Week 01 : Programming Assignment 1

Due on 2025-08-07, 23:59 IST

Write a Java program to check if a given integer is “Positive” or “Negative”.

(0 (Zero) should be considered positive by this program.)

NOTE:

The code you see is **not complete**.

Your task is to complete the code as per the question.
Think of it like a programming puzzle.

(Remember to match the output given exactly, including the spaces and new lines)

(**Passed with presentation error means you will get full marks**)

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	1	Positive	Positive\n	Passed
Test Case 2	-4	Negative	Negative\n	Passed

The due date for submitting this assignment has passed.
2 out of 2 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-06, 23:34 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W01_P1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int number = in.nextInt();
        // Check if the number is Positive or Negative and print accordingly
        if (number >= 0) {
            System.out.println("Positive");
        } else {
            System.out.println("Negative");
        }



in.close();
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W01_P1 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int number = in.nextInt();
// Check if the number is even or odd
        if (number < 0) {
            System.out.print("Negative");
        } else {
            System.out.print("Positive");
        }
in.close();
    }
}
```

Week 01 : Programming Assignment 2

Due on 2025-08-07, 23:59 IST

Write a Java program to calculate the volume of a cylinder given its radius and height.

Formula:

$V = \pi * r^2 * h$

You can use Math.PI for the computation.

NOTE:

The code you see is **not complete**.

Your task is to complete the code as per the question.
Think of it like a programming puzzle.

(This question can be solved in just one line of code)

Sample Test Cases

	Input	Output
Test Case 1	1 1	Volume is: 3.14
Test Case 2	3.5 5.0	Volume is: 192.42

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W01_P2 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        double radius = in.nextDouble();
        double height = in.nextDouble();
double volume = Math.PI * radius * radius * height;
// Display the result
    System.out.printf("Volume is: %.2f", volume);
    in.close();
  }
}
```

Week 01 : Programming Assignment 3

Due on 2025-08-07, 23:59 IST

Write a Java program to print the multiplication table of a given number up to 4.

NOTE:

Print EXACTLY as shown in the sample output.

DO NOT MISS a single space otherwise you will not be scored.

(Remember to match the output given exactly, including the spaces and new lines)

(passed with presentation error means you will get full marks)

Sample Test Cases

	Input	Output
Test Case 1	10	10 x 1 = 10 10 x 2 = 20 10 x 3 = 30 10 x 4 = 40
Test Case 2	5	5 x 1 = 5 5 x 2 = 10 5 x 3 = 15 5 x 4 = 20

The due date for submitting this assignment has passed.

As per our records you have not submitted this assignment.

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W01_P3 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int number = in.nextInt();
// Print the multiplication table
        for (int i = 1; i < 5; i++) {
            System.out.printf("%d x %d = %d\n", number, i, number * i);
        }
in.close();
    }
}
```

Week 01 : Programming Assignment 4

Due on 2025-08-07, 23:59 IST

Complete the code fragment that reads two integer inputs from keyboard and compute the quotient and remainder.

Sample Test Cases

	Input	Output
Test Case 1	−756 8	The Quotient is = −94 The Remainder is = −4
Test Case 2	556 9	The Quotient is = 61 The Remainder is = 7

The due date for submitting this assignment has passed.

As per our records you have not submitted this assignment.

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
public class W01_P4{
	public static void main(String[] args) {
	Scanner sc = new Scanner(System.in);
	int x=sc.nextInt();
	int y=sc.nextInt();
int quotient=x/y;
	 int remainder=x%y;
	 System.out.println("The Quotient is = " + quotient);
	 System.out.println("The Remainder is = " + remainder);
sc.close();
  }
}
```

Week 01 : Programming Assignment 5

Due on 2025-08-07, 23:59 IST

Write a Java program to print the area and perimeter of a rectangle.

Sample Test Cases

	Input	Output
Test Case 1	2.5 6.8	Perimeter is 2*(6.8 + 2.5) = 18.60 Area is 2.5 * 6.8 = 17.00
Test Case 2	5.6 8.5	Perimeter is 2*(8.5 + 5.6) = 28.20 Area is 5.6 * 8.5 = 47.60

The due date for submitting this assignment has passed.

