner;

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

        // Taking marks as input
        System.out.print("Enter the marks of the student: ");
        int marks = scanner.nextInt();

        // Determine grade based on marks
        if (marks >= 90) {
            System.out.println("Grade: A");
        } else if (marks >= 80) {
            System.out.println("Grade: B");
        } else if (marks >= 70) {
            System.out.println("Grade: C");
        } else if (marks >= 60) {
            System.out.println("Grade: D");
        } else {
            System.out.println("Grade: F");
        }
    }
}
```

```
run:
Enter the marks of the student: 90
Grade: A
BUILD SUCCESSFUL (total time: 4 seconds)
```

```
run:
Enter the marks of the student: 65
Grade: D
BUILD SUCCESSFUL (total time: 5 seconds)
```

C. A student will not be allowed to sit in exam if his/her attendance is less than 75%. Take following input from user, Number of classes held, Number of classes attended and print percentage of class attended. Is student is allowed to sit in exam or not.

```java
import java.util.Scanner;

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

        // Taking number of classes held and attended
        System.out.print("Enter the number of classes held: ");
        int classesHeld = scanner.nextInt();

        System.out.print("Enter the number of classes attended: ");
        int classesAttended = scanner.nextInt();

        // Calculate attendance percentage
        double attendancePercentage = (double) classesAttended / classesHeld * 100;

        // Check if the student is allowed to sit in the exam
        System.out.println("Attendance Percentage: " + attendancePercentage + "%");
        if (attendancePercentage >= 75) {
            System.out.println("Student is allowed to sit in the exam.");
        } else {
            System.out.println("Student is not allowed to sit in the exam.");
        }
    }
}
```

```
run:
Enter the number of classes held: 45
Enter the number of classes attended: 40
Attendance Percentage: 88.88888888888889%
Student is allowed to sit in the exam.
BUILD SUCCESSFUL (total time: 9 seconds)
```

```
run:
Enter the number of classes held: 45
Enter the number of classes attended: 30
Attendance Percentage: 66.66666666666666%
Student is not allowed to sit in the exam.
BUILD SUCCESSFUL (total time: 19 seconds)
```

D. Explore if-else-if ladder and write a program of your choice to demonstrate its Implementation.

```java
import java.util.Scanner;

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

        // Taking age input
        System.out.print("Enter your age: ");
        int age = scanner.nextInt();

        // Using if-else-if ladder to check the category
        if (age < 13) {
            System.out.println("You are a child.");
        } else if (age >= 13 && age < 20) {
            System.out.println("You are a teenager.");
        } else if (age >= 20 && age < 60) {
            System.out.println("You are an adult.");
        } else {
            System.out.println("You are a senior citizen.");
        }
    }
}
```

```
run:
Enter your age: 13
You are a teenager.
BUILD SUCCESSFUL (total time: 4 seconds)
```

```
run:
Enter your age: 45
You are an adult.
BUILD SUCCESSFUL (total time: 8 seconds)
```

```
run:
Enter your age: 70
You are a senior citizen.
BUILD SUCCESSFUL (total time: 5 seconds)
```

E. Write a program that generates a random number between 1 and 25, then converts that number to its corresponding character using ASCII values.

```java
import java.util.Random;

public class RandomCharacter {
    public static void main(String[] args) {
        Random random = new Random();

        // Generate random number between 1 and 25
        int randomNumber = random.nextInt(25) + 1;

        // Convert the number to a corresponding ASCII character
        char character = (char) (randomNumber + 64);  // ASCII value of 'A' is 65
        System.out.println("Random Number: " + randomNumber);
        System.out.println("Corresponding Character: " + character);
    }
}
```

```
run:
Random Number: 23
Corresponding Character: W
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
run:
Random Number: 19
Corresponding Character: S
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
run:
Random Number: 10
Corresponding Character: J
BUILD SUCCESSFUL (total time: 0 seconds)
```

F. Write a program that calculates the distance between two points (x1, y1) and (x2, y2) using the distance formula, which involves Math.sqrt. $d=\sqrt{((x\_2-x\_1)^2+(y\_2-y\_1)^2)}$

```java
import java.util.Scanner;

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

        // Taking coordinates of two points
        System.out.print("Enter x1: ");
        double x1 = scanner.nextDouble();

        System.out.print("Enter y1: ");
        double y1 = scanner.nextDouble();

        System.out.print("Enter x2: ");
        double x2 = scanner.nextDouble();

        System.out.print("Enter y2: ");
        double y2 = scanner.nextDouble();

        // Calculate the distance using the formula
        double distance = Math.sqrt(Math.pow(x2 - x1, 2) + Math.pow(y2 - y1, 2));
        System.out.println("The distance between the points is: " + distance);
    }
}
```

```
run:
Enter x1: 30
Enter y1: 40
Enter x2: 10
Enter y2: 20
The distance between the points is: 28.284271247461902
BUILD SUCCESSFUL (total time: 18 seconds)
```

G. Write a program that takes two numbers as input, base and exponent, and calculates the result of base raised to the power of exponent using Math.pow.

```java
import java.util.Scanner;

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

        // Taking base and exponent as input
        System.out.print("Enter the base: ");
        double base = scanner.nextDouble();

        System.out.print("Enter the exponent: ");
        double exponent = scanner.nextDouble();

        // Calculate the result of base raised to the power of exponent
        double result = Math.pow(base, exponent);
        System.out.println("Result: " + result);
    }
}
```

```
run:
Enter the base: 12
Enter the exponent: 3
Result: 1728.0
BUILD SUCCESSFUL (total time: 18 seconds)
```

LAB # 04

Implementation of switch case statement in java & Construct various loops in java

Objective

Implementation of switch case statement in java. To gain familiarity with the concept in loops.

LAB TASK

A. Write a program using switch which choose between following cases:

a: Add two numbers
b: Find power using math function
c: Exit

```
1  package lab4;
2  import java.util.Scanner;
3
4  public class Calculator {
5      public static void main(String[] args) {
6          Scanner sc = new Scanner(System.in);
7          int choice;
8
9          do {
10             System.out.println("1. Add two numbers");
11             System.out.println("2. Find power using Math function");
12             System.out.println("3. Exit");
13             System.out.print("Enter your choice: ");
14             choice = sc.nextInt();
15
16             switch (choice) {
17                 case 1:
18                     System.out.print("Enter two numbers: ");
19                     int a = sc.nextInt();
20                     int b = sc.nextInt();
21                     System.out.println("Sum = " + (a + b));
22                     break;
23                 case 2:
24                     System.out.print("Enter base and exponent: ");
25                     double base = sc.nextDouble();
26                     double exponent = sc.nextDouble();
27                     System.out.println("Result = " + Math.pow(base, exponent));
28                     break;
29                 case 3:
30                     System.out.println("Exiting program.");
31                     break;
32                 default:
33                     System.out.println("Invalid choice!");
34             }
35         } while (choice != 3);
36     }
37 }
38
```

```
run:
1. Add two numbers
2. Find power using Math function
3. Exit
Enter your choice: 1
Enter two numbers: 23 34
Sum = 57
1. Add two numbers
2. Find power using Math function
3. Exit
Enter your choice: 2
Enter base and exponent: 2 5
Result = 32.0
1. Add two numbers
2. Find power using Math function
3. Exit
Enter your choice: 3
Exiting program.
BUILD SUCCESSFUL (total time: 32 seconds)
```

B. Write a program in Java to calculate the total cost of items purchased based on their category. The program should prompt the user to enter the category of the item (1 for electronics, 2 for groceries, and 3 for clothing) and the amount spent on that category. Then, calculate and display the total amount spent on each category separately. Use a switch-case statement to implement this program.

```java
import java.util.Scanner;
public class ItemCost {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double electronics = 0, groceries = 0, clothing = 0;

        System.out.print("Enter item category (1: Electronics, 2: Groceries, 3: Clothing): ");
        int category = sc.nextInt();

        System.out.print("Enter amount spent: ");
        double amount = sc.nextDouble();

        switch (category) {
            case 1:
                electronics += amount;
                System.out.println("Total spent on Electronics: " + electronics);
                break;
            case 2:
                groceries += amount;
                System.out.println("Total spent on Groceries: " + groceries);
                break;
            case 3:
                clothing += amount;
                System.out.println("Total spent on Clothing: " + clothing);
                break;
            default:
                System.out.println("Invalid category!");
        }
    }
}
```

```
run:
Enter item category (1: Electronics, 2: Groceries, 3: Clothing): 1
Enter amount spent: 2000
Total spent on Electronics: 2000.0
BUILD SUCCESSFUL (total time: 14 seconds)
```

```
Enter item category (1: Electronics, 2: Groceries, 3: Clothing): 2
Enter amount spent: 1500
Total spent on Groceries: 1500.0
BUILD SUCCESSFUL (total time: 9 seconds)
```

```
run:
Enter item category (1: Electronics, 2: Groceries, 3: Clothing): 3
Enter amount spent: 2400
Total spent on Clothing: 2400.0
BUILD SUCCESSFUL (total time: 7 seconds)
```

C. Write a program in Java to convert a numerical grade into a letter grade. The program should prompt the user to enter a numerical grade between 0 and 100 and then display the corresponding letter grade based on the following grading scale:

- 90-100: A
- 80-89: B
- 70-79: C
- 60-69: D
- Below 60: F

Use a switch-case statement to implement this program.

