1. Write a program in Java with an algorithm to display a student mark sheet with total marks, percentage, and grade.

Welcome | import java.util.Scanner; Untitled-1

```java
import java.util.Scanner;

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

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

        System.out.print("Enter Roll Number: ");
        int rollNumber = scanner.nextInt();

        int subjects = 5;
        int[] marks = new int[subjects];
        int totalMarks = 0;
        int maxMarksPerSubject = 100;
        int maxTotalMarks = subjects * maxMarksPerSubject;

        // Input marks
        for (int i = 0; i < subjects; i++) {
            System.out.print("Enter marks for Subject " + (i + 1) + ": ");
            marks[i] = scanner.nextInt();
            totalMarks += marks[i];
        }

        // Calculate percentage
        double percentage = (double) totalMarks / maxTotalMarks * 100;

        // Assign grade
        String grade;
        if (percentage >= 90) {
            grade = "A+";
        } else if (percentage >= 80) {
            grade = "A";
        } else if (percentage >= 70) {
            grade = "B";
        } else if (percentage >= 60) {
            grade = "C";
        } else if (percentage >= 50) {
            grade = "D";
        } else {
            grade = "Fail";
        }

        // Display Mark Sheet
        System.out.println("\n===== Student Mark Sheet =====");
        System.out.println("Student Name: " + name);
        System.out.println("Roll Number: " + rollNumber);
        System.out.println("Total Marks: " + totalMarks + " / " + maxTotalMarks);
        System.out.println("Percentage: " + String.format("%.2f", percentage) + "%");
        System.out.println("Grade: " + grade);
        System.out.println("============================");

        scanner.close();
    }
}
```

Output:

```
PS C:\Users\Akshad> javac StudentMarkSheet.java
PS C:\Users\Akshad> java StudentMarkSheet
Enter Student Name: John Doe
Enter Roll Number: 101
Enter marks for Subject 1: 85
Enter marks for Subject 2: 78
Enter marks for Subject 3: 90
Enter marks for Subject 4: 88
Enter marks for Subject 5: 76

===== Student Mark Sheet =====
Student Name: John Doe
Roll Number: 101
Total Marks: 417 / 500
Percentage: 83.40%
Grade: A
============================
PS C:\Users\Akshad>
```

2. Write a program in Java with an algorithm to print given pattern:

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

Welcome | import java.util.Scanner; Untitled-1

```java
import java.util.Scanner;

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

        System.out.print("Enter the number of rows for the upper half: ");
        int n = scanner.nextInt(); // Example: 4 (for 7-star width)

        // Upper half pyramid
        for (int i = 1; i <= n; i += 2) {
            printSpaces(n - i);
            printStars(i);
        }

        // Lower half inverted pyramid
        for (int i = n - 2; i >= 1; i -= 2) {
            printSpaces(n - i);
            printStars(i);
        }

        scanner.close();
    }

    // Method to print spaces
    private static void printSpaces(int count) {
        for (int i = 0; i < count; i++) {
            System.out.print(" ");
        }
    }

    // Method to print stars
    private static void printStars(int count) {
        for (int i = 0; i < count; i++) {
            System.out.print("*");
        }
        System.out.println();
    }
}
```

Output:

```
PS C:\Users\Akshad> javac StarPattern.java
PS C:\Users\Akshad> java StarPattern
Enter the number of rows for the upper half: 4
   *
  ***
 *****
*******
 *****
  ***
   *
PS C:\Users\Akshad>
```

3. Write a program in Java with an algorithm to print transpose of a matrix.

```java
import java.util.Scanner;

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

        // Input rows and columns
        System.out.print("Enter the number of rows: ");
        int rows = scanner.nextInt();
        System.out.print("Enter the number of columns: ");
        int cols = scanner.nextInt();

        // Declare matrix
        int[][] matrix = new int[rows][cols];
        int[][] transpose = new int[cols][rows]; // Transposed matrix size

        // Input matrix elements
        System.out.println("Enter matrix elements:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Compute transpose
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                transpose[j][i] = matrix[i][j];
            }
        }

        // Print original matrix
        System.out.println("\nOriginal Matrix:");
        printMatrix(matrix, rows, cols);

        // Print transposed matrix
        System.out.println("\nTranspose of the Matrix:");
        printMatrix(transpose, cols, rows);

        scanner.close();
    }

    // Method to print a matrix
    private static void printMatrix(int[][] matrix, int rows, int cols) {
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Output:

```
PS C:\Users\Akshad> javac MatrixTranspose.java
PS C:\Users\Akshad> java MatrixTranspose
Enter the number of rows: 2
Enter the number of columns: 3
Enter matrix elements:
1 2 3
4 5 6

Original Matrix:
1 2 3
4 5 6

Transpose of the Matrix:
1 4
2 5
3 6
PS C:\Users\Akshad>
```

4. Write a program in Java with an algorithm to implement parameterized constructor with two parameters id and name. While creating the objects obj1 and obj2 passed two arguments so that this constructor gets invoked after creation of obj1 and obj2.