As per our records you have not submitted this assignment.

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
public class W01_P5 {
   public static void main(String[] strings) {
        double width ;
        double height;
        Scanner in = new Scanner(System.in);
        width = in.nextDouble();
        height = in.nextDouble();
// Calculate the perimeter of the rectangle
    double perimeter = 2 * (height + width);

    // Calculate the area of the rectangle
    double area = width * height;
// Print the calculated perimeter using placeholders for values
        System.out.printf("Perimeter is 2*(%.1f + %.1f) = %.2f\n", height, width, perimeter);

// Print the calculated area using placeholders for values
        System.out.printf("Area is %.1f * %.1f = %.2f", width, height, area);
   }
}
```

W02 Programming Assignments 1

Due on 2025-08-07, 23:59 IST

Write a Java program to calculate the area of a rectangle.

The formula for area is:
Area = length × width

You are required to read the length and width from the user, compute the area, and print the result.

This task helps you practice using variables, arithmetic operations, and printing output in Java.

Sample Test Cases

	Input	Output
Test Case 1	7 3	Area is: 21
Test Case 2	5 10	Area is: 50

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.
Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W02_P1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read length and width of the rectangle
        int length = sc.nextInt();
        int width = sc.nextInt();
int area = length * width; // Multiply length and width to get the area
// Print the area
        System.out.print("Area is: " + area);

        sc.close();
    }
}
```

W02 Programming Assignments 2

Due on 2025-08-07, 23:59 IST

Problem Statement

Write a Java program to calculate the perimeter of a rectangle.

The formula for perimeter is:
Perimeter = 2 multiplied by (length + width)

You are required to read the length and width as integers from the user, compute the perimeter, and print the result.

This problem helps in practicing arithmetic operations and output printing in Java.

Sample Test Cases

	Input	Output
Test Case 1	3 5	Perimeter is: 16
Test Case 2	4 6	Perimeter is: 20

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W02_P2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read length and width of the rectangle
        int length = sc.nextInt();
        int width = sc.nextInt();
int perimeter = 2 * (length + width); // Calculate perimeter using the correct formula

// The formula is perimeter = 2 multiplied by (length + width)
System.out.println("Perimeter is: " + perimeter);

        sc.close();
    }
}
```

W02 Programming Assignments 3

Due on 2025-08-07, 23:59 IST

Finding the Maximum Element in an Array

Problem Statement

What is the Maximum Element?
In an array of numbers, the maximum is the largest number among all elements.

In this assignment:

- You will read `n` numbers from the user
- Store them in an array
- Find the largest number among them
- Print the maximum number

This task helps you apply loops and arrays together to solve a real logic-based problem.

Sample Test Cases

	Input	Output
Test Case 1	4 100 20 300 50	Maximum is: 300
Test Case 2	5 15 42 9 28 37	Maximum is: 42

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.
Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W02_P3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();
        int[] arr = new int[n];

        // Read n numbers into array
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        int max = arr[0];  // Assume first element is maximum
for (int i = 1; i < n; i++) {
    if (arr[i] > max) {
        max = arr[i];  // Update maximum if current element is larger
    }
}

/*
Explanation:
- Start with first element as assumed maximum
- Loop through remaining elements
- Update max whenever a larger number is found
- At the end, max holds the largest number
*/
System.out.println("Maximum is: " + max);

        sc.close();
    }
}
```

W02 Programming Assignments 4

Due on 2025-08-07, 23:59 IST

Create a Class and Access Its Member Variable

Problem Statement

In this task, you will practice creating and using a class in Java.

You need to:

1. Create a class called `Rectangle`
2. Declare two integer member variables `length` and `width`
3. In the `main` method, create an object of the `Rectangle` class, assign values to `length` and `width`, and print their sum

This problem helps you understand how to define a class, create objects, and access class members in Java.

