1. **Write a program to demonstrate encapsulation in Java.**

```
class Student {
   private String name;
   private int rollNo;
   public void setName(String n) {
      name = n;
   }
   public void setRollNo(int r) {
      rollNo = r;
   }
   public String getName() {
      return name;
   }
   public int getRollNo() {
      return rollNo;
   }
}
public class Main {
   public static void main(String[] args) {
      Student s = new Student();
      s.setName("Ravindra Jadeja");
      s.setRollNo(8);
      System.out.println("Name: " + s.getName());
      System.out.println("Roll No: " + s.getRollNo());
   }
}
```

2. Create a program showing the use of inheritance and polymorphism.

```
class Player {
   String name;
   Player(String name) {
      this.name = name;
   }
   void play() {
```

```
        System.out.println(name + " is playing cricket");
    }
}
class Batsman extends Player {
    Batsman(String name) {
        super(name);
    }
    void play() {
        System.out.println(name + " is batting");
    }
}
class Bowler extends Player {
    Bowler(String name) {
        super(name);
    }
    void play() {
        System.out.println(name + " is bowling");
    }
}
public class Main {
    public static void main(String[] args) {
        Player p;

        p = new Batsman("Ravindra Jadeja");
        p.play();

        p = new Bowler("Jasprit Bumrah");
        p.play();
    }
}
```

```
Ravindra Jadeja is batting
Jasprit Bumrah is bowling

=== Code Execution Successful ===
```

3. Explain and implement the concept of abstraction in Java using interfaces.

```
abstract class Student {
    abstract void studying();
    abstract void playing();
}
class SchoolStudent extends Student {
    void studying() {
        System.out.println("Student is studying.");
    }
    void playing() {
        System.out.println("Student is playing.");
    }
}
public class Main {
```

```
    public static void main(String[] args) {
        Student s = new SchoolStudent();
        s.studying();
        s.playing();
    }
}
```

Output Clear

```
Student is studying.
Student is playing.

=== Code Execution Successful ===
```

4. Write a program to demonstrate method overloading and method overriding.

```
class Student {
    void activity() {
        System.out.println("Student is studying in class.");
    }
}
class SportsStudent extends Student {
    void activity() {
        System.out.println("Sports student is playing football.");
    }
}
class Marks {
    void showResult(int rollNo, int marks) {
        System.out.println("Roll No: " + rollNo + ", Marks: " + marks);
    }
    void showResult(int rollNo, int m1, int m2, int m3) {
        int total = m1 + m2 + m3;
        System.out.println("Roll No: " + rollNo + ", Total Marks: " + total);
    }
}
public class Main {
    public static void main(String[] args) {
        Student s = new SportsStudent();
        s.activity();
        Marks result = new Marks();
        result.showResult(101, 85);
        result.showResult(102, 70, 80, 90);
    }
}
```

```
Sports student is playing football.
Roll No: 101, Marks: 85
Roll No: 102, Total Marks: 240

=== Code Execution Successful ===
```

5. Create a class hierarchy for animals that demonstrates polymorphism.

```
class Animal {
   void sound() {
      System.out.println("Animal makes a sound");
   }
}
class Dog extends Animal {
   void sound() {
      System.out.println("Dog barks");
   }
}
class Cat extends Animal {
   void sound() {
      System.out.println("Cat meows");
   }
}
class Cow extends Animal {
   void sound() {
      System.out.println("Cow moos");
   }
}
public class Main {
   public static void main(String[] args) {
      Animal a;
      a = new Dog();
      a.sound();
      a = new Cat();
      a.sound();
      a = new Cow();
      a.sound();
   }
}
```

```
Dog barks
Cat meows
Cow moos

=== Code Execution Successful ===
```

6. Develop a program to implement multiple inheritance using interfaces.

```
interface Walkable {
   void walk();
}

interface Swimmable {
   void swim();
}
class Duck implements Walkable, Swimmable {
   public void walk()
   {
```

```
        System.out.println("Duck is walking.");
    }
    public void swim()
    {
        System.out.println("Duck is swimming.");
    }
}
class Main {
    public static void main(String[] args)
    {
        Duck duck = new Duck();
        duck.walk();
        duck.swim();
    }
}
```

Output Clear

```
Duck is walking.
Duck is swimming.

=== Code Execution Successful ===
```

7.Write a Java program to showcase the use of this and super keywords.

```
class Person {
    String name;
    int age;

    Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    void display() {
        System.out.println("Name: " + name + ", Age: " + age);
    }
}
class Student extends Person {
    int rollNo;
    Student(String name, int age, int rollNo) {
        super(name, age);
        this.rollNo = rollNo;
    }
    void display() {
        super.display();
        System.out.println("Roll No: " + this.rollNo);
    }
}
public class Main {
```

```
    public static void main(String[] args) {
        Student s = new Student("Raghvi", 7, 777);
        s.display();
    }
}
```

```
Name: Raghvi, Age: 7
Roll No: 777

=== Code Execution Successful ===
```