```java
package lab4;
import java.util.Scanner;

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

        System.out.print("Enter your numerical grade (0-100): ");
        int grade = input.nextInt();

        int gradeGroup = grade / 10;

        switch (gradeGroup) {
            case 10:
            case 9:
                System.out.println("Your grade is: A");
                break;
            case 8:
                System.out.println("Your grade is: B");
                break;
            case 7:
                System.out.println("Your grade is: C");
                break;
            case 6:
                System.out.println("Your grade is: D");
                break;
            default:
                if (grade >= 0 && grade < 60) {
                    System.out.println("Your grade is: F");
                } else {
                   System.out.println("Invalid grade entered.");
                }
        }
    }
}
```

```
run:
Enter your numerical grade (0-100): 89
Your grade is: B
BUILD SUCCESSFUL (total time: 6 seconds)
```

```
run:
Enter your numerical grade (0-100): 78
Your grade is: C
BUILD SUCCESSFUL (total time: 4 seconds)
```

D. Write a program in Java to calculate the area of different geometric shapes. The program should prompt the user to choose the shape (1 for circle, 2 for rectangle, and 3 for triangle) and then input the necessary parameters (radius for circle, length and width for rectangle, and base and height for triangle). Finally, calculate and display the area of the chosen shape. Use a switch-case statement to implement this program.

```
1  package lab4;
2  import java.util.Scanner;
3
4  public class AreaCalculator {
5      public static void main(String[] args) {
6          Scanner sc = new Scanner(System.in);
7          System.out.println("Choose shape: 1) Circle 2) Rectangle 3) Triangle");
8          int shape = sc.nextInt();
           switch (shape) {
               case 1:
11                 System.out.print("Enter radius: ");
12                 double r = sc.nextDouble();
13                 System.out.println("Area of circle: " + (Math.PI * r * r));
14                 break;
15             case 2:
16                 System.out.print("Enter length and width: ");
17                 double l = sc.nextDouble();
18                 double w = sc.nextDouble();
19                 System.out.println("Area of rectangle: " + (l * w));
20                 break;
21             case 3:
22                 System.out.print("Enter base and height: ");
23                 double base = sc.nextDouble();
24                 double height = sc.nextDouble();
25                 System.out.println("Area of triangle: " + (0.5 * base * height));
26                 break;
27             default:
28                 System.out.println("Invalid shape");
29         }
30     }
31 }
```

```
run:
Choose shape: 1) Circle 2) Rectangle 3) Triangle
2
Enter length and width: 34 24
Area of rectangle: 816.0
BUILD SUCCESSFUL (total time: 13 seconds)
```

```
run:
Choose shape: 1) Circle 2) Rectangle 3) Triangle
1
Enter radius: 45
Area of circle: 6361.725123519332
BUILD SUCCESSFUL (total time: 9 seconds)
```

```
run:
Choose shape: 1) Circle 2) Rectangle 3) Triangle
3
Enter base and height: 65 32
Area of triangle: 1040.0
BUILD SUCCESSFUL (total time: 11 seconds)
```

E. Write a program in java in which print 20 to 30 numbers using for loop.

```
1   package lab4;
2
3   public class ForLoopPrint {
4       public static void main(String[] args) {
5           for (int i = 20; i <= 30; i++) {
6               System.out.print(i + " ");
7           }
8       }
9   }
10
```

```
run:
20   21   22   23   24   25   26   27   28   29   30   BUILD SUCCESSFUL (total time: 0 seconds)
```

F. Write a program in java and print table of 2 using do while loop

```
1   package lab4;
2
3   public class TableOfTwo {
4       public static void main(String[] args) {
5           int i = 1;
6           do {
7               System.out.println("2 * " + i + " = " + (2 * i));
8               i++;
9           } while (i <= 10);
10      }
11  }
12
```

```
run:
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
2 * 10 = 20
BUILD SUCCESSFUL (total time: 0 seconds)
```

G. Write a program in java and print * 100 times using while loop.

```
1   package lab4;
2   public class StarPrinter {
3       public static void main(String[] args) {
4           int count = 0;
5           while (count < 100) {
6               System.out.print("*");
7               count++;
8           }
9       }
10  }
```

```
run:
**********BUILD SUCCESSFUL (total time: 0 seconds)
```

H. Write a program to print odd number from 1 to 100 using all three loops.

```java
package lab4;

public class OddNumbers {
    public static void main(String[] args) {
        System.out.println("Using for loop:");
        for (int i = 1; i <= 100; i += 2) {
            System.out.print(i + " ");
        }

        System.out.println("\n\nUsing while loop:");
        int j = 1;
        while (j <= 100) {
            System.out.print(j + " ");
            j += 2;
        }

        System.out.println("\n\nUsing do-while loop:");
        int k = 1;
        do {
            System.out.print(k + " ");
            k += 2;
        } while (k <= 100);
    }
}
```

```
run:
Using for loop:
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99

Using while loop:
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99

Using do-while loop:
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

I. Write a program to print diamond pattern using nested while and for loop in java

```java
package lab4;
public class DiamondPattern {
    public static void main(String[] args) {
        int n = 5;

        int i = 1;
        while (i <= n) {
            for (int j = i; j < n; j++) System.out.print(" ");
            for (int j = 1; j <= (2 * i - 1); j++) System.out.print("*");
            System.out.println();
            i++;
        }

        i = n - 1;
        while (i >= 1) {
            for (int j = n; j > i; j--) System.out.print(" ");
            for (int j = 1; j <= (2 * i - 1); j++) System.out.print("*");
            System.out.println();
            i--;
        }
    }
}

```

```
run:
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 05

Implementation of break and continue keyword in java & Construct 1D arrays in java

Objective

To gain familiarity with the Implementation of break and continue keyword in java. To Construct 1D arrays in java.

LAB TASK

A. Write a Java program to search for a specific character in a given string. If found, print "Character found at index" followed by its index. If not found, print "Character not found".

INPUT:

```
1   package lab5;
2
3   public class SearchCharacter {
4       public static void main(String[] args) {
5           String str = "HelloWorld";
6           char target = 'o';
7           int index = str.indexOf(target);
8
9           if (index != -1)
10              System.out.println("Character found at index " + index);
11          else
12              System.out.println("Character not found");
13      }
14  }
15
```

OUTPUT:

```
run:
Character found at index 4
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
run:
Character found at index 9
BUILD SUCCESSFUL (total time: 0 seconds)
```

B. Write a Java program to print the first 10 numbers of the Fibonacci sequence, skipping numbers that are divisible by 3.

INPUT:

```
1   package lab5;
2
3   public class FibonacciSkip3 {
4       public static void main(String[] args) {
5           int a = 0, b = 1, count = 0;
6           while (count < 10) {
7               int next = a + b;
8               if (next % 3 != 0) {
9                   System.out.print(next + " ");
10                  count++;
11              }
12              a = b;
13              b = next;
14          }
15      }
16  }
17
```

OUTPUT:

```
run:
1 2 5 8 13 34 55 89 233 377 BUILD SUCCESSFUL (total time: 0 seconds)
```

C. Write a Java program to find the first non-repeated character in a given string. If all characters are repeated, print "All characters are repeated".

INPUT:

```
1  package lab5;
2
3  public class FirstUniqueChar {
4      public static void main(String[] args) {
5          String str = "MuhammadAsadKhan";
6          boolean found = false;
7
8          for (int i = 0; i < str.length(); i++) {
9              char c = str.charAt(i);
10             if (str.indexOf(c) == str.lastIndexOf(c)) {
11                 System.out.println("First non-repeated character: " + c);
12                 found = true;
13                 break;
14             }
15         }
16
17         if (!found) {
18             System.out.println("All characters are repeated");
19         }
20     }
21 }
```

OUTPUT:

```
run:
First non-repeated character: M
BUILD SUCCESSFUL (total time: 0 seconds)
```

D. Create an array of String variables and initialize the array with the names of the months from January to December. Create an array containing 12 random decimal values between 0.0 and 100.0. Display the names of each month along with the corresponding decimal value. Calculate and display the average of the 12 values.

INPUT:

```
1   package lab5;
2
3   import java.util.Random;
4
5   public class MonthValues {
6       public static void main(String[] args) {
7           String[] months = {
8               "January", "February", "March", "April", "May", "June",
9               "July", "August", "September", "October", "November", "December"
10          };
11
12          double[] values = new double[12];
13          Random rand = new Random();
14          double sum = 0;
15
16          for (int i = 0; i < 12; i++) {
17              values[i] = rand.nextDouble() * 100;
18              System.out.printf("%s: %.2f%n", months[i], values[i]);
19              sum += values[i];
20          }
21
22          System.out.printf("Average value: %.2f%n", sum / 12);
23      }
24  }
```

OUTPUT:

```
run:
January: 3.66
February: 82.21
March: 46.17
April: 99.66
May: 50.86
June: 60.35
July: 62.89
August: 50.31
September: 39.49
October: 95.12
November: 44.48
December: 23.38
Average value: 54.88
BUILD SUCCESSFUL (total time: 0 seconds)
```

E. Write a Java program to calculate the average value of array elements.

INPUT:

```
1  package lab5;
2
3  public class ArrayAverage {
4      public static void main(String[] args) {
5          int[] arr = {10, 20, 30, 40, 50};
6          int sum = 0;
7
8          for (int value : arr) {
9              sum += value;
10         }
11
12         double average = (double) sum / arr.length;
13         System.out.println("Average: " + average);
14     }
15 }
16
```

OUTPUT:

```
run:
Average: 30.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

F. Write a Java program to reverse an array of integer values.

INPUT:

```java
package lab5;

import java.util.Arrays;

public class ReverseArray {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5};

        for (int i = 0; i < arr.length / 2; i++) {
            int temp = arr[i];
            arr[i] = arr[arr.length - 1 - i];
            arr[arr.length - 1 - i] = temp;
        }

        System.out.println("Reversed Array: " + Arrays.toString(arr));
    }
}
```

OUTPUT:

```
run:
Reversed Array: [5, 4, 3, 2, 1]
BUILD SUCCESSFUL (total time: 0 seconds)
```

G. Write a Java program to sort a numeric array and a string array.

INPUT:

```java
package lab5;

import java.util.Arrays;

public class SortArrays {
    public static void main(String[] args) {
        int[] nums = {3, 1, 4, 1, 5, 9};
        String[] words = {"banana", "apple", "cherry"};

        Arrays.sort(nums);
        Arrays.sort(words);

        System.out.println("Sorted Numbers: " + Arrays.toString(nums));
        System.out.println("Sorted Strings: " + Arrays.toString(words));
    }
}
```

OUTPUT:

```
run:
Sorted Numbers: [1, 1, 3, 4, 5, 9]
Sorted Strings: [apple, banana, cherry]
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 06

Implementation of 2D arrays in java

Objective

To gain familiarity with the Implementation of 2D arrays in java.

LAB TASK

A. Write a program to create a rectangular array containing a multiplication table from 1 x 1 up to 12 x 12. Output the table as 13 columns with the numeric values right aligned in columns. (The first line of the output will be the column headings, the first column with no heading, then the numbers 1 to 12 for the remaining columns. The first item in each of the succeeding lines is the row heading, which ranges from 1 to 12.).

INPUT:

```
1   package lab6;
2
3   public class MultiplicationTable {
4       public static void main(String[] args) {
5           System.out.printf("%4s", " ");
6           for (int i = 1; i <= 12; i++) {
7               System.out.printf("%4d", i);
8           }
9           System.out.println();
10
11          for (int i = 1; i <= 12; i++) {
12              System.out.printf("%4d", i);
13              for (int j = 1; j <= 12; j++) {
14                  System.out.printf("%4d", i * j);
15              }
16              System.out.println();
17          }
18      }
19  }
```

OUTPUT:

```
run:
      1   2   3   4   5   6   7   8   9  10  11  12
 1    1   2   3   4   5   6   7   8   9  10  11  12
 2    2   4   6   8  10  12  14  16  18  20  22  24
 3    3   6   9  12  15  18  21  24  27  30  33  36
 4    4   8  12  16  20  24  28  32  36  40  44  48
 5    5  10  15  20  25  30  35  40  45  50  55  60
 6    6  12  18  24  30  36  42  48  54  60  66  72
 7    7  14  21  28  35  42  49  56  63  70  77  84
 8    8  16  24  32  40  48  56  64  72  80  88  96
 9    9  18  27  36  45  54  63  72  81  90  99 108
10   10  20  30  40  50  60  70  80  90 100 110 120
11   11  22  33  44  55  66  77  88  99 110 121 132
12   12  24  36  48  60  72  84  96 108 120 132 144
BUILD SUCCESSFUL (total time: 0 seconds)
```

B. Write a program to find the transpose of a 2x3 matrix.

INPUT:

```java
package lab6;

public class TransposeMatrix {
    public static void main(String[] args) {
        int[][] matrix = {{1, 2, 3}, {4, 5, 6}};
        int[][] transpose = new int[3][2];

        for (int i = 0; i < 2; i++) {
            for (int j = 0; j < 3; j++) {
                transpose[j][i] = matrix[i][j];
            }
        }

        System.out.println("Transpose of the matrix:");
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 2; j++) {
                System.out.print(transpose[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

OUTPUT:

```
run:
Transpose of the matrix:
1 4
2 5
3 6
BUILD SUCCESSFUL (total time: 0 seconds)
```

C. Write a program to check if a 3x3 2D array is symmetric or not. A 2D array is symmetric if matrix[i][j] == matrix[j][i] for all valid i and j.

INPUT:

```
1   package lab6;
2
3   public class SymmetricCheck {
4       public static void main(String[] args) {
5           int[][] matrix = {
6               {1, 2, 3},
7               {2, 5, 6},
8               {3, 6, 9}
9           };
10          boolean isSymmetric = true;
11
12          for (int i = 0; i < 3; i++) {
13              for (int j = 0; j < 3; j++) {
14                  if (matrix[i][j] != matrix[j][i]) {
15                      isSymmetric = false;
16                      break;
17                  }
18              }
19          }
20
21          if (isSymmetric)
22              System.out.println("Matrix is symmetric.");
23          else
24              System.out.println("Matrix is not symmetric.");
25      }
26  }
```

OUTPUT:

```
run:
Matrix is symmetric.
BUILD SUCCESSFUL (total time: 0 seconds)
```

D. Write a program to find the sum of each row in a 3x3 2D array and print the sum for each row.

INPUT:

```
1   package lab6;
2
3   public class RowSum {
4       public static void main(String[] args) {
5           int[][] matrix = {
6               {1, 2, 3},
7               {4, 5, 6},
8               {7, 8, 9}
9           };
10
11          for (int i = 0; i < 3; i++) {
12              int sum = 0;
13              for (int j = 0; j < 3; j++) {
14                  sum += matrix[i][j];
15              }
16              System.out.println("Sum of row " + (i+1) + ": " + sum);
17          }
18      }
19  }
```

OUTPUT:

```
run:
Sum of row 1: 6
Sum of row 2: 15
Sum of row 3: 24
BUILD SUCCESSFUL (total time: 0 seconds)
```

E. Write a program to print a pyramid pattern of asterisks with 5 rows.

INPUT:

```
1   package lab6;
2
3   public class PyramidPattern {
4       public static void main(String[] args) {
5           int rows = 5;
6           for (int i = 1; i <= rows; i++) {
7               for (int space = 1; space <= rows - i; space++) {
8                   System.out.print(" ");
9               }
10              for (int star = 1; star <= 2 * i - 1; star++) {
11                  System.out.print("*");
12              }
13              System.out.println();
14          }
15      }
16  }
17
```

OUTPUT:

```
run:
    *
   ***
  *****
 *******
*********
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 07

Manipulation of String in java.

Objective

Introducing string class and to explore the strings class built-in methods.

LAB TASK:

A. Construct a program which shows the implementation of concat() string class builtin method.

INPUT:

```
1   package lab.pkg9;
2
3   public class Concat {
4       public static void main(String[] args) {
5           String str1 = "Hello";
6           String str2 = " World";
7           String result = str1.concat(str2);
8           System.out.println("Concatenated String: " + result);
9       }
10  }
```

OUTPUT:

```
run:
Concatenated String: Hello World
BUILD SUCCESSFUL (total time: 0 seconds)
```

B. Construct a program which shows the implementation of compareTo() string class builtin method.

INPUT:

```
1   package lab.pkg9;
2
3   public class CompareToExample {
4       public static void main(String[] args) {
5           String str1 = "Apple";
6           String str2 = "Banana";
7           int result = str1.compareTo(str2);
8           System.out.println("Result of compareTo(): " + result);
9       }
10  }
```

OUTPUT:

```
run:
Result of compareTo(): -1
BUILD SUCCESSFUL (total time: 0 seconds)
```

C. Construct a program which shows the implementation of equals() string class built-in method.

INPUT:

```
1   package lab.pkg9;
2
3   public class EqualsExample {
4       public static void main(String[] args) {
5           String str1 = "Hello";
6           String str2 = "Hello";
7           String str3 = "World";
8           System.out.println("str1 equals str2: " + str1.equals(str2));
9           System.out.println("str1 equals str3: " + str1.equals(str3));
10      }
11  }
```

OUTPUT:

```
run:
str1 equals str2: true
str1 equals str3: false
BUILD SUCCESSFUL (total time: 0 seconds)
```

D. Construct a program which shows the implementation of length() string class built-in method.

INPUT:

```
1   package lab.pkg9;
2
3   public class LengthExample {
4       public static void main(String[] args) {
5           String str = "Programming";
6           int length = str.length();
7           System.out.println("Length of the string: " + length);
8       }
9   }
```

OUTPUT:

```
run:
Length of the string: 11
BUILD SUCCESSFUL (total time: 0 seconds)
```

E. Construct a program which shows the implementation of to Uppercase() string class built-in method.

INPUT:

```
1   package lab.pkg9;
2
3   public class ToUpperCaseExample {
4       public static void main(String[] args) {
5           String str = "hello world";
6           String upperStr = str.toUpperCase();
7           System.out.println("Uppercase String: " + upperStr);
8       }
9   }
10
```

OUTPUT:

```
run:
Uppercase String: HELLO WORLD
BUILD SUCCESSFUL (total time: 0 seconds)
```

F. Write a Java program that demonstrates the usage of String Buffer. Your program should prompt the user to enter two strings, concatenate them using String Buffer, and then reverse the concatenated string. Finally, output the reversed string.

INPUT:

```
1   package lab.pkg9;
2
3   import java.util.Scanner;
4
5   public class StringBufferExample {
6       public static void main(String[] args) {
7           Scanner sc = new Scanner(System.in);
8
9           System.out.print("Enter first string: ");
10          String str1 = sc.nextLine();
11
12          System.out.print("Enter second string: ");
13          String str2 = sc.nextLine();
14
15          StringBuffer sb = new StringBuffer(str1);
16          sb.append(str2);
17
18          System.out.println("Concatenated String: " + sb);
19
20          sb.reverse();
21          System.out.println("Reversed Concatenated String: " + sb);
22
23      }
24  }
25
```

OUTPUT:

```
run:
Enter first string: Welcome to
Enter second string:  Java Programming
Concatenated String: Welcome to Java Programming
Reversed Concatenated String: gnimmargorP avaJ ot emocleW
BUILD SUCCESSFUL (total time: 22 seconds)
```

G. Write a Java program that demonstrates the usage of String Builder. Your program should initialize a StringBuilder object with the value "Java is", then insert the text "awesome" at index 7, and finally replace "awesome" with "powerful". Output the final string.

INPUT:

```
1   package lab.pkg9;
2
3   public class StringBuilderExample {
4       public static void main(String[] args) {
5           StringBuilder sb = new StringBuilder("Java is");
6
7           sb.insert(7, "awesome ");
8           System.out.println("After insert: " + sb);
9
10          int start = sb.indexOf("awesome");
11          int end = start + "awesome".length();
12          sb.replace(start, end, "powerful");
13
14          System.out.println("After replace: " + sb);
15      }
16  }
17
```

OUTPUT:

```
run:
After insert: Java isawesome
After replace: Java ispowerful
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 08

Construct classes and objects in java using NetBeans IDE. (methods)

Objective

Get familiar with concept of classes and objects in java.

LAB TASK

A. Write a program with a method named getTotal that accepts two integers as an argument and return its sum. Call this method from main( ) and print the results. INPUT:

```
1   package lab8;
2   public class SumExample {
3       public static int getTotal(int a, int b) {
4           return a + b;
5       }
6
7       public static void main(String[] args) {
8           int result = getTotal(10, 20);
9           System.out.println("Sum is: " + result);
10      }
11  }
12
```

OUTPUT:

```
run:
Sum is: 30
BUILD SUCCESSFUL (total time: 0 seconds)
```

B. Write a program that create two classes one is Power_law another is the main method class. Power_law class contains current, voltage and power fields in integer datatype, and a method to calculate power and display the output.

INPUT:

```
1   package lab8;
2   class Power_law {
3       int current;
4       int voltage;
5       int power;
6
7       void calculatePower() {
8           power = current * voltage;
9           System.out.println("Power = " + power + " watts");
10      }
11  }
12
13  public class PowerMain {
14      public static void main(String[] args) {
15          Power_law obj = new Power_law();
16          obj.current = 5;
17          obj.voltage = 220;
18          obj.calculatePower();
19      }
20  }
```

OUTPUT:

```
run:
Power = 1100 watts
BUILD SUCCESSFUL (total time: 0 seconds)
```

C. Assign and print the roll number, phone number and address of two students having names "Babar" and "Danish" respectively by creating two objects of class 'Student'.

INPUT:

```
1   package lab8;
2   class Student {
3       int rollNo;
4       String phone;
5       String address;
6       String name;
7
8       void display() {
9           System.out.println("Name: " + name);
10          System.out.println("Roll No: " + rollNo);
11          System.out.println("Phone: " + phone);
12          System.out.println("Address: " + address);
13          System.out.println();
14      }
15  }
16
17  public class StudentMain {
18      public static void main(String[] args) {
19          Student babar = new Student();
20          babar.name = "Babar";
21          babar.rollNo = 1;
22          babar.phone = "0300-1234567";
23          babar.address = "Karachi";
24
25          Student danish = new Student();
26          danish.name = "Danish";
27          danish.rollNo = 2;
28          danish.phone = "0301-7654321";
29          danish.address = "Lahore";
30
31          babar.display();
32          danish.display();
33      }
34  }
35
```

OUTPUT:

```
run:
Name: Babar
Roll No: 1
Phone: 0300-1234567
Address: Karachi

Name: Danish
Roll No: 2
Phone: 0301-7654321
Address: Lahore

BUILD SUCCESSFUL (total time: 0 seconds)
```

D. Add two distances in inch-feet by creating a class named 'AddDistance' INPUT:

```java
package lab8;
class AddDistance {
    int feet;
    int inches;

    AddDistance(int f, int i) {
        feet = f;
        inches = i;
    }

    AddDistance add(AddDistance d) {
        int totalFeet = this.feet + d.feet;
        int totalInches = this.inches + d.inches;

        if (totalInches >= 12) {
            totalFeet += totalInches / 12;
            totalInches = totalInches % 12;
        }

        return new AddDistance(totalFeet, totalInches);
    }

    void display() {
        System.out.println("Distance: " + feet + " feet " + inches + " inches");
    }
}
public class DistanceMain {
    public static void main(String[] args) {
        AddDistance d1 = new AddDistance(5, 9);
        AddDistance d2 = new AddDistance(3, 11);

        AddDistance result = d1.add(d2);
        result.display();
    }
}
```

OUTPUT:

```
run:
Distance: 9 feet 8 inches
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 9

Implementation of constructors in java, Construct java programs using the concepts of method and constructor overloading

Objective

Implementation of constructors in java, Construct java programs using the concepts of method and constructor overloading.

LAB TASK

A. Write a program that creates two classes one is Student another is the main method class. Student contains private fields of roll-no, semester, GPA, and use a constructor and a method to get and show the above information of three students using get and show method in java.

INPUT:

```java
import java.util.Scanner;

class Student {
    private int rollNo;
    private int semester;
    private double gpa;

    Student() {
        rollNo = 0;
        semester = 0;
        gpa = 0.0;
    }

    public void get(Scanner input) {
        System.out.print("Enter Roll Number: ");
        rollNo = input.nextInt();
        System.out.print("Enter Semester: ");
        semester = input.nextInt();
        System.out.print("Enter GPA: ");
        gpa = input.nextDouble();
    }

    public void show() {
        System.out.println("Roll No: " + rollNo);
        System.out.println("Semester: " + semester);
        System.out.println("GPA: " + gpa);
        System.out.println("-------------------------");
    }
}

public class StudentInfo {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        Student[] students = new Student[3];

        for (int i = 0; i < 3; i++) {
            System.out.println("Enter details for Student " + (i + 1));
            students[i] = new Student();
            students[i].get(input);
        }

        System.out.println("\n--- Student Details ---");
        for (int i = 0; i < 3; i++) {
            students[i].show();
        }

        input.close();
    }
}
```

OUTPUT:

```
run:
Enter details for Student 1
Enter Roll Number: 101
Enter Semester: 2
Enter GPA: 3.5
Enter details for Student 2
Enter Roll Number: 102
Enter Semester: 2
Enter GPA: 3.4
Enter details for Student 3
Enter Roll Number: 103
Enter Semester: 2
Enter GPA: 3.6

--- Student Details ---
Roll No: 101
Semester: 2
GPA: 3.5
-------------------------
Roll No: 102
Semester: 2
GPA: 3.4
-------------------------
Roll No: 103
Semester: 2
GPA: 3.6
-------------------------
BUILD SUCCESSFUL (total time: 35 seconds)
```

B. Write a Java program to implement a Rectangle class with the following specifications:

1. **The class should have two instance variables: length and width, both of type integer.**
2. **Implement a constructor that takes two parameters l and w and initializes the length and width variables accordingly.**
3. **Implement a method named calculateArea that calculates and returns the area of the rectangle (length * width).**
4. **In the main method:**
   - **Create an object of the Rectangle class with length 5 and width 4.**
   - **Call the calculateArea method on the object and store the result. Print the calculated area.**

INPUT:

```
class Rectangle {
    int length;
    int width;

    Rectangle(int l, int w) {
        length = l;
        width = w;
    }

    int calculateArea() {
        return length * width;
    }
}

public class Rect {
    public static void main(String[] args) {
        // Create a Rectangle object with length 5 and width 4
        Rectangle rect = new Rectangle(5, 4);

        int area = rect.calculateArea();