Sample Test Cases

	Input	Output
Test Case 1	7 3	Sum of length and width is: 10
Test Case 2	5 10	Sum of length and width is: 15

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.
Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W02_P4 {

    // Declare a class named Rectangle
    static class Rectangle {
        int length;
        int width;
    }

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

        // Read length and width
        int l = sc.nextInt();
        int w = sc.nextInt();

        // Create an object of the Rectangle class
        Rectangle rect = new Rectangle();

        // Assign values to the object's member variables
        rect.length = l;
        rect.width = w;
System.out.println("Sum of length and width is: " + (rect.length + rect.width)); // Print sum using object members
sc.close();
    }
}
```

W02 Programming Assignments 5

Due on 2025-08-07, 23:59 IST

Working with Multiple Classes, Constructors, and the `this` Keyword

Problem Statement

In this task, you will learn how to:

- Declare multiple classes in the same Java program
- Use constructors to initialize values
- Apply the `this` keyword to refer to instance variables

What you need to do:

1. Declare a class called `Circle` with one member variable `radius`
2. Write a constructor for `Circle` that takes `radius` as a parameter and assigns it using the `this` keyword
3. In the `main` method, create an object of `Circle` and print its radius

This task helps understand how classes work together and how constructors and the `this` keyword are used for clarity.

Sample Test Cases

	Input	Output
Test Case 1	`10`	`Radius of the circle is: 10`
Test Case 2	`7`	`Radius of the circle is: 7`

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.
Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W02_P5 {

    // Declare a separate class named Circle
    static class Circle {

        int radius;
Circle(int radius) {
    this.radius = radius; // Assign the parameter to the instance variable using 'this'
}
}

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

        // Read radius value from user
        int r = sc.nextInt();

        // Create an object of Circle class using constructor
        Circle c = new Circle(r);

        // Print the radius using object member
        System.out.println("Radius of the circle is: " + c.radius);

        sc.close();
    }
}
```

Week 03 : Programming Assignment 1

Due on 2025-08-14, 23:59 IST

Write a program which will print a pattern of "*" 's of height "n".
For example:
Input:
n = 3
Output:
***
**
*
**
***

(Remember to match the output given exactly, including the spaces and new lines)

(passed with presentation error means you will get full marks)

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	4	****\n ***\n **\n *\n **\n ***\n ****	****\n ***\n **\n *\n **\n ***\n ****\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-13, 22:54 IST

Your last recorded submission was :

```java
import java.util.*;
public class W03_P1{
    public static void main(String[] args) {
        Scanner inr = new Scanner(System.in);
       int n = inr.nextInt();
        // Top part
        for (int i = n; i > 1; i--) {
            for (int j = 1; j <= i; j++) {
                System.out.print("*");
            }
            System.out.println();
        }

        // Bottom part
        for (int i = 1; i <= n; i++) {
            for (int j = 1; j <= i; j++) {
                System.out.print("*");
            }
            System.out.println();
        }
}
}
```

Sample solutions (Provided by instructor)

```java
import java.util.*;
public class W03_P1{
    public static void main(String[] args) {
        Scanner inr = new Scanner(System.in);
       int n = inr.nextInt();
int i, j;

        // outer loop to handle rows
        for (i = n; i >= 1; i--) {

            // inner loop to handle columns
            for (j = 1; j <= i; j++) {
                System.out.print("*");
            }

            // printing new line for each row
            System.out.println();
        }

        // outer loop to handle rows
        for (i = 2; i <= n; i++) {

            // inner loop to handle columns
            for (j = 1; j <= i; j++) {
                System.out.print("*");
            }

            // printing new line for each row
            System.out.println();
        }
}
```

Week 03 : Programming Assignment 2

Due on 2025-08-14, 23:59 IST

Complete the code segment to display the factors of a number n.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	60	1 2 3 4 5 6 10 12 15 20 30 60	1 2 3 4 5 6 10 12 15 20 30 60	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-13, 22:56 IST