8.Demonstrate the concept of constructors in OOP with a program.

```
import java.io.*;
class Geeks {
    Geeks()
    {
        super();
        System.out.println("Constructor Called");
    }
    public static void main(String[] args)
    {
        Geeks geek = new Geeks();
    }
}
```

Output

```
Constructor Called

=== Code Execution Successful ===
```

9.Explain and implement the concept of access modifiers in Java.

```
class Student {
    public String name;
    private int rollNo;
    protected int age;
    Student(String name, int rollNo, int age) {
        this.name = name;
        this.rollNo = rollNo;
        this.age = age;
    }
    public void display() {
        System.out.println("Name: " + name);
        System.out.println("Roll No: " + rollNo);
        System.out.println("Age: " + age);
    }
}
public class Main {
    public static void main(String[] args) {
```

```
        Student s = new Student("Raghvi", 101, 20);
        System.out.println("Name (public): " + s.name);

        s.display();
    }
}
```

```
Output                                                    Clear

Name (public): Raghvi
Name: Raghvi
Roll No: 101
Age: 20

=== Code Execution Successful ===
```

10.Show an example of the final keyword for variables, methods, and classes.

```
final class Shape {
    final double PI = 3.14159;

    final void displayPI() {
        System.out.println("Value of PI: " + PI);
    }
}
public class Main {
    public static void main(String[] args) {
        Shape s = new Shape();
        s.displayPI();

    }
}
```

```
Output

Value of PI: 3.14159

=== Code Execution Successful ===
```

11.Write a program that uses Java's StringBuilder for efficient string operations.

```
public class Main {
    public static void main(String[] args) {
        StringBuilder sb = new StringBuilder("Ravindrasinh Anirudhsinh Jadeja");
        System.out.println("Original: " + sb);
        sb.append(" - The All-Rounder");
        System.out.println("After append: " + sb);
        sb.insert(12, " (Sir)");
        System.out.println("After insert: " + sb);
        int start = sb.indexOf("Jadeja");
```

```
        sb.replace(start, start + 6, "Legend");
        System.out.println("After replace: " + sb);
        sb.reverse();
        System.out.println("After reverse: " + sb);
    }
}
```

Output Clear

```
Original: Ravindrasinh Anirudhsinh Jadeja
After append: Ravindrasinh Anirudhsinh Jadeja - The All-Rounder
After insert: Ravindrasinh (Sir) Anirudhsinh Jadeja - The All-Rounder
After replace: Ravindrasinh (Sir) Anirudhsinh Legend - The All-Rounder
After reverse: rednuoR-llA ehT - dnegeL hnishdurinA )riS( hnisardnivaR

=== Code Execution Successful ===
```

12.Write a program to demonstrate the immutability of the String class.

```
public class Mutable {
    public static void main(String[] s) {
        String a = "abc";
        String b = a;
        a = a.concat("d");
        System.out.println(a);
        System.out.println(b);

        StringBuffer c = new StringBuffer("abc");
        StringBuffer d = c;
        c.append("d");
        System.out.println(c);
        System.out.println(d);

    }
}
```

Output Clear

```
abcd
abc
abcd
abcd

=== Code Execution Successful ===
```

13.Write a program to declare variables of all primitive data types in Java and print their default values.

```
class Player {
   int runs;
   float average;
   boolean isCaptain;
   void display() {
      System.out.println("Stats of Ravindra Jadeja:");
      System.out.println("Runs: " + runs);
      System.out.println("Average: " + average);
      System.out.println("Is Captain: " + isCaptain);
   }
}
public class Main {
   public static void main(String[] args) {
      Player jadeja = new Player();
      jadeja.display();
   }
}
```

Output

```
Stats of Ravindra Jadeja:
Runs: 0
Average: 0.0
Is Captain: false
```

14.Implement a program to demonstrate the use of if-else, switch, and for loops.

```
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a number: ");
      int n = sc.nextInt();
      boolean isPrime = true;
      if (n <= 1) {
         isPrime = false;
      } else {
         for (int i = 2; i <= n / 2; i++) {
            if (n % i == 0) {
               isPrime = false;
               break;
            }
         }
      }
      if (isPrime)
         System.out.println(n + " is Prime");
      else
         System.out.println(n + " is Not Prime");
      switch (n % 2) {
         case 0:
```

```
            System.out.println(n + " is Even");
            break;
        default:
            System.out.println(n + " is Odd");
    }
    System.out.println("First 5 multiples of " + n + ":");
    for (int i = 1; i <= 5; i++) {
        System.out.print(n * i + " ");
    }
  }
}
```

Output Clear

```
Enter a number: 77
77 is Not Prime
77 is Odd
First 5 multiples of 77:
77 154 231 308 385
=== Code Execution Successful ===
```