```java
// Class with a parameterized constructor
class Student {
    int id;
    String name;

    // Parameterized constructor
    Student(int id, String name) {
        this.id = id;
        this.name = name;
    }

    // Method to display student details
    void display() {
        System.out.println("Student ID: " + id + ", Name: " + name);
    }
}

// Main class
public class ParameterizedConstructorExample {
    public static void main(String[] args) {
        // Creating objects with constructor arguments
        Student obj1 = new Student(101, "Akshad");
        Student obj2 = new Student(102, "Shrushti");

        // Displaying student details
        obj1.display();
        obj2.display();
    }
}
```

Output:

```
PS C:\Users\Akshad> javac ParameterizedConstructorExample.java
PS C:\Users\Akshad> java ParameterizedConstructorExample
Student ID: 101, Name: Akshad
Student ID: 102, Name: Shrushti
PS C:\Users\Akshad>
```

5. Write a Java program with an algorithm to create a class called Vehicle with a method called drive (). Create a subclass called Car that overrides the drive () method to print "Repairing a car".

```java
// Base class
class Vehicle {
    // Method to be overridden
    void drive() {
        System.out.println("Driving a vehicle");
    }
}

// Subclass overriding the drive method
class Car extends Vehicle {
    @Override
    void drive() {
        System.out.println("Repairing a car");
    }
}

// Main class
public class MethodOverridingExample {
    public static void main(String[] args) {
        Vehicle myVehicle = new Vehicle(); // Creating Vehicle object
        myVehicle.drive(); // Calls Vehicle's drive() method

        Car myCar = new Car(); // Creating Car object
        myCar.drive(); // Calls Car's overridden drive() method
    }
}
```

Output:

```
PS C:\Users\Akshad> javac MethodOverridingExample.java
PS C:\Users\Akshad> java MethodOverridingExample
Driving a vehicle
Repairing a car
PS C:\Users\Akshad>
```

6. Write a Java program to create an interface Shape with the getArea() method. Create three classes Rectangle, Circle, and Triangle that implement the Shape interface. Implement the getArea() method for each of the three classes.

Welcome | // Shape interface Untitled-1

```java
// Rectangle class implementing Shape
class Rectangle implements Shape {
    private double length, width;

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

    // Implement getArea() for Rectangle
    @Override
    public double getArea() {
        return length * width;
    }
}

// Circle class implementing Shape
class Circle implements Shape {
    private double radius;
    private static final double PI = 3.14159; // Constant value for Pi

    // Constructor
    public Circle(double radius) {
        this.radius = radius;
    }

    // Implement getArea() for Circle
    @Override
    public double getArea() {
        return PI * radius * radius;
    }
}

// Triangle class implementing Shape
class Triangle implements Shape {
    private double base, height;

    // Constructor
    public Triangle(double base, double height) {
        this.base = base;
        this.height = height;
    }

    // Implement getArea() for Triangle
    @Override
    public double getArea() {
        return 0.5 * base * height;
    }
}

// Main class
public class ShapeInterfaceExample {
    public static void main(String[] args) {
        // Create objects of different shapes
        Shape rectangle = new Rectangle(10, 5);
        Shape circle = new Circle(7);
        Shape triangle = new Triangle(6, 4);

        // Display areas
        System.out.println("Rectangle Area: " + rectangle.getArea());
        System.out.println("Circle Area: " + circle.getArea());
        System.out.println("Triangle Area: " + triangle.getArea());
    }
}
```

Output:

```
PS C:\Users\Akshad> javac ShapeInterfaceExample.java
PS C:\Users\Akshad> java ShapeInterfaceExample
Rectangle Area: 50.0
Circle Area: 153.93791
Triangle Area: 12.0
PS C:\Users\Akshad>
```

7. Write a java program with an algorithm that implements a multi-thread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.

Welcome | import java.util.Random; Untitled-1

```java
import java.util.Random;