Your last recorded submission was :

```java
import java.util.Scanner;
public class W03_P2{
    public static void main(String[] args) {
       Scanner sc = new Scanner(System.in);
       int num =sc.nextInt();
        for (int i = 1; i <= num; i++) {
            if (num % i == 0) {
                System.out.print(i + " ");
            }
        }

}
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
public class W03_P2{
    public static void main(String[] args) {
       Scanner sc = new Scanner(System.in);
       int num =sc.nextInt();
for (int i = 1; i <= num; i++) {

            // if number is divided by i
            // i is the factor
            if (num % i == 0) {
              System.out.print(i + " ");
            }
          }
}
}
```

Week 03 : Programming Assignment 3

Due on 2025-08-14, 23:59 IST

Complete the code segment to count number of digits in an integer using while loop.

(Remember to match the output given exactly, including the spaces and new lines)

(passed with presentation error means you will get full marks)

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	123456	6	6\n	Passed

The due date for submitting this assignment has passed.

1 out of 1 tests passed.

You scored 100.0/100.

Assignment submitted on 2025-08-13, 22:56 IST

Your last recorded submission was :

```java
import java.util.Scanner;
public class W03_P3{
    public static void main(String[] args) {
       Scanner sc = new Scanner(System.in);
       int num=sc.nextInt();

        int count = 0;
        while (num != 0) {
            num /= 10;
            count++;
        }
        System.out.println(count);

}
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
public class W03_P3{
    public static void main(String[] args) {
       Scanner sc = new Scanner(System.in);
       int num=sc.nextInt();
int count = 0;

      while (num != 0) {
        // num = num/10
        num /= 10;
        ++count;
      }

      System.out.print(count);
}
}
```

Week 03 : Programming Assignment 4

Due on 2025-08-14, 23:59 IST

A Student class with private fields (name, age) is provided,
Your task is to make the following:
a parameterized constructor to initialize the private fields
the getter/setter methods for each field
Follow the naming convention as given in the main method of the suffix code.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	Bob 18	Name: Bob, Age: 18	Name: Bob, Age: 18	Passed
Test Case 2	Eve 22	Name: Eve, Age: 22	Name: Eve, Age: 22	Passed

The due date for submitting this assignment has passed.
2 out of 2 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-13, 23:04 IST

Your last recorded submission was :

```java
import java.util.Scanner;
class Student {
    private String name;
    private int age;

    public Student(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }


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

        // System.out.print("Enter student name: ");
        String name = scanner.next();

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

        Student student = new Student(name, age);

        System.out.print("Name: " + student.getName() + ", Age: " + student.getAge());

        scanner.close();
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
class Student {
    private String name;
    private int age;
public Student(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }
public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // System.out.print("Enter student name: ");
        String name = scanner.next();

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

        Student student = new Student(name, age);

        System.out.print("Name: " + student.getName() + ", Age: " + student.getAge());

        scanner.close();
    }
}
```

Week 03 : Programming Assignment 5

Due on 2025-08-14, 23:59 IST

There are two class cls1 and cls2 which is subclass of cls1. cls1 having a method "add" which add two numbers. Create two method inside cls2 which will take 2 parameters as input i.e. a and b and print the sum , multiplication and sum of their squares i.e (a^2) + (b 2).
(Remember to match the output given exactly, including the spaces and new lines)
(passed with presentation error means you will get full marks)

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	3 5	8\n 15\n 34	8\n 15\n 34\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-13, 23:00 IST

Your last recorded submission was :

```java
import java.util.Scanner;
class cls1
{
    void add(int p,int q)
    {
        System.out.println(p+q);
    }
}

class cls2 extends cls1 {
    void mul(int p, int q) {
        System.out.println(p * q);
    }
    void task(int p, int q) {
        System.out.println((p * p) + (q * q));
    }
}


public class W03_P5{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);

        cls2 obj=new cls2();
        int a=sc.nextInt();
        int b=sc.nextInt();
        //String tilde=sc.next();
        obj.add(a,b);
        obj.mul(a,b);
        obj.task(a,b);

    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
class cls1
{
    void add(int p,int q)
    {
        System.out.println(p+q);
    }
}
class cls2 extends cls1
{
    void mul(int p,int q)
    {
        System.out.println(p*q);
    }
    void task(int p,int q)
    {
      System.out.print((p*p)+(q*q));
    }
}
public class W03_P5{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);

        cls2 obj=new cls2();
        int a=sc.nextInt();
        int b=sc.nextInt();
        //String tilde=sc.next();
        obj.add(a,b);
        obj.mul(a,b);
        obj.task(a,b);

    }
}
```