15.Write a program to check if a number is prime using a while loop.

```
import java.util.Scanner;
public class Main {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter a number: ");
    int n = sc.nextInt();
    if (n <= 1) {
        System.out.println(n + " is Not Prime");
        return;
    }
    int i = 2;
    boolean isPrime = true;
    while (i <= n / 2) {
        if (n % i == 0) {
            isPrime = false;
            break;
        }
        i++;
    }
    if (isPrime)
        System.out.println(n + " is Prime");
    else
        System.out.println(n + " is Not Prime");
  }
}
```

Output Clear

```
Enter a number: 56
56 is Not Prime

=== Code Execution Successful ===
```

16.Create a program to calculate the factorial of a number using recursion.

```
import java.util.Scanner;
public class Main {
   static long factorial(int n) {
      if (n == 0 || n == 1)
         return 1;
      else
         return n * factorial(n - 1);
   }
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a number: ");
      int num = sc.nextInt();
      long result = factorial(num);
      System.out.println("Factorial of " + num + " = " + result);
   }
}
```

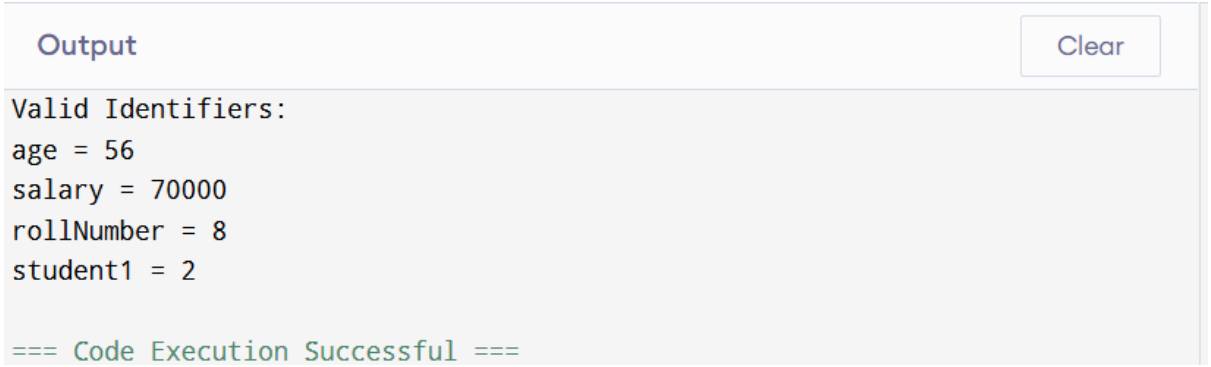
Output Clear

```
Enter a number: 56
56 is Not Prime

=== Code Execution Successful ===
```

17.Write a program to identify valid and invalid identifiers in Java.

```
public class Main {
   public static void main(String[] args) {
      int age = 56;
      int salary = 70000;
      int rollNumber = 8;
      int student1 = 2;
      System.out.println("Valid Identifiers:");
      System.out.println("age = " + age);
      System.out.println("salary = " + salary);
      System.out.println("rollNumber = " + rollNumber);
      System.out.println("student1 = " + student1);
   }
}
```



Arrays

18.Write a program to find the largest and smallest numbers in an array.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter number of elements: ");
      int n = sc.nextInt();
      int[] arr = new int[n];
      System.out.println("Enter " + n + " numbers:");
      for (int i = 0; i < n; i++) {
         arr[i] = sc.nextInt();
      }
      int largest = arr[0];
      int smallest = arr[0];
      for (int i = 1; i < n; i++) {
         if (arr[i] > largest)
            largest = arr[i];
         if (arr[i] < smallest)
            smallest = arr[i];
      }
      System.out.println("Largest number: " + largest);
      System.out.println("Smallest number: " + smallest);
   }
}
```

Output Clear

```
Enter number of elements: 7
Enter 7 numbers:
1
2
3
4
78
98
77
Largest number: 98
Smallest number: 1

=== Code Execution Successful ===
```

19.Write a program to check if a given number is odd or even.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a number: ");
      int n = sc.nextInt();
      if (n % 2 == 0) {
         System.out.println(n + " is Even");
      } else {
         System.out.println(n + " is Odd");
      }
   }
}
```

Output Clear

```
Enter a number: 67
67 is Odd

=== Code Execution Successful ===
```

20.Write a program to find the largest of three numbers entered by the user.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter first number: ");
      int a = sc.nextInt();
      System.out.print("Enter second number: ");
      int b = sc.nextInt();
```

```
        System.out.print("Enter third number: ");
        int c = sc.nextInt();
        int largest;
        if (a >= b && a >= c) {
            largest = a;
        } else if (b >= a && b >= c) {
            largest = b;
        } else {
            largest = c;
        }
        System.out.println("The largest number is: " + largest);
    }
}
```

Output

```
Enter first number: 45
Enter second number: 67
Enter third number: 55
The largest number is: 67

=== Code Execution Successful ===
```

21.Write a program to calculate the factorial of a given number using recursion.