// Thread 1: Generates a random number every 1 second
class RandomNumberGenerator extends Thread {
    public void run() {
        Random rand = new Random();
        while (true) {
            int num = rand.nextInt(100); // Generate a random number between 0-99
            System.out.println("\nGenerated Number: " + num);

            // Check if even or odd and create corresponding thread
            if (num % 2 == 0) {
                new SquareThread(num).start();
            } else {
                new CubeThread(num).start();
            }

            try {
                Thread.sleep(1000); // Pause for 1 second
            } catch (InterruptedException e) {
                System.out.println(e);
            }
        }
    }
}

// Thread 2: Computes square if number is even
class SquareThread extends Thread {
    int number;

    SquareThread(int num) {
        this.number = num;
    }

    public void run() {
        System.out.println("Square of " + number + " is: " + (number * number));
    }
}

// Thread 3: Computes cube if number is odd
class CubeThread extends Thread {
    int number;

    CubeThread(int num) {
        this.number = num;
    }

    public void run() {
        System.out.println("Cube of " + number + " is: " + (number * number * number));
    }
}

// Main Class
public class MultiThreadExample {
    public static void main(String[] args) {
        RandomNumberGenerator thread1 = new RandomNumberGenerator();
        thread1.start(); // Start the first thread
    }
}
```

Output:

```
PS C:\Users\Akshad> javac MultiThreadExample.java
PS C:\Users\Akshad> java MultiThreadExample

Generated Number: 12
Square of 12 is: 144

Generated Number: 37
Cube of 37 is: 50653

Generated Number: 28
Square of 28 is: 784

Generated Number: 15
Cube of 15 is: 3375

Generated Number: 42
Square of 42 is: 1764
```

8. Write a Java program with an algorithm to create a class called Bank Account with private instance variables account Number and balance. Provide public getter and setter methods to access and modify these variables.

Welcome // BankAccount class Untitled-1

```java
// BankAccount class
class BankAccount {
    // Private instance variables (Encapsulation)
    private String accountNumber;
    private double balance;

    // Constructor to initialize account
    public BankAccount(String accountNumber, double initialBalance) {
        this.accountNumber = accountNumber;
        this.balance = initialBalance;
    }

    // Getter for accountNumber
    public String getAccountNumber() {
        return accountNumber;
    }

    // Setter for accountNumber
    public void setAccountNumber(String accountNumber) {
        this.accountNumber = accountNumber;
    }

    // Getter for balance
    public double getBalance() {
        return balance;
    }

    // Setter for balance with validation
    public void setBalance(double balance) {
        if (balance >= 0) {
            this.balance = balance;
        } else {
            System.out.println("Error: Balance cannot be negative!");
        }
    }

    // Method to deposit money
    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: $" + amount);
        } else {
            System.out.println("Error: Deposit amount must be positive!");
        }
    }

    // Method to withdraw money
    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawn: $" + amount);
        } else {
            System.out.println("Error: Insufficient balance or invalid amount!");
        }
    }

```

```java
    }

    // Method to display account details
    public void displayAccount() {
        System.out.println("\nAccount Number: " + accountNumber);
        System.out.println("Current Balance: $" + balance);
    }
}

// Main class
public class BankSystem {
    public static void main(String[] args) {
        // Creating a bank account object
        BankAccount myAccount = new BankAccount("123456789", 5000);

        // Display initial details
        myAccount.displayAccount();

        // Deposit money
        myAccount.deposit(2000);

        // Withdraw money
        myAccount.withdraw(1500);

        // Display updated details
        myAccount.displayAccount();
    }
}
```

Output:

```
PS C:\Users\Akshad> javac BankSystem.java
PS C:\Users\Akshad> java BankSystem

Account Number: 123456789
Current Balance: $5000.0

Deposited: $2000.0
Withdrawn: $1500.0

Account Number: 123456789
Current Balance: $5500.0
```

9. Write a Java program with an algorithm to create a method that takes an integer as a parameter and throws an exception if the number is odd.