W04 Programming Assignments 1

Due on 2025-08-21, 23:59 IST

Understanding Default Access Modifier in Java

Problem Statement

In Java, if no access modifier is written before a class member (variable or method), it is called **Default Access Modifier**.

Members with default access are accessible within the same package, which in beginner programs means within the same file.

Task:

- Create a class called `Student`
- Declare an integer variable `rollNo` with default access (do not write any modifier)
- In the `main` method, create an object of `Student` and print the `rollNo`

This task shows that default access members can be accessed within the same file.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	105	Roll Number is: 105	Roll Number is: 105\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-13, 23:09 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W04_P1 {

    // Declare a class Student with one member variable of default access
    static class Student {
        int rollNo; // Default access modifier (no keyword written)

        // Constructor to assign rollNo
        Student(int r) {
            rollNo = r;
        }
    }

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

        // Read roll number
        int r = sc.nextInt();

        // Create Student object with roll number
        Student s = new Student(r);
        System.out.println("Roll Number is: " + s.rollNo);

sc.close();
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W04_P1 {

    // Declare a class Student with one member variable of default access
    static class Student {
        int rollNo; // Default access modifier (no keyword written)

        // Constructor to assign rollNo
        Student(int r) {
            rollNo = r;
        }
    }

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

        // Read roll number
        int r = sc.nextInt();

        // Create Student object with roll number
        Student s = new Student(r);
System.out.print("Roll Number is: " + s.rollNo); // Access default member within same file
sc.close();
    }
}
```

W04 Programming Assignments 2

Due on 2025-08-21, 23:59 IST

Understanding Public Access Modifier in Java

Problem Statement

In Java, members (variables or methods) declared with the **public** access modifier can be accessed from anywhere, including outside their class.

Task:

- Create a class called `Car`
- Declare a `public` integer variable called `speed`
- In the `main` method, create an object of `Car`, assign a value to `speed`, and print it

This demonstrates how public members can be accessed from outside the class.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	100	Speed is: 100	Speed is: 100\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-13, 23:09 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W04_P2 {

    // Declare class Car with a public member variable
    static class Car {
        public int speed; // Public access, can be accessed from outside the class
    }

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

        // Read speed value
        int s = sc.nextInt();

        // Create Car object
        Car c = new Car();

        // Assign speed to the object
        c.speed = s;
        System.out.println("Speed is: " + c.speed);

sc.close();
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W04_P2 {

    // Declare class Car with a public member variable
    static class Car {
        public int speed; // Public access, can be accessed from outside the class
    }

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

        // Read speed value
        int s = sc.nextInt();

        // Create Car object
        Car c = new Car();

        // Assign speed to the object
        c.speed = s;
System.out.println("Speed is: " + c.speed); // Access and print public member

/*
Explanation:
- The 'speed' variable is declared as public inside the Car class.
- Public members can be accessed anywhere in the program using the object reference.
- Here, c.speed accesses the variable, and we print it.
- This shows how public access works in Java.
*/
sc.close();
    }
}
```

Problem Statement

In Java, members declared with the **private** access modifier cannot be accessed directly from outside the class.
To access private members, you must use **methods within the same class**, often called "getter" methods.