```
import java.util.Scanner;
public class Main {
    static long factorial(int n) {
        if (n == 0 || n == 1)
            return 1;
        else
            return n * factorial(n - 1);
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();
        long result = factorial(num);
        System.out.println("Factorial of " + num + " = " + result);
    }
}
```

Output Clear

```
Enter a number: 55
Factorial of 55 = 6711489344688881664

=== Code Execution Successful ===
```

22.Write a program to check if a given string or number is a palindrome.

```
import java.util.Scanner;
```

```
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a string or number: ");
      String input = sc.nextLine();
      String reversed = "";
      for (int i = input.length() - 1; i >= 0; i--) {
         reversed += input.charAt(i);
      }
      if (input.equals(reversed)) {
         System.out.println(input + " is a Palindrome");
      } else {
         System.out.println(input + " is Not a Palindrome");
      }
   }
}
```

Output Clear

```
Enter a string or number: himachal
himachal is Not a Palindrome

=== Code Exution Successful ===
```

23.Write a program to generate the first n terms of the Fibonacci series.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter number of terms: ");
      int n = sc.nextInt();
      int first = 0, second = 1;
      System.out.println("Fibonacci Series up to " + n + " terms:");
      for (int i = 1; i <= n; i++) {
         System.out.print(first + " ");
         int next = first + second;
         first = second;
         second = next;
      }
   }
}
```

Output Clear

```
Enter number of terms: 77
Fibonacci Series up to 77 terms:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946
    17711 28657 46368 75025 121393 196418 317811 514229 832040 1346269
    2178309 3524578 5702887 9227465 14930352 24157817 39088169 63245986
    102334155 165580141 267914296 433494437 701408733 1134903170 1836311903
    -1323752223 512559680 -811192543 -298632863 -1109825406 -1408458269
    1776683621 368225352 2144908973 -1781832971 363076002 -1418756969
    -1055680967 1820529360 764848393 -1709589543 -944741150 1640636603
    695895453 -1958435240 -1262539787 1073992269 -188547518 885444751
    696897233 1582341984 -2015728079 -433386095 1845853122 1412467027
=== Code Execution Successful ===
```

24.Write a program to check whether a given number is prime.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a number: ");
      int n = sc.nextInt();
      if (n <= 1) {
         System.out.println(n + " is Not Prime");
         return;
      }
      boolean isPrime = true;
      for (int i = 2; i <= n / 2; i++) {
         if (n % i == 0) {
            isPrime = false;
            break;
         }
      }
      if (isPrime)
         System.out.println(n + " is Prime");
      else
         System.out.println(n + " is Not Prime");
   }
}
```

Output Clear

```
Enter a number: 77
77 is Not Prime

=== Code Execution Successful ===
```

25.Write a program to find the sum of all elements in an array.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter number of elements: ");
      int n = sc.nextInt();
      int[] arr = new int[n];
      System.out.println("Enter " + n + " numbers:");
      for (int i = 0; i < n; i++) {
         arr[i] = sc.nextInt();
      }
      int sum = 0;
      for (int i = 0; i < n; i++) {
         sum += arr[i];
      }
      System.out.println("Sum of array elements: " + sum);
   }
}
```

```
Output                                                    Clear
Enter number of elements: 3
Enter 3 numbers:
1
4
9
Sum of array elements: 14

=== Code Execution Successful ===
```

26.Implement a program to reverse the elements of an array.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter number of elements: ");
      int n = sc.nextInt();
      int[] arr = new int[n];
      System.out.println("Enter " + n + " numbers:");
      for (int i = 0; i < n; i++) {
         arr[i] = sc.nextInt();
      }
      System.out.println("Original array:");
      for (int i = 0; i < n; i++) {
         System.out.print(arr[i] + " ");
      }
      System.out.println("\nReversed array:");
      for (int i = n - 1; i >= 0; i--) {
         System.out.print(arr[i] + " ");
      }
   }
}
```

```
Output                                                    Clear
Enter number of elements: 5
Enter 5 numbers:
987654
2345
46546
56457
3464564
Original array:
987654 2345 46546 56457 3464564
Reversed array:
3464564 56457 46546 2345 987654
=== Code Execution Successful ===
```

27.Write a Java program to perform matrix addition and multiplication.