        System.out.println("The area of the rectangle is: " + area);
    }
}
```

OUTPUT:

```
run:
The area of the rectangle is: 20
BUILD SUCCESSFUL (total time: 0 seconds)
```

C. What is the "type promotion Overloading"? Explore and demonstrate with your own program.

INPUT:

```
class PromotionExample {
    void show(int x) {
        System.out.println("int version called with: " + x);
    }

    void show(double x) {
        System.out.println("double version called with: " + x);
    }
}

public class TypePromotion {
    public static void main(String[] args) {
        PromotionExample obj = new PromotionExample();

        byte b = 10;
        obj.show(b); // byte → int (promoted)

        float f = 5.6f;
        obj.show(f); // float → double (promoted)

        char ch = 'A';
        obj.show(ch); // char → int (promoted)
    }
}
```

OUTPUT:

```
run:
int version called with: 10
double version called with: 5.599999904632568
int version called with: 65
BUILD SUCCESSFUL (total time: 0 seconds)
```

D. Create a class to print the area of a square and a rectangle. The class has two methods with the same name but different number of parameters. The method for printing area of rectangle has two parameters which are length and breadth respectively while the other method for printing area of square has one parameter which is side of square.

INPUT:

```
class Shape {

    void area(int side) {
        int result = side * side;
        System.out.println("Area of square: " + result);
    }

    void area(int length, int breadth) {
        int result = length * breadth;
        System.out.println("Area of rectangle: " + result);
    }
}

public class Area {
    public static void main(String[] args) {
        Shape obj = new Shape();
        obj.area(5);
        obj.area(5, 3);
    }
}
```

OUTPUT:

```
run:
Area of square: 25
Area of rectangle: 15
BUILD SUCCESSFUL (total time: 0 seconds)
```

E. Define a class with the name "Mathematics" and define a method "Square()" for calculating the square of any given number. Now redefine this method with different type of argument variables, but with the same name "quare()".

INPUT:

```
1   class Mathematics {
2
3       void Square(int number) {
4           System.out.println("Square of int: " + (number * number));
5       }
6
7       void Square(float number) {
8           System.out.println("Square of float: " + (number * number));
9       }
10
11      void Square(double number) {
12          System.out.println("Square of double: " + (number * number));
13      }
14  }
15
16  public class MethodOverLoading {
17      public static void main(String[] args) {
18          Mathematics m = new Mathematics();
19          m.Square(5);
20          m.Square(4.2f);
21          m.Square(6.7);
22      }
23  }
```

OUTPUT:

```
run:
Square of int: 25
Square of float: 17.639997
Square of double: 44.89
BUILD SUCCESSFUL (total time: 0 seconds)
```

F. Write a program to print the names of students by creating a Student class. If no name is passed while creating an object of Student class, then the name should be "Unknown", otherwise the name should be equal to the String value passed while creating object of Student class.

INPUT:

```
class Student {
    String name;

    Student(String n) {
        name = n;
    }

    Student() {
        name = "Unknown";
    }

    void showName() {
        System.out.println("Student Name: " + name);
    }
}

public class PConstructor{
    public static void main(String[] args) {
        Student s1 = new Student("Asad");
        Student s2 = new Student();

        s1.showName();
        s2.showName();
    }
}
```

OUTPUT:

```
run:
Student Name: Asad
Student Name: Unknown
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 10
Manipulation of inheritance and super keyword in java

Objective

To study inheritance and types of inheritance and using super to call superclass constructors.

Lab Task

A. Create a class Card, and its children classes are Valentine, Holiday, and Birthday. A card class will have a greeting () method that writes out a greeting. Each type of card contains an appropriate greeting. The Holiday card says "Season's Greetings." The Birthday card says "Happy Birthday." The Valentine card says "Happy Valentine Day". Create objects of the three child classes: Valentine, Holiday, and Birthday and show the polymorphic behavior (call the greeting methods from each object).

INPUT:

```
1   package lab10;
2
    class Card {
        public void greeting() {
5           System.out.println("Generic Greeting.");
6       }
7   }
8
9   class Valentine extends Card {
10      @Override
        public void greeting() {
12          System.out.println("Happy Valentine Day");
13      }
14  }
15
16  class Holiday extends Card {
17      @Override
        public void greeting() {
19          System.out.println("Season's Greetings");
20      }
21  }
22
23  class Birthday extends Card {
24      @Override
        public void greeting() {
26          System.out.println("Happy Birthday");
27      }
28  }
29
```

```
30  public class GreetingTest {
31      public static void main(String[] args) {
32          Card val = new Valentine();
33          Card hol = new Holiday();
34          Card bday = new Birthday();
35
36          val.greeting();
37          hol.greeting();
38          bday.greeting();
39      }
40  }
41
```

OUTPUT:

```
run:
Happy Valentine Day
Season's Greetings
Happy Birthday
BUILD SUCCESSFUL (total time: 0 seconds)
```

B. Develop a registration system for a University. It should consist of three classes namely Student, Teacher, and Course. For our example, a student needs to have a name, roll number, address

and GPA to be eligible for registration. Therefore choose appropriate data types for encapsulating these properties in a Student object/s. Similarly a teacher needs to have name, address, telephone number, and a degree (or a list of degrees) he has received. Finally courses must have a name, students (5 students) registered for the course, and a teacher assigned to conduct the course. Create an object of Course with 5 Students and a Teacher. A call to a method, say printDetails(), of the selected course reference should print name of the course, details of the teacher assigned to that course, and names and roll numbers of the students enrolled with the course.

INPUT:

```
1   class Student {
2       String name;
3       String rollNo;
        String address;
        double gpa;
6
7       Student(String name, String rollNo, String address, double gpa) {
8           this.name = name;
9           this.rollNo = rollNo;
10          this.address = address;
11          this.gpa = gpa;
12      }
13  }
14
15  class Teacher {
16      String name;
        String address;
        String phone;
19      String[] degrees;
20
21      Teacher(String name, String address, String phone, String[] degrees) {
22          this.name = name;
23          this.address = address;
24          this.phone = phone;
25          this.degrees = degrees;
26      }
27  }
28
```

```
29    class Course {
30        String courseName;
31        Student[] students;
32        Teacher teacher;
33
34        Course(String courseName, Student[] students, Teacher teacher) {
35            this.courseName = courseName;
36            this.students = students;
37            this.teacher = teacher;
38        }
39
40        void printDetails() {
41            System.out.println("Course Name: " + courseName);
42            System.out.println("Teacher: " + teacher.name + ", Degrees: ");
43            for (String d : teacher.degrees) {
44                System.out.println(" - " + d);
45            }
46
47            System.out.println("\nEnrolled Students:");
48            for (Student s : students) {
49                System.out.println("Name: " + s.name + ", Roll No: " + s.rollNo);
50            }
51        }
52    }
53
54    public class UniversitySystem {
55        public static void main(String[] args) {
56            Student[] students = {
57                new Student("Ali", "CS001", "Karachi", 3.2),
58                new Student("Sara", "CS002", "Lahore", 3.5),
59                new Student("Ayan", "CS003", "Islamabad", 3.0),
60                new Student("Hina", "CS004", "Peshawar", 3.7),
61                new Student("Usman", "CS005", "Quetta", 2.9)
62            };
63
64            String[] degrees = {"BSCS", "MSCS"};
65            Teacher teacher = new Teacher("Dr. Ahmed", "Karachi", "03211234567", degrees);
66
67            Course course = new Course("Object-Oriented Programming", students, teacher);
68            course.printDetails();
69        }
70    }
71
```

OUTPUT:

```
run:
Course Name: Object-Oriented Programming
Teacher: Dr. Ahmed, Degrees:
 - BSCS
 - MSCS

Enrolled Students:
Name: Ali, Roll No: CS001
Name: Sara, Roll No: CS002
Name: Ayan, Roll No: CS003
Name: Hina, Roll No: CS004
Name: Usman, Roll No: CS005
BUILD SUCCESSFUL (total time: 0 seconds)
```

C. Create a superclass Vehicle with a method start(). Then, create a subclass Car that extends Vehicle and adds a method drive(). Demonstrate single inheritance by creating an instance of Car and calling both methods.

INPUT:

```java
class Vehicle {
    void start() {
        System.out.println("Vehicle started.");
    }
}

class Car extends Vehicle {
    void drive() {
        System.out.println("Car is driving.");
    }
}

public class SingleInheritance {
    public static void main(String[] args) {
        Car car = new Car();
        car.start();
        car.drive();
    }
}
```

OUTPUT:

```
run:
Vehicle started.
Car is driving.
BUILD SUCCESSFUL (total time: 0 seconds)
```

D. Create a superclass Person with a method walk(). Create a subclass Employee that extends Person and adds a method work(). Then, create another subclass Manager that extends Employee and adds a method lead(). Demonstrate multilevel inheritance by creating an instance of Manager and calling all three methods.