Task:

- Create a class called `Account`
- Declare a private integer variable `balance`
- Create a public method `getBalance()` to return the balance
- In the `main` method, create an object of `Account`, set a balance value, and print it using the method

This demonstrates how private members can be accessed safely using methods.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	`0`	`Account Balance is: 0`	`Account Balance is: 0\n`	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-13, 23:11 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W04_P3 {

    // Declare class Account with a private member variable
    static class Account {
        private int balance; // Private member, cannot be accessed directly from outside

        // Method to set balance value
        public void setBalance(int b) {
            balance = b;
        }
        public int getBalance() {
            return balance;
        }



}

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

        // Read balance value
        int b = sc.nextInt();

        // Create Account object
        Account acc = new Account();

        // Set balance using method
        acc.setBalance(b);

        // Print balance using the method
        System.out.println("Account Balance is: " + acc.getBalance());

        sc.close();
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W04_P3 {

    // Declare class Account with a private member variable
    static class Account {
        private int balance; // Private member, cannot be accessed directly from outside

        // Method to set balance value
        public void setBalance(int b) {
            balance = b;
        }
public int getBalance() {
    return balance; // Return the private member value
}

/*
Explanation:
- The 'balance' variable is private, so it cannot be accessed directly from outside the class.
- The method getBalance() is public and provides controlled access to the private variable.
- This follows the principle of encapsulation, which improves security and protects data.
*/
}

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

        // Read balance value
        int b = sc.nextInt();

        // Create Account object
        Account acc = new Account();

        // Set balance using method
        acc.setBalance(b);

        // Print balance using the method
        System.out.println("Account Balance is: " + acc.getBalance());

        sc.close();
    }
```

W04 Programming Assignments 4

Due on 2025-08-21, 23:59 IST

Understanding Protected Access Modifier in Java

Problem Statement

In Java, members declared with the **protected** access modifier are accessible:

- Within the same class
- Within subclasses (even if in different files or packages)
- Within the same package

In beginner programs (single file), protected members behave similarly to default members, but understanding this modifier is important for object-oriented concepts.

Task:

- Create a class called `Employee` with a protected variable `salary`
- Create a subclass `Manager` that inherits from `Employee`
- In the `main` method, create an object of `Manager`, assign a salary value, and print it

This demonstrates how protected members can be accessed through inheritance.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	100000	Salary is: 100000	Salary is: 100000\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-13, 23:12 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W04_P4 {

    // Declare parent class Employee with a protected member variable
    static class Employee {
        protected int salary; // Protected member
    }

    // Subclass Manager inherits from Employee
    static class Manager extends Employee {
        // No additional members required for this task
    }

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

        // Read salary value
        int s = sc.nextInt();

        // Create Manager object
        Manager m = new Manager();
        m.salary = s; // inherited protected member
        System.out.println("Salary is: " + m.salary);

sc.close();
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W04_P4 {

    // Declare parent class Employee with a protected member variable
    static class Employee {
        protected int salary; // Protected member
    }

    // Subclass Manager inherits from Employee
    static class Manager extends Employee {
        // No additional members required for this task
    }

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

        // Read salary value
        int s = sc.nextInt();

        // Create Manager object
        Manager m = new Manager();
m.salary = s; // Assign salary using protected member
System.out.println("Salary is: " + m.salary); // Access and print protected member

/*
Explanation:
- The variable 'salary' is declared protected in the Employee class.
- It can be accessed in the Manager class because of inheritance.
- In this case, both classes are in the same file, so access is allowed directly.
- This demonstrates how protected members can be used safely in subclass relationships.
*/
```

W04 Programming Assignments 5

Due on 2025-08-21, 23:59 IST

Access Modifiers with Method Overloading

Problem Statement

In Java, **Method Overloading** means defining multiple methods with the same name but different parameters. These methods can have different access modifiers like `public`, `private`, etc.

Task:

- Create a class called `Calculator`
- Overload the method `add` as follows:
  - A `public` method `add` that adds two integers
  - A `private` method `add` that adds three integers
- In the `main` method, use the public method to add two numbers
- Inside the class, use the private method to add three numbers and print both results

This shows how access modifiers work with method overloading.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	5 10 2 3 4	Sum of two numbers: 15\n Sum of three numbers: 9	Sum of two numbers: 15\n Sum of three numbers: 9\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-13, 23:13 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W04_P5 {

    // Declare Calculator class with overloaded add methods
    static class Calculator {