```
import java.util.Scanner;
```

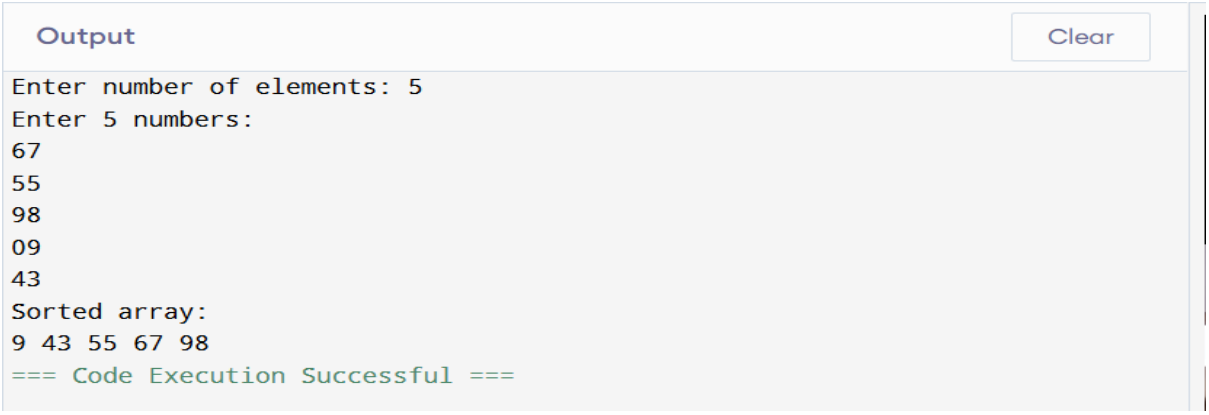
```
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter rows: ");
      int r = sc.nextInt();
      System.out.print("Enter columns: ");
      int c = sc.nextInt();
      int[][] A = new int[r][c];
      int[][] B = new int[r][c];
      System.out.println("Enter elements of Matrix A:");
      for (int i = 0; i < r; i++)
         for (int j = 0; j < c; j++)
            A[i][j] = sc.nextInt();
      System.out.println("Enter elements of Matrix B:");
      for (int i = 0; i < r; i++)
         for (int j = 0; j < c; j++)
            B[i][j] = sc.nextInt();
      int[][] sum = new int[r][c];
      for (int i = 0; i < r; i++)
         for (int j = 0; j < c; j++)
            sum[i][j] = A[i][j] + B[i][j];
      System.out.println("\nMatrix Addition:");
      for (int i = 0; i < r; i++) {
         for (int j = 0; j < c; j++)
            System.out.print(sum[i][j] + " ");
         System.out.println();
      }
      int[][] product = new int[r][c];
      for (int i = 0; i < r; i++)
         for (int j = 0; j < c; j++)
            for (int k = 0; k < c; k++)
               product[i][j] += A[i][k] * B[k][j];
      System.out.println("\nMatrix Multiplication:");
      for (int i = 0; i < r; i++) {
         for (int j = 0; j < c; j++)
            System.out.print(product[i][j] + " ");
         System.out.println();
      }
   }
}
```

```
Enter rows: 2
Enter columns: 3
Enter elements of Matrix A:
1
2
3
4
5
6
Enter elements of Matrix B:
3
4
5
6
7
8

Matrix Addition:
4 6 8
10 12 14
```

28.Create a program to sort an array using the bubble sort algorithm.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter number of elements: ");
      int n = sc.nextInt();
      int[] arr = new int[n];
      System.out.println("Enter " + n + " numbers:");
      for (int i = 0; i < n; i++) {
         arr[i] = sc.nextInt();
      }
      for (int i = 0; i < n - 1; i++) {
         for (int j = 0; j < n - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
               int temp = arr[j];
               arr[j] = arr[j + 1];
               arr[j + 1] = temp;
            }
         }
      }
      System.out.println("Sorted array:");
      for (int num : arr) {
         System.out.print(num + " ");
      }
   }
}
```



29.Write a program to demonstrate a 2D array and print its elements.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter rows: ");
      int rows = sc.nextInt();
      System.out.print("Enter columns: ");
      int cols = sc.nextInt();
      int[][] arr = new int[rows][cols];
      System.out.println("Enter elements of the 2D array:");
```

```
      for (int i = 0; i < rows; i++) {
         for (int j = 0; j < cols; j++) {
            arr[i][j] = sc.nextInt();
         }
      }
      System.out.println("Elements of the 2D array are:");
      for (int i = 0; i < rows; i++) {
         for (int j = 0; j < cols; j++) {
            System.out.print(arr[i][j] + " ");
         }
         System.out.println();
      }
   }
}
```

30.Write a program to search for an element in a sorted array using the binary search algorithm.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter number of elements: ");
      int n = sc.nextInt();
      int[] arr = new int[n];
      System.out.println("Enter " + n + " sorted numbers:");
      for (int i = 0; i < n; i++) {
         arr[i] = sc.nextInt();
      }
      System.out.print("Enter element to search: ");
      int target = sc.nextInt();
      int low = 0, high = n - 1, mid;
      boolean found = false;
      while (low <= high) {
         mid = (low + high) / 2;
         if (arr[mid] == target) {
            System.out.println("Element found at index: " + mid);
            found = true;
            break;
         } else if (arr[mid] < target) {
```

```
            low = mid + 1;
        } else {
            high = mid - 1;
        }
    }
    if (!found) {
        System.out.println("Element not found in the array.");
    }
  }
}
```

```
Output
Enter number of elements: 4
Enter 4 sorted numbers:
4
5
6
7
Enter element to search: 5
Element found at index: 1

=== Code Execution Successful ===
```