INPUT:

```
run:
Person is walking.
Employee is working.
Manager is leading.
BUILD SUCCESSFUL (total time: 0 seconds)
```

```java
class Person {
    void walk() {
        System.out.println("Person is walking.");
    }
}

class Employee extends Person {
    void work() {
        System.out.println("Employee is working.");
    }
}

class Manager extends Employee {
    void lead() {
        System.out.println("Manager is leading.");
    }
}

public class MultilevelInheritance {
    public static void main(String[] args) {
        Manager mgr = new Manager();
        mgr.walk();
        mgr.work();
        mgr.lead();
    }
}
```

OUTPUT:

E. Create a superclass Device with a method powerOn(). Create two subclasses Phone and Laptop that both extend Device. Phone should add a method call(), and Laptop should add a method code(). Demonstrate hierarchical inheritance by creating instances of Phone and Laptop, and calling their respective methods along with the inherited method.

INPUT:

```java
class Device {
    void powerOn() {
        System.out.println("Device powered on.");
    }
}

class Phone extends Device {
    void call() {
        System.out.println("Phone is calling.");
    }
}

class Laptop extends Device {
    void code() {
        System.out.println("Laptop is coding.");
    }
}

public class HierarchicalInheritance {
    public static void main(String[] args) {
        Phone phone = new Phone();
        Laptop laptop = new Laptop();

        phone.powerOn();
        phone.call();

        laptop.powerOn();
        laptop.code();
    }
}
```

OUTPUT:

```
run:
Device powered on.
Phone is calling.
Device powered on.
Laptop is coding.
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 09 b

Open Ended Lab

Objective

Implementation of open ended lab.

Lab Goals

To provide working insight into the usage of operators, conditional statements, loop, math class, random class in java programming:

a. Conditional Statement.
b. Operators
c. Loop
d. Math Class
e. Random Class

Exercise:

A. Given Scenario: You are developing a simple Java application for a mathematics quiz game. In this quiz, the player will be asked to solve basic arithmetic problems such as addition, subtraction, multiplication, and division. You want to implement a loop to keep the game running until the player decides to exit. Additionally, you want to include a condition where the player can choose the difficulty level of the questions. To achieve this, you decide to use the Math class to generate random numbers for the questions.

Task: To solve the scenario follow the below given guidelines:

1. Write a Java program that implements the mathematics quiz game with the following specifications:
2. The program should display a menu to the player where they can choose the difficulty level of the questions (Easy, Medium, or Hard).
3. Based on the difficulty level chosen by the player, the program should generate random arithmetic questions accordingly.
4. After each question, the player should be prompted to enter their answer.
5. If the player's answer is correct, display a message congratulating them; otherwise, display a message indicating the correct answer.
6. After each question, ask the player if they want to continue playing. If they choose to continue, present them with another question; if not, exit the game.
7. Use loops to keep the game running until the player decides to exit.

Your task is to write the Java code to implement the above specifications.

INPUT:

```
1  package openendedlab1;
2
3  import java.util.Scanner;
4  import java.util.Random;
5
6  public class MathQuizGame {
7      public static void main(String[] args) {
           Scanner input = new Scanner(System.in);
9          Random rand = new Random();
10         System.out.println("Welcome to Math Quiz Game!");
11
12         boolean playAgain = true;
13
14         while (playAgain) {
15             System.out.println("\nChoose Difficulty Level:");
16             System.out.println("1. Easy");
17             System.out.println("2. Medium");
18             System.out.println("3. Hard");
19             System.out.print("Enter your choice: ");
20             int choice = input.nextInt();
21
22             int num1, num2;
23             String operation = "";
24             double correctAnswer = 0;
25
               switch (choice) {
27                 case 1:
28                     num1 = rand.nextInt(10) + 1;
29                     num2 = rand.nextInt(10) + 1;
30                     break;
31                 case 2:
32                     num1 = rand.nextInt(50) + 10;
33                     num2 = rand.nextInt(50) + 10;
34                     break;
35                 case 3:
36                     num1 = rand.nextInt(100) + 50;
37                     num2 = rand.nextInt(100) + 50;
38                     break;
39                 default:
40                     System.out.println("Invalid choice. Setting difficulty to Easy.");
41                     num1 = rand.nextInt(10) + 1;
42                     num2 = rand.nextInt(10) + 1;
43                     break;
44             }
45
46             int op = rand.nextInt(4);
               switch (op) {
48                 case 0:
49                     operation = "+";
50                     correctAnswer = num1 + num2;
51                     break;
52                 case 1:
53                     operation = "-";
54                     correctAnswer = num1 - num2;
55                     break;
```

```
                case 2:
                    operation = "*";
                    correctAnswer = num1 * num2;
                    break;
                case 3:
                    operation = "/";
                    while (num2 == 0 || num1 % num2 != 0) {
                        num1 = rand.nextInt(10) + 1;
                        num2 = rand.nextInt(9) + 1;
                    }
                    correctAnswer = (double) num1 / num2;
                    break;
            }

            System.out.printf("%d %s %d = ?\n", num1, operation, num2);
            double userAnswer = input.nextDouble();

            if (Math.abs(userAnswer - correctAnswer) < 0.0001) {
                System.out.println("Congratulations! Your answer is correct.");
            } else {
                System.out.println("Incorrect. The correct answer is: " + correctAnswer);
            }

            System.out.print("Do you want to play again? (yes/no): ");
            input.nextLine();
            String again = input.nextLine();
            playAgain = again.equalsIgnoreCase("yes");
        }

        System.out.println("Thank you for playing Math Quiz Game!");
        input.close();
    }
}
```

OUTPUT:

```
run:
Welcome to Math Quiz Game!

Choose Difficulty Level:
1. Easy
2. Medium
3. Hard
Enter your choice: 1
5 * 9 = ?
45
Congratulations! Your answer is correct.
Do you want to play again? (yes/no): yes

Choose Difficulty Level:
1. Easy
2. Medium
3. Hard
Enter your choice: 2
37 - 32 = ?
5
Congratulations! Your answer is correct.
Do you want to play again? (yes/no): yes

Choose Difficulty Level:
1. Easy
2. Medium
3. Hard
Enter your choice: 3
9 / 9 = ?
1
Congratulations! Your answer is correct.
Do you want to play again? (yes/no): yes

Choose Difficulty Level:
1. Easy
2. Medium
3. Hard
Enter your choice: 3
61 * 130 = ?
7930
Congratulations! Your answer is correct.
Do you want to play again? (yes/no): no
Thank you for playing Math Quiz Game!
BUILD SUCCESSFUL (total time: 1 minute 53 seconds)
```

LAB # 11

Construct java programs using the concepts polymorphism and abstract classes

Objective

To study Method Overriding(polymorphism) and Abstract class in java.

LAB TASK

A. Create a class called computers and two classes MyComputer and YourComputer which inherits computer class. Define appropriate fields for the three classes. Create another class processor as a composite class of computer. Write a method which prints the differences between the processors of two computers.

INPUT:

```
1   class Processor {
2       String brand;
3       double speedGHz;
4
5       Processor(String brand, double speedGHz) {
6           this.brand = brand;
7           this.speedGHz = speedGHz;
8       }
9
10      void printSpecs() {
11          System.out.println("Brand: " + brand + ", Speed: " + speedGHz + "GHz");
12      }
13  }
14
    class Computer {
        String model;
        int ram;
18      Processor processor;
19
20      Computer(String model, int ram, Processor processor) {
21          this.model = model;
22          this.ram = ram;
23          this.processor = processor;
24      }
25  }
26
27  class MyComputer extends Computer {
28      MyComputer(String model, int ram, Processor processor) {
29          super(model, ram, processor);
30      }
31  }
```

```
33  class YourComputer extends Computer {
34      YourComputer(String model, int ram, Processor processor) {
35          super(model, ram, processor);
36      }
37  }
38
39  public class TestA {
40      static void compareProcessors(Computer c1, Computer c2) {
41          System.out.println("Comparing Processors:");
42          c1.processor.printSpecs();
43          c2.processor.printSpecs();
44          if (!c1.processor.brand.equals(c2.processor.brand)) {
45              System.out.println("Different Brands");
46          } else if (c1.processor.speedGHz != c2.processor.speedGHz) {
47              System.out.println("Different Speeds");
48          } else {
49              System.out.println("Same Processor Specs");
50          }
51      }
52
53      public static void main(String[] args) {
54          MyComputer myPC = new MyComputer("Dell", 16, new Processor("Intel", 3.5));
55          YourComputer yourPC = new YourComputer("HP", 8, new Processor("AMD", 2.8));
56          compareProcessors(myPC, yourPC);
57      }
58  }
59
```

OUTPUT:

```
run:
Comparing Processors:
Brand: Intel, Speed: 3.5GHz
Brand: AMD, Speed: 2.8GHz
Different Brands
BUILD SUCCESSFUL (total time: 0 seconds)
```

B. Create a hierarchy of Liquid types. The parent class is Liquid, and its children classes are Coffee and Milk. A Liquid object will have an Add() method that takes some amount of liquid from user and adds it to the original amount of liquid, a remove () method that remove some amount of liquid, and a removeAll() method that removes all amount of liquid. Each subtype of Liquid overrides the three methods of liquid class. Coffee class have some specialize behavior that is it has a Swirl() method that prints "Swirling coffee" Create a test class that create an array of Liquid class having ten elements and populate it randomly with objects (all classes) and Call there method to show polymorphic behavior. Note: Each of the methods in the hierarchy classes has to be called in the Test class. Using for loop and use the InstanceOf() method to show Coffee class specialized Behavior.

INPUT:

```
1  import java.util.Random;
2
   abstract class Liquid {
       int amount;
5
6      Liquid(int amount) {
7          this.amount = amount;
8      }
9
       abstract void add(int amt);
       abstract void remove(int amt);
       abstract void removeAll();
13 }
14
15 class Coffee extends Liquid {
16     Coffee(int amount) {
17         super(amount);
18     }
19
20     @Override
       void add(int amt) {
22         amount += amt;
23         System.out.println("Coffee added: " + amt + "ml");
24     }
25
26     @Override
       void remove(int amt) {
28         amount -= amt;
29         System.out.println("Coffee removed: " + amt + "ml");
30     }
```

```
31
32          @Override
33          void removeAll() {
34              amount = 0;
35              System.out.println("All coffee removed.");
36          }
37
38          void swirl() {
39              System.out.println("Swirling coffee...");
40          }
41      }
42
43      class Milk extends Liquid {
44          Milk(int amount) {
45              super(amount);
46          }
47
48          @Override
49          void add(int amt) {
50              amount += amt;
51              System.out.println("Milk added: " + amt + "ml");
52          }
53
54          @Override
55          void remove(int amt) {
56              amount -= amt;
57              System.out.println("Milk removed: " + amt + "ml");
58          }
59
```

```
60          @Override
61          void removeAll() {
62              amount = 0;
63              System.out.println("All milk removed.");
64          }
65      }
66
67      public class TestB {
68          public static void main(String[] args) {
69              Liquid[] liquids = new Liquid[10];
70              Random rand = new Random();
71
72              for (int i = 0; i < liquids.length; i++) {
73                  if (rand.nextBoolean()) {
74                      liquids[i] = new Coffee(rand.nextInt(100) + 1);
75                  } else {
76                      liquids[i] = new Milk(rand.nextInt(100) + 1);
77                  }
78              }
79
80              for (Liquid l : liquids) {
81                  l.add(10);
82                  l.remove(5);
83                  l.removeAll();
84
85                  if (l instanceof Coffee) {
86                      ((Coffee) l).swirl();
87                  }
88                  System.out.println("-----");
89              }
90          }
91      }
92
```

OUTPUT:

```
run:
Coffee added: 10ml
Coffee removed: 5ml
All coffee removed.
Swirling coffee...
-----
Milk added: 10ml
Milk removed: 5ml
All milk removed.
-----
```

C. Create a class purse, which contains hidden data, rupees and paisas. The class contains 2 constructors, one with zero argument that set the data to Zero, and others set the data provided by the arguments. Instance method that reset data to Zero should be provided. Method for conversion of rupees into paisas should be provided. Also it must contain addmoney, deletemoney methods.

INPUT:

```
1   class Purse {
2       private int rupees;
3       private int paisas;
4
6       Purse() {
7           rupees = 0;
8           paisas = 0;
9       }
10
11      Purse(int rupees, int paisas) {
12          this.rupees = rupees;
13          this.paisas = paisas;
14      }
15
16      void reset() {
17          rupees = 0;
18          paisas = 0;
19      }
20
21      int toPaisas() {
22          return rupees * 100 + paisas;
23      }
24
25      void addMoney(int r, int p) {
26          int total = this.toPaisas() + (r * 100 + p);
27          rupees = total / 100;
28          paisas = total % 100;
29      }
```

```
30      void deleteMoney(int r, int p) {
31          int total = this.toPaisas() - (r * 100 + p);
32          if (total < 0) {
33              rupees = 0;
34              paisas = 0;
35          } else {
36              rupees = total / 100;
37              paisas = total % 100;
38          }
39      }
40
41      void show() {
42          System.out.println("Purse: Rs " + rupees + " and " + paisas + " paisas");
43      }
44  }
45
```

```
46  public class TestC {
47      public static void main(String[] args) {
48          Purse p = new Purse(10, 50);
49          p.show();
50
51          p.addMoney(5, 75);
52          p.show();
53
54          p.deleteMoney(8, 30);
55          p.show();
56
57          p.reset();
58          p.show();
59      }
60  }
61
```

OUTPUT:

```
run:
Purse: Rs 10 and 50 paisas
Purse: Rs 16 and 25 paisas
Purse: Rs 7 and 95 paisas
Purse: Rs 0 and 0 paisas
BUILD SUCCESSFUL (total time: 0 seconds)
```

D. You are given an abstract class Shape that defines a blueprint for shapes with an abstract method calculateArea(). Your task is to:

- **Create two subclasses, Circle and Rectangle, that extend the Shape class.**
- **Implement the calculateArea() method in both subclasses.**
- **Create instances of Circle and Rectangle in a Main class and display their areas.**

INPUT:

```
abstract class Shape {
    abstract double calculateArea();
}

class Circle extends Shape {
    double radius;

    Circle(double radius) {
        this.radius = radius;
    }

    @Override
    double calculateArea() {
        return Math.PI * radius * radius;
    }
}

class Rectangle extends Shape {
    double length, width;

    Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    @Override
    double calculateArea() {
        return length * width;
    }
}
public class TestD {
    public static void main(String[] args) {
        Shape s1 = new Circle(5);
        Shape s2 = new Rectangle(4, 6);

        System.out.println("Area of Circle: " + s1.calculateArea());
        System.out.println("Area of Rectangle: " + s2.calculateArea());
    }
}
```

OUTPUT:

```
run:
Area of Circle: 78.53981633974483
Area of Rectangle: 24.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 12

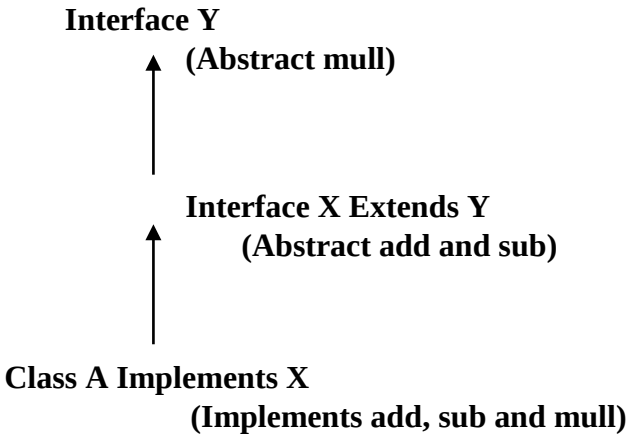
INTERFACE AND FINAL KEYWORD

OBJECTIVE:

To study Interface and final keyword.

LAB TASK

A. Implement below hierarchal diagram in java using interfaces.



INPUT:

```
1   package lab12;
2
    interface Y {
        void mull(int a, int b);
5   }
6
    interface X extends Y {
        void add(int a, int b);
        void sub(int a, int b);
10  }
11
12  class A implements X {
        public void add(int a, int b) {
14          System.out.println("Addition: " + (a + b));
15      }
16
        public void sub(int a, int b) {
18          System.out.println("Subtraction: " + (a - b));
19      }
20
        public void mull(int a, int b) {
22          System.out.println("Multiplication: " + (a * b));
23      }
24  }
25
```

```
26    public class InterfaceTest {
27        public static void main(String[] args) {
28            A obj = new A();
29            obj.add(10, 5);
30            obj.sub(10, 5);
31            obj.mull(10, 5);
32        }
33    }
```

OUTPUT:

```
run:
Addition: 15
Subtraction: 5
Multiplication: 50
BUILD SUCCESSFUL (total time: 0 seconds)
```

B. Create an Interface Shape with three child classes Circle, Square and Rectangle. Create a method Draw in each class. Create an Object of each class that can call the method Draw and it will call automatically the Draw method of each class. The text of Draw method e.g. Circle is "This is a Circle".