        // Public method to add two integers
        public int add(int a, int b) {
            return a + b;
        }


        private int add(int a, int b, int c) { // private method
            return a + b + c;
        }


// Method to demonstrate private method access within class
        public void printThreeSum(int x, int y, int z) {
            int sum = add(x, y, z); // Call private method within class
            System.out.println("Sum of three numbers: " + sum);
        }
    }

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

        // Read two numbers
        int a = sc.nextInt();
        int b = sc.nextInt();

        // Read three numbers
        int x = sc.nextInt();
        int y = sc.nextInt();
        int z = sc.nextInt();

        Calculator calc = new Calculator();

        // Call public method to add two numbers
        int sumTwo = calc.add(a, b);
        System.out.println("Sum of two numbers: " + sumTwo);

        // Call method that prints sum of three numbers
        calc.printThreeSum(x, y, z);

        sc.close();
    }
}
```

Sample solutions (Provided by instructor)

```
        private int add(int a, int b, int c) { // private method
            return a + b + c;
        }


// Method to demonstrate private method access within class
        public void printThreeSum(int x, int y, int z) {
            int sum = add(x, y, z); // Call private method within class
            System.out.println("Sum of three numbers: " + sum);
        }
    }

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

        // Read two numbers
        int a = sc.nextInt();
        int b = sc.nextInt();

        // Read three numbers
        int x = sc.nextInt();
        int y = sc.nextInt();
        int z = sc.nextInt();

        Calculator calc = new Calculator();

        // Call public method to add two numbers
        int sumTwo = calc.add(a, b);
        System.out.println("Sum of two numbers: " + sumTwo);

        // Call method that prints sum of three numbers
        calc.printThreeSum(x, y, z);

        sc.close();
    }
}
```

Sample solutions (Provided by instructor)

```
import java.util.Scanner;

public class W04_P5 {

    // Declare Calculator class with overloaded add methods
    static class Calculator {

        // Public method to add two integers
        public int add(int a, int b) {
            return a + b;
        }
private int add(int a, int b, int c) {
    return a + b + c; // Sum of three integers using private method
}

/*
Explanation:
- The add method with two parameters is public and can be called from anywhere.
- The overloaded add method with three parameters is private and only accessible within the class.
- This shows how access modifiers control visibility in method overloading.
*/
// Method to demonstrate private method access within class
        public void printThreeSum(int x, int y, int z) {
            int sum = add(x, y, z); // Call private method within class
            System.out.println("Sum of three numbers: " + sum);
        }
    }

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

        // Read two numbers
        int a = sc.nextInt();
        int b = sc.nextInt();

        // Read three numbers
        int x = sc.nextInt();
        int y = sc.nextInt();
        int z = sc.nextInt();

        Calculator calc = new Calculator();

        // Call public method to add two numbers
        int sumTwo = calc.add(a, b);
        System.out.println("Sum of two numbers: " + sumTwo);

        // Call method that prints sum of three numbers
        calc.printThreeSum(x, y, z);

        sc.close();
    }
}
```

Week 05 : Programming Assignment 1

Due on 2025-08-28, 23:59 IST

An interface Number is defined in the following program.

You have to declare a class A, which will implement the interface Number.

Note that the method findSqr(n) will return the square of the number n.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	-2	4	4	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-28, 23:25 IST

Your last recorded submission was :

```java
import java.util.Scanner;

interface Number {
    int findSqr(int i);  // Returns the square of n
}
class A implements Number {
    // Define a method to find the square of a number
    public int findSqr(int i) {
        return i * i;  // returns the square
    }
}
public class W05_P1{
        public static void main (String[] args){
          A a = new A();   //Create an object of class A
           // Read a number from the keyboard
           Scanner sc = new Scanner(System.in);
           int i = sc.nextInt();
           System.out.print(a.findSqr(i));
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