31.Write a program to remove duplicate elements from an array.

```
import java.util.Scanner;
import java.util.Arrays;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter number of elements: ");
      int n = sc.nextInt();
      int[] arr = new int[n];
      System.out.println("Enter " + n + " numbers:");
      for (int i = 0; i < n; i++) {
         arr[i] = sc.nextInt();
      }
      Arrays.sort(arr); // sort array first
      int[] temp = new int[n];
      int j = 0;
      for (int i = 0; i < n - 1; i++) {
         if (arr[i] != arr[i + 1]) {
            temp[j++] = arr[i];
         }
      }
      temp[j++] = arr[n - 1]; // add last element
      System.out.println("Array after removing duplicates:");
      for (int i = 0; i < j; i++) {
         System.out.print(temp[i] + " ");
      }
   }
}
```

Output Clear

```
Enter number of elements: 7
Enter 7 numbers:
1
34
78
98
78
44
1
Array after removing duplicates:
1 34 44 78 98
=== Code Execution Successful ===
```

32.Write a program to demonstrate the use of arithmetic, relational, and logical operators.

```
public class Main {
   public static void main(String[] args) {
      int a = 7, b = 19;
      System.out.println("Arithmetic Operators:");
      System.out.println("a + b = " + (a + b));
      System.out.println("a - b = " + (a - b));
      System.out.println("a * b = " + (a * b));
      System.out.println("b / a = " + (b / a));
      System.out.println("b % a = " + (b % a));
      System.out.println("\nRelational Operators:");
      System.out.println("a == b: " + (a == b));
      System.out.println("a != b: " + (a != b));
      System.out.println("a > b: " + (a > b));
      System.out.println("a < b: " + (a < b));
      boolean x = true, y = false;
      System.out.println("\nLogical Operators:");
      System.out.println("x && y: " + (x && y));
      System.out.println("x || y: " + (x || y));
      System.out.println("!x: " + (!x));
   }
}
```

Output Clear

```
Arithmetic Operators:
a + b = 26
a - b = -12
a * b = 133
b / a = 2
b % a = 5

Relational Operators:
a == b: false
a != b: true
a > b: false
a < b: true

Logical Operators:
x && y: false
x || y: true
!x: false

=== Code Execution Successful ===
```

33.Create a program to show the difference between == and equals() for string comparison.

```
public class Main {
   public static void main(String[] args) {
      String s1 = "Jadeja";
      String s2 = "Ravindra";
      String s3 = new String("Ravindra");

      System.out.println("Comparing s1 and s2:");
      System.out.println("Using == : " + (s1 == s2));       // true (same memory reference)
      System.out.println("Using equals() : " + s1.equals(s2)); // true (same content)

      System.out.println("\nComparing s1 and s3:");
      System.out.println("Using == : " + (s1 == s3));       // false (different memory objects)
      System.out.println("Using equals() : " + s1.equals(s3)); // true (same content)
   }
}
```

```
Output

Comparing s1 and s2:
Using == : false
Using equals() : false

Comparing s1 and s3:
Using == : false
Using equals() : false

=== Code Execution Successful ===
```

34.Write a program to illustrate the use of the ternary operator.

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a number: ");
      int num = sc.nextInt();
      String result = (num % 2 == 0) ? "Even" : "Odd";
      System.out.println(num + " is " + result);
   }
}
```

```
Output                                                    Clear

Enter a number: 45
45 is Odd

=== Code Execution Successful ===
```

35.Implement a program to perform bitwise operations in Java.

```
public class Main {
   public static void main(String[] args) {
      int a = 5;
      int b = 3;
      System.out.println("a = " + a + ", b = " + b);
      System.out.println("a & b = " + (a & b));
```

```
        System.out.println("a | b = " + (a | b));
        System.out.println("a ^ b = " + (a ^ b));
        System.out.println("~a = " + (~a));
        System.out.println("a << 1 = " + (a << 1));
        System.out.println("a >> 1 = " + (a >> 1));
    }
}
```

Output

```
a = 5, b = 3
a & b = 1
a | b = 7
a ^ b = 6
~a = -6
a << 1 = 10
a >> 1 = 2

=== Code Execution Successful ===
```

36.Write a program to demonstrate operator precedence in Java.

```
public class Main {
    public static void main(String[] args) {
        int a = 10, b = 5, c = 2;
        int result1 = a + b * c;
        int result2 = (a + b) * c;
        int result3 = a + b / c;
        int result4 = a - b + c;
        System.out.println("a + b * c = " + result1);
        System.out.println("(a + b) * c = " + result2);
        System.out.println("a + b / c = " + result3);
        System.out.println("a - b + c = " + result4);
    }
}
```

Output Clear

```
a + b * c = 20
(a + b) * c = 30
a + b / c = 12
a - b + c = 7

=== Code Execution Successful ===
```

37.Write a program to create a class with multiple constructors (constructor overloading).