```java
// Main class
public class OddNumberExceptionDemo {
    // Method to check if number is even
    public static void checkEvenNumber(int num) {
        if (num % 2 != 0) {
            throw new RuntimeException("Error: The number " + num + " is odd!");
        } else {
            System.out.println("The number " + num + " is even. No exception thrown.");
        }
    }

    // Main method
    public static void main(String[] args) {
        try {
            checkEvenNumber(10);  // Even number, should not throw exception
            checkEvenNumber(7);   // Odd number, should throw exception
        } catch (RuntimeException e) {
            System.out.println("Exception Caught: " + e.getMessage());
        }
    }
}
```

Output:

```
PS C:\Users\Akshad> javac OddNumberExceptionDemo.java
PS C:\Users\Akshad> java OddNumberExceptionDemo

The number 10 is even. No exception thrown.
Exception Caught: Error: The number 7 is odd!
```

10. Write a Java program with an algorithm to insert the specified element at the specified position in the linked list.

```java
import java.util.LinkedList;

public class LinkedListInsertion {
    // Method to insert element at a specified position
    public static void insertAtPosition(LinkedList<Integer> list, int element, int position) {
        if (position < 0 || position > list.size()) {
            System.out.println("Error: Invalid position! Position must be between 0 and " + list.size());
            return;
        }
        list.add(position, element);
        System.out.println("Inserted " + element + " at position " + position);
    }

    public static void main(String[] args) {
        // Creating a LinkedList
        LinkedList<Integer> numbers = new LinkedList<>();
        numbers.add(10);
        numbers.add(20);
        numbers.add(30);
        numbers.add(40);

        // Display original LinkedList
        System.out.println("Original LinkedList: " + numbers);

        // Insert element at a specified position
        insertAtPosition(numbers, 25, 2);

        // Display updated LinkedList
        System.out.println("Updated LinkedList: " + numbers);
    }
}
```

Output:

```
PS C:\Users\Akshad> javac LinkedListInsertion.java
PS C:\Users\Akshad> java LinkedListInsertion

Original LinkedList: [10, 20, 30, 40]
Inserted 25 at position 2
Updated LinkedList: [10, 20, 25, 30, 40]
```

11. Write a Java program with an algorithm to implement: a) Hash Map b)Tree Map

```java
import java.util.*;

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

        // Creating HashMap and TreeMap
        HashMap<Integer, String> hashMap = new HashMap<>();
        TreeMap<Integer, String> treeMap = new TreeMap<>();

        // Taking user input for map entries
        System.out.println("Enter number of elements:");
        int n = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        for (int i = 0; i < n; i++) {
            System.out.println("Enter key (integer) and value (string) separated by space:");
            int key = scanner.nextInt();
            String value = scanner.next();
            hashMap.put(key, value);
            treeMap.put(key, value);
        }

        scanner.close();

        // Display HashMap (unordered)
        System.out.println("\nHashMap (Unordered):");
        for (Map.Entry<Integer, String> entry : hashMap.entrySet()) {
            System.out.println("Key: " + entry.getKey() + ", Value: " + entry.getValue());
        }

        System.out.println("\n-----------------------\n");

        // Display TreeMap (Sorted by Key)
        System.out.println("TreeMap (Sorted by Key):");
        for (Map.Entry<Integer, String> entry : treeMap.entrySet()) {
            System.out.println("Key: " + entry.getKey() + ", Value: " + entry.getValue());
        }
    }
}
```

Output:

```
PS C:\Users\Akshad> javac MapUserInputExample.java
PS C:\Users\Akshad> java MapUserInputExample

Enter number of elements:
4
Enter key (integer) and value (string) separated by space:
3 Apple
Enter key (integer) and value (string) separated by space:
1 Mango
Enter key (integer) and value (string) separated by space:
4 Banana
Enter key (integer) and value (string) separated by space:
2 Grapes

HashMap (Unordered):
Key: 3, Value: Apple
Key: 1, Value: Mango
Key: 4, Value: Banana
Key: 2, Value: Grapes

-----------------------

TreeMap (Sorted by Key):
Key: 1, Value: Mango
Key: 2, Value: Grapes
Key: 3, Value: Apple
Key: 4, Value: Banana
```

12.Write a Java program with an algorithm for Student Information Management System with Database Cnnectivity.

Welcome | import java.sql.*; Untitled-1

```java
import java.sql.*;
import java.util.Scanner;

public class StudentManagementSystem {
    // Database credentials (Dummy)
    static final String JDBC_URL = "jdbc:mysql://localhost:3306/StudentDB";
    static final String USER = "root"; // Replace with your MySQL username
    static final String PASSWORD = "password"; // Replace with your MySQL password

    public static void main(String[] args) {
        try (Connection conn = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
             Scanner scanner = new Scanner(System.in)) {