INPUT:

```
1     interface Shape {
2         void draw();
3     }
4
5     class Circle implements Shape {
7         public void draw() {
8             System.out.println("This is a Circle");
9         }
10    }
11
11    class Square implements Shape {
13        public void draw() {
14            System.out.println("This is a Square");
15        }
16    }
17
17    class Rectangle implements Shape {
19        public void draw() {
20            System.out.println("This is a Rectangle");
21        }
22    }
```

```
23    public class ShapeTest {
24        public static void main(String[] args) {
25            Shape circle = new Circle();
26            Shape square = new Square();
27            Shape rectangle = new Rectangle();
28
29            circle.draw();
30            square.draw();
31            rectangle.draw();
32        }
33    }
```

OUTPUT:

```
run:
This is a Circle
This is a Square
This is a Rectangle
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 13

PACKAGES

OBJECTIVE:

An apparent detour to understand what is package is in Java.

LAB TASK

A. Create a package name **bank_account**, containing two classes named as **Calculate_interest** for customers and **Calculate_special_Interest** for employees respectively. Each class should carry a method of calculating the amount.

Formulae:

interestAmount = amount * rate/100; special_interestAmount (amount * rate/100) + 100;

INPUT:

```
1  package bank_account;
2
3  public class Calculate_interest {
4      public double calculate(double amount, double rate) {
5          // Formula: interestAmount = amount * rate / 100
6          return amount * rate / 100;
7      }
8  }
```

```
1  package bank_account;
2
3  public class Calculate_special_Interest {
4      public double calculate(double amount, double rate) {
5          // Formula: special_interestAmount = (amount * rate / 100) + 100
6          return (amount * rate / 100) + 100;
7      }
8  }
```

B. Import the bank_account package in to a class name **Calculation_record** and display the calculated amount of interest given to the customer and the employees respectively.

INPUT:

```
import bank_account.Calculate_interest;
import bank_account.Calculate_special_Interest;

public class Calculation_record {
    public static void main(String[] args) {
        // Example values
        double amount = 1000;
        double rate = 5;

        // Customer interest
        Calculate_interest customer = new Calculate_interest();
        double interest = customer.calculate(amount, rate);
        System.out.println("Customer Interest Amount: " + interest);

        // Employee special interest
        Calculate_special_Interest employee = new Calculate_special_Interest();
        double specialInterest = employee.calculate(amount, rate);
        System.out.println("Employee Special Interest Amount: " + specialInterest);
    }
}
```

OUTPUT:

```
run:
Customer Interest Amount: 50.0
Employee Special Interest Amount: 150.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 14 a

Demonstrate the concept of Exception Handling & create your own exception.

Objective:

To demonstrate the concept of Exception Handling & create your own exception.

LAB TASK

A. What will be the output?

INPUT:

```
1   package lab14a;
2   public class Example1 {
3       public static void main(String[] args) {
4           try {
5               int[] numbers = {1, 2, 3};
6               System.out.println(numbers[4]);
7           } catch (ArrayIndexOutOfBoundsException e) {
8               System.out.println("Array index out of bounds.");
9           } finally {
10              System.out.println("Finally block executed.");
11          }
12      }
13  }
```

OUTPUT:

```
run:
Array index out of bounds.
Finally block executed.
BUILD SUCCESSFUL (total time: 0 seconds)
```

B. Create a Java program to handle ArithmeticException when dividing by zero using try, catch, and finally blocks.

INPUT:

```
1   public class ArithmeticExample {
2       public static void main(String[] args) {
3           try {
4               int a = 10;
5               int b = 0;
6               int c = a / b;
7               System.out.println("Result: " + c);
8           } catch (ArithmeticException e) {
9               System.out.println("Cannot divide by zero.");
10          } finally {
11              System.out.println("Execution complete.");
12          }
13      }
14  }
15
```

OUTPUT:

```
run:
Cannot divide by zero.
Execution complete.
BUILD SUCCESSFUL (total time: 0 seconds)
```

C. Write a program in which you have to take amount and if amount is less than 10000 so exception occurs with some greeting message i.e. "You are unable to get the requested withdraw amount" using throw Keyword else "Amount Successfully withdraws".

INPUT:

```
1   public class ThrowExample {
2    public static void main(String[] args) {
3        int amount = 5000;
4
5        try {
6            if (amount < 10000) {
7                throw new Exception("You are unable to get the requested withdraw amount");
8            } else {
9                System.out.println("Amount Successfully withdraws.");
10           }
11       } catch (Exception e) {
12           System.out.println(e.getMessage());
13       }
14   }
15  }
16
```

OUTPUT:

```
run:
You are unable to get the requested withdraw amount
BUILD SUCCESSFUL (total time: 0 seconds)
```

D. Write a program in which you have to take amount and if amount is less than 10000 so exception occurs with some greeting message i.e. "You are unable to get the requested withdraw amount" using throws Keyword else "Amount Successfully withdraws".

INPUT:

```
1  public class ThrowsExample {
2
3      public static void checkAmount(int amount) throws Exception {
4          if (amount < 10000) {
5              throw new Exception("You are unable to get the requested withdraw amount");
6          } else {
7              System.out.println("Amount Successfully withdraws.");
8          }
9      }
10
11     public static void main(String[] args) {
12         try {
13             checkAmount(5000);
14         } catch (Exception e) {
15             System.out.println(e.getMessage());
16         }
17     }
18 }
```

OUTPUT:

```
run:
You are unable to get the requested withdraw amount
BUILD SUCCESSFUL (total time: 0 seconds)
```

E. Add an exception class that calls a method to throw an exception to invigilate customer's age in the cinema hall. When the user enters age less than 18 so an exception should be created displaying not eligibility message in the output.

INPUT:

```
class AgeException extends Exception {
public AgeException(String message) {
    super(message);
}
}

public class Cinema {
public static void checkAge(int age) throws AgeException {
    if (age < 18) {
        throw new AgeException("You are not eligible to enter the cinema hall.");
    } else {
        System.out.println("Welcome to the cinema hall!");
    }
}

public static void main(String[] args) {
    try {
        checkAge(16);
    } catch (AgeException e) {
        System.out.println(e.getMessage());
    }
}
}
```

OUTPUT:

```
run:
You are not eligible to enter the cinema hall.
BUILD SUCCESSFUL (total time: 0 seconds)
```

F. Drive two Classes. One is of exception class and another is of using the exception class, prompt to deposit the money ranging from 1000 to 50,000 and display "money has been deposit." else generate an "error." by calling the exception in both the cases.

(Hint: try-catch)

INPUT:

```
class DepositException extends Exception {
 public DepositException(String message) {
     super(message);
 }
}

public class Bank {
 public static void depositMoney(int amount) throws DepositException {
     if (amount < 1000 || amount > 50000) {
         throw new DepositException("Invalid amount. Must be between 1000 and 50000.");
     } else {
         System.out.println("Money has been deposited.");
     }
 }

 public static void main(String[] args) {
     try {
         depositMoney(60000); // change this to 2000 for valid test
     } catch (DepositException e) {
         System.out.println("Error: " + e.getMessage());
     }
 }
}

```

OUTPUT:

```
run:
Error: Invalid amount. Must be between 1000 and 50000.
BUILD SUCCESSFUL (total time: 0 seconds)
```

LAB # 14 b

Open Ended Lab

Objective

Implementation of open ended lab.

LAB TASK

- ❖ Write a program by carefully reading the below given scenario;

1. Create a custom exception class named **`AgeOutOfRangeException`** that extends **`Exception`**.
   - This exception should have a constructor that accepts a message parameter.
   - Override the **`toString()`** method to return a formatted string with the exception message.

INPUT:

```
package lab.pkg14b;
public class AgeOutOfRangeException extends Exception {
    public AgeOutOfRangeException(String message) {
        super(message);
    }

    @Override
    public String toString() {
        return "AgeOutOfRangeException: " + getMessage();
    }
}
```

2. Write a class named `Voter` with a method `registerVoter(String name, int age)`.
   - Inside **`registerVoter()`**, throw an **`AgeOutOfRangeException`** if the age is less than 18 or greater than 120.
   - Use the **`throws`** keyword in the method signature to declare that it throws **`AgeOutOfRangeException`**.

INPUT:

```
import lab.pkg14b.AgeOutOfRangeException;

public class Voter {
    public void registerVoter(String name, int age) throws AgeOutOfRangeException {
        if (age < 18 || age > 120) {
            throw new AgeOutOfRangeException("Age must be between 18 and 120 to register as a voter.");
        } else {
            System.out.println("Voter registered successfully: " + name + " (" + age + " years old)");
        }
    }
}

```

3. In the main method of another class (`Main`), demonstrate the usage of **`registerVoter()`:**
   - Prompt the user to enter their name and age.
   - Call **`registerVoter()`** method with the provided inputs and handle any thrown exceptions using a try-catch block.
   - Print appropriate messages based on whether the voter registration was successful or if an exception was caught.

INPUT:

```
import java.util.Scanner;
import lab.pkg14b.AgeOutOfRangeException;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        Voter voter = new Voter();

        System.out.print("Enter your name: ");
        String name = input.nextLine();

        System.out.print("Enter your age: ");
        int age = input.nextInt();

        try {
            voter.registerVoter(name, age);
        } catch (AgeOutOfRangeException e) {
            System.out.println("Registration failed.");
            System.out.println(e);
        }

        input.close();
    }
}
```

OUTPUT:

```
run:
Enter your name: Muhammad Asad Khan
Enter your age: 20
Voter registered successfully: Muhammad Asad Khan (20 years old)
BUILD SUCCESSFUL (total time: 13 seconds)
```

```
run:
Enter your name: Muhammad Umar
Enter your age: 17
Registration failed.
AgeOutOfRangeException: Age must be between 18 and 120 to register as a voter.
BUILD SUCCESSFUL (total time: 15 seconds)
```