```
public class Student {
    String name;
    int age;
    int rollNo;

    // Constructor with name only
    Student(String n) {
        name = n;
```

```
        age = 0;
        rollNo = 0;
    }

    // Constructor with name and age
    Student(String n, int a) {
        name = n;
        age = a;
        rollNo = 0;
    }

    // Constructor with name, age, and roll number
    Student(String n, int a, int r) {
        name = n;
        age = a;
        rollNo = r;
    }

    void display() {
        System.out.println("Name: " + name + ", Age: " + age + ", Roll No: " + rollNo);
    }

    public static void main(String[] args) {
        Student s1 = new Student("Ravindra");
        Student s2 = new Student("Jadeja", 34);
        Student s3 = new Student("Ravi", 21, 7);

        s1.display();
        s2.display();
        s3.display();
    }
}
```

Output

```
Name: Ravindra, Age: 0, Roll No: 0
Name: Jadeja, Age: 34, Roll No: 0
Name: Ravi, Age: 21, Roll No: 7

=== Code Execution Successful ===
```

38.Implement a program to demonstrate the use of a copy constructor in Java.

```
public class Student {
    String name;
    int age;
    int rollNo;
    Student(String n, int a, int r) {
        name = n;
        age = a;
```

```
        rollNo = r;
    }
    Student(Student s) {
        name = s.name;
        age = s.age;
        rollNo = s.rollNo;
    }
    void display() {
        System.out.println("Name: " + name + ", Age: " + age + ", Roll No: " + rollNo);
    }
    public static void main(String[] args) {
        Student s1 = new Student("Ravindra Jadeja", 34, 8);
        Student s2 = new Student(s1); // Using copy constructor
        System.out.println("Original Student:");
        s1.display();
        System.out.println("Copied Student:");
        s2.display();
    }
}
```

Output Clear

```
Original Student:
Name: Ravindra Jadeja, Age: 34, Roll No: 8
Copied Student:
Name: Ravindra Jadeja, Age: 34, Roll No: 8

=== Code Execution Successful ===
```

39.Create a program that initializes class fields using a parameterized constructor.

```
public class Student {
    String name;
    int age;
    int rollNo;
    Student(String n, int a, int r) {
        name = n;
        age = a;
        rollNo = r;
    }
    void display() {
        System.out.println("Name: " + name + ", Age: " + age + ", Roll No: " + rollNo);
    }
    public static void main(String[] args) {
        Student s1 = new Student("Ravindra Jadeja", 34, 8);
        Student s2 = new Student("Ravi", 21, 7);
        s1.display();
        s2.display();
    }
}
```

```
Output                                                    Clear
Name: Ravindra Jadeja, Age: 34, Roll No: 8
Name: Ravi, Age: 21, Roll No: 7

=== Code Execution Successful ===
```

40.Write a program to demonstrate the use of static and non-static methods.

```
public class Student {
    String name;
    int rollNo;
    void setDetails(String n, int r) {
        name = n;
        rollNo = r;
    }
    void display() {
        System.out.println("Name: " + name + ", Roll No: " + rollNo);
    }
    static void collegeName() {
        System.out.println("College: NIT Hamirpur");
    }
    public static void main(String[] args) {
        Student.collegeName();
        Student s1 = new Student();
        Student s2 = new Student();
        s1.setDetails("Ravindra Jadeja", 8);
        s2.setDetails("Ravi Kumar", 21);
        s1.display();
        s2.display();
    }
}
```

```
Output                                                    Clear
Name: Ravindra Jadeja, Age: 34, Roll No: 8
Name: Ravi, Age: 21, Roll No: 7

=== Code Execution Successful ===
```

41.Implement a singleton class in Java.

```
class Singleton {
    private static Singleton singleInstance = null;
    public String message;
    private Singleton() {
        message = "Hello! I am a Singleton instance.";
    }
    public static Singleton getInstance() {
        if (singleInstance == null)
            singleInstance = new Singleton();
        return singleInstance;
```

```
    }
}
public class Main {
    public static void main(String[] args) {
        Singleton obj1 = Singleton.getInstance();
        Singleton obj2 = Singleton.getInstance();
        System.out.println(obj1.message);
        if (obj1 == obj2) {
            System.out.println("Both references point to the same instance.");
        } else {
            System.out.println("Different instances exist.");
        }
    }
}
```

Output Clear

```
Hello! I am a Singleton instance.
Both references point to the same instance.

=== Code Execution Successful ===
```

1. **Write a program to demonstrate multilevel inheritance in Java.**

```
class Student {
    int rollNo;
    String name;
    Student(int r, String n) {
        rollNo = r;
        name = n;
    }
}
class CollegeStudent extends Student {
    String course;
    CollegeStudent(int r, String n, String c) {
        super(r, n);
        course = c;
    }
}
class Result extends CollegeStudent {
    int marks;
    Result(int r, String n, String c, int m) {
        super(r, n, c);
        marks = m;
    }
    void display() {
        System.out.println("Roll No: " + rollNo + ", Name: " + name +
                           ", Course: " + course + ", Marks: " + marks);
    }
}
public class Main {
```

```
    public static void main(String[] args) {
        Result r1 = new Result(7, "Ravindra Jadeja", "B.Tech AI", 95);
        r1.display();
    }
}
```

Output Clear