            System.out.println("Connected to Database!");
            int choice;
            do {
                System.out.println("\nStudent Information Management System");
                System.out.println("1. Add Student");
                System.out.println("2. View All Students");
                System.out.println("3. Update Student");
                System.out.println("4. Delete Student");
                System.out.println("5. Exit");
                System.out.print("Enter your choice: ");
                choice = scanner.nextInt();
                scanner.nextLine(); // Consume newline

                switch (choice) {
                    case 1:
                        addStudent(conn, scanner);
                        break;
                    case 2:
                        viewStudents(conn);
                        break;
                    case 3:
                        updateStudent(conn, scanner);
                        break;
                    case 4:
                        deleteStudent(conn, scanner);
                        break;
                    case 5:
                        System.out.println("Exiting program...");
                        break;
                    default:
                        System.out.println("Invalid choice! Try again.");
                }
            } while (choice != 5);

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    // Method to insert a student record
    private static void addStudent(Connection conn, Scanner scanner) throws SQLException {
        System.out.print("Enter student name: ");
        String name = scanner.nextLine();
        System.out.print("Enter student age: ");
        int age = scanner.nextInt();
        scanner.nextLine(); // Consume newline
        System.out.print("Enter student course: ");
        String course = scanner.nextLine();

        String query = "INSERT INTO students (name, age, course) VALUES (?, ?, ?)";
        try (PreparedStatement stmt = conn.prepareStatement(query)) {
            stmt.setString(1, name);
            stmt.setInt(2, age);
            stmt.setString(3, course);
            stmt.executeUpdate();
```

Welcome | import java.sql.*; Untitled-1

```java
public class StudentManagementSystem {
    private static void addStudent(Connection conn, Scanner scanner) throws SQLException {
        try (PreparedStatement stmt = conn.prepareStatement(query)) {
            System.out.println("Student added successfully!");
        }
    }

    // Method to display all students
    private static void viewStudents(Connection conn) throws SQLException {
        String query = "SELECT * FROM students";
        try (Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {
            System.out.println("\nID | Name | Age | Course");
            System.out.println("----------------------------");
            while (rs.next()) {
                System.out.printf("%d | %s | %d | %s\n", rs.getInt("id"), rs.getString("name"),
                        rs.getInt("age"), rs.getString("course"));
            }
        }
    }

    // Method to update student information
    private static void updateStudent(Connection conn, Scanner scanner) throws SQLException {
        System.out.print("Enter student ID to update: ");
        int id = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        System.out.print("Enter new student name: ");
        String name = scanner.nextLine();
        System.out.print("Enter new age: ");
        int age = scanner.nextInt();
        scanner.nextLine(); // Consume newline
        System.out.print("Enter new course: ");
        String course = scanner.nextLine();

        String query = "UPDATE students SET name=?, age=?, course=? WHERE id=?";
        try (PreparedStatement stmt = conn.prepareStatement(query)) {
            stmt.setString(1, name);
            stmt.setInt(2, age);
            stmt.setString(3, course);
            stmt.setInt(4, id);
            int rowsUpdated = stmt.executeUpdate();
            if (rowsUpdated > 0) {
                System.out.println("Student updated successfully!");
            } else {
                System.out.println("Student ID not found!");
            }
        }
    }

    // Method to delete a student record
    private static void deleteStudent(Connection conn, Scanner scanner) throws SQLException {
        System.out.print("Enter student ID to delete: ");
        int id = scanner.nextInt();

        String query = "DELETE FROM students WHERE id=?";
        try (PreparedStatement stmt = conn.prepareStatement(query)) {
            stmt.setInt(1, id);
            int rowsDeleted = stmt.executeUpdate();
            if (rowsDeleted > 0) {
                System.out.println("Student deleted successfully!");
            } else {
                System.out.println("Student ID not found!");
            }
        }
    }
}
```

Output:

```
PS C:\Users\Akshad> javac StudentManagementSystem.java
PS C:\Users\Akshad> java StudentManagementSystem
Connected to Database!

Student Information Management System
1. Add Student
2. View All Students
3. Update Student
4. Delete Student
5. Exit
Enter your choice: 1
Enter student name: John
Enter student age: 20
Enter student course: B.Tech
Student added successfully!

Student Information Management System
1. Add Student
2. View All Students
3. Update Student
4. Delete Student
5. Exit
Enter your choice: 2

ID | Name | Age | Course
---------------------------
1 | Akshad | 21 | BCA
2 | Shrushti | 22 | MCA
3 | Rahul | 20 | B.Sc IT
4 | John | 20 | B.Tech
```