```
Roll No: 7, Name: Ravindra Jadeja, Course: B.Tech AI, Marks: 95

=== Code Execution Successful ===
```

2. **Create a program to show method overriding and the use of super to call the parent class method.**

```
import java.util.Scanner;
class Student {
    int rollNo;
    String name;
    Student(int r, String n) {
        rollNo = r;
        name = n;
    }
    void display() {
        System.out.println("Roll No: " + rollNo + ", Name: " + name);
    }
}
class CollegeStudent extends Student {
    String course;
    CollegeStudent(int r, String n, String c) {
        super(r, n);
        course = c;
    }
    void display() {
        super.display(); // call parent method
        System.out.println("Course: " + course);
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter Roll No: ");
        int roll = sc.nextInt();
        sc.nextLine(); // consume newline
        System.out.print("Enter Name: ");
        String name = sc.nextLine();
        System.out.print("Enter Course: ");
        String course = sc.nextLine();
        CollegeStudent s1 = new CollegeStudent(roll, name, course);
        s1.display();
    }
}
```

3. **Implement an abstract class and override its methods in a subclass.**

```
import java.util.Scanner;
abstract class Student {
   int rollNo;
   String name;
   Student(int r, String n) {
      rollNo = r;
      name = n;
   }
   abstract void display(); // abstract method
}
class CollegeStudent extends Student {
   String course;
   CollegeStudent(int r, String n, String c) {
      super(r, n);
      course = c;
   }
   void display() {
      System.out.println("Roll No: " + rollNo + ", Name: " + name + ", Course: " + course);
   }
}
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter Roll No: ");
      int roll = sc.nextInt();
      sc.nextLine();
      System.out.print("Enter Name: ");
      String name = sc.nextLine();
      System.out.print("Enter Course: ");
      String course = sc.nextLine();
      CollegeStudent s1 = new CollegeStudent(roll, name, course);
      s1.display();
   }
}
```

4. Write a program to demonstrate final classes and methods.

```
final class Cricketer {
   String name;
   int runs;
   Cricketer(String n, int r) {
      name = n;
      runs = r;
   }
   final void showPerformance() {
      System.out.println(name + " scored " + runs + " runs.");
   }
}
public class Main {
   public static void main(String[] args) {
      Cricketer jadeja = new Cricketer("Ravindra Jadeja", 87);
      jadeja.showPerformance();
   }
```

```
}
```

5. Create a program to show run-time polymorphism using dynamic method dispatch.

```
import java.util.Scanner;
class Cricketer {
   void action() {
      System.out.println("A cricketer is on the ground.");
   }
}
class Batsman extends Cricketer {
   void action() {
      System.out.println("The batsman hits a six!");
   }
}
class Bowler extends Cricketer {
   void action() {
      System.out.println("The bowler delivers a fast yorker!");
   }
}
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter role (batsman/bowler): ");
      String role = sc.nextLine();

      Cricketer player;
      if (role.equalsIgnoreCase("batsman"))
         player = new Batsman();
      else
         player = new Bowler();
      player.action(); // dynamic method dispatch
   }
}
```

String Class and Operations

1.Write a program to reverse a string without using built-in methods.

```
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a string: ");
      String str = sc.nextLine();

      String reversed = "";
      for (int i = str.length() - 1; i >= 0; i--) {
         reversed += str.charAt(i);
      }

      System.out.println("Reversed string: " + reversed);
   }
}
```

2.Implement a program to count the frequency of characters in a string.

```
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a string: ");
      String str = sc.nextLine();

      int[] freq = new int[256]; // ASCII character frequencies

      for (int i = 0; i < str.length(); i++) {
         freq[str.charAt(i)]++;
      }

      System.out.println("Character frequencies:");
      for (int i = 0; i < 256; i++) {
         if (freq[i] > 0) {
            System.out.println((char) i + " : " + freq[i]);
         }
      }
   }
}
```

3.Write a program to demonstrate the immutability of the String class.

```
public class Main {
   public static void main(String[] args) {
      String name = "Jadeja";
      System.out.println("Original String: " + name);
      String newName = name.concat(" Sir");
      System.out.println("After concat operation:");
      System.out.println("name: " + name);
      System.out.println("newName: " + newName);
   }
}
```

4.**Create a program to check if a given string is a palindrome.**

```
import java.util.Scanner;
public class Main {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a string: ");
      String str = sc.nextLine();
      String reversed = "";
      for (int i = str.length() - 1; i >= 0; i--) {
         reversed += str.charAt(i);
      }
      if (str.equalsIgnoreCase(reversed)) {
         System.out.println(str + " is a Palindrome.");
      } else {
         System.out.println(str + " is not a Palindrome.");
      }
   }
```

```
}
```

5.Implement a program to split a string into words and print each word on a new line.

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a sentence: ");
        String str = sc.nextLine();
        String[] words = str.split(" "); // split by space
        System.out.println("Words in the sentence:");
        for (String word : words) {
            System.out.println(word);
        }
    }
}
```