interface Number {
    int findSqr(int i);  // Returns the square of n
}
class A implements Number {
    //Define a method to find the square of a number
     int i, square;
     public int findSqr(int i) {
         square=i*i;
            return square;
      }
}
public class W05_P1{
        public static void main (String[] args){
          A a = new A();   //Create an object of class A
           // Read a number from the keyboard
           Scanner sc = new Scanner(System.in);
           int i = sc.nextInt();
           System.out.print(a.findSqr(i));
    }
}
```

Week 05 : Programming Assignment 2

Due on 2025-08-28, 23:59 IST

This program is to find the GCD (greatest common divisor) of two integers writing a recursive function findGCD(n1,n2).
Your function should return -1, if the argument(s) is(are) negative (zero is allowed).

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	3 -1	-1	-1	Passed
Test Case 2	30 20	10	10	Passed

The due date for submitting this assignment has passed.
2 out of 2 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-28, 23:26 IST

Your last recorded submission was :

```java
import java.util.Scanner;

interface GCD {
    public int findGCD(int n1,int n2);
}
// Create a class B, which implements the interface GCD
class B implements GCD {
    // Create a method to calculate GCD
    public int findGCD(int n1, int n2) {
        if (n1 < 0 || n2 < 0) {
            return -1;  // Handle negative input case
        }
        if (n2 == 0) {
            return n1;  // Base case
        } else {
            return findGCD(n2, n1 % n2);  // Recursive step (Euclidean Algorithm)
        }
    }
}
public class W05_P2{
        public static void main (String[] args){
          B a = new B();   //Create an object of class B
            // Read two numbers from the keyboard
            Scanner sc = new Scanner(System.in);
            int p1 = sc.nextInt();
           int p2 = sc.nextInt();
            System.out.print(a.findGCD(p1,p2));
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

interface GCD {
    public int findGCD(int n1,int n2);
}
class B implements GCD {
    int n1,n2;

    //Create a method to calculate GCD
    public int findGCD(int n1, int n2){
        if(n1==0&& n2==0) {
            return -1;
        }
        else if(n2 == 0){
            return n1;
        }

        else {
            return findGCD(n2, n1%n2);
        }
        }
 }
public class W05_P2{
        public static void main (String[] args){
          B a = new B();   //Create an object of class B
            // Read two numbers from the keyboard
            Scanner sc = new Scanner(System.in);
            int p1 = sc.nextInt();
           int p2 = sc.nextInt();
            System.out.print(a.findGCD(p1,p2));
    }
}
```

Week 05 : Programming Assignment 3

Due on 2025-08-28, 23:59 IST

Complete the code segment to catch the ArithmeticException in the following, if any.
On the occurrence of such an exception, your program should print “Exception caught: Division by zero.”
If there is no such exception, it will print the result of division operation on two integer values.

Sample Test Cases

	Input	Output
Test Case 1	5 4	1
Test Case 2	1 0	Exception caught: Division by zero.
Test Case 3	4 5	0
Test Case 4	20 0	Exception caught: Division by zero.

The due date for submitting this assignment has passed.
0 out of 0 tests passed.
You scored 0.0/100.

Assignment submitted on 2025-08-28, 23:26 IST

Your last recorded submission was :

```java
import java.util.Scanner;

  public class W05_P3 {
  public static void main(String[] args) {
      int a, b;
      Scanner input = new Scanner(System.in);
       // Read any two values for a and b
       int result;

       a = input.nextInt();
       b = input.nextInt();
//Get the result of a/b;
        int result;

        // try block to divide two numbers and display the result
        try {
            result = a / b;  // division
            System.out.print(result);
        }
        // catch block to catch the ArithmeticException
        catch (ArithmeticException e) {
            System.out.print("Exception caught: Division by zero.");
        }
}
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

  public class W05_P3 {
  public static void main(String[] args) {
      int a, b;
      Scanner input = new Scanner(System.in);
       // Read any two values for a and b
       int result;

       a = input.nextInt();
       b = input.nextInt();
// try block to divide two numbers and display the result
      try {
               result = a/b;
               System.out.print(result);
              }
         // catch block to catch the ArithmeticException
         catch (ArithmeticException e) {
            System.out.print("Exception caught: Division by zero.");
         }
}
}
```

Week 05 : Programming Assignment 4

Due on 2025-08-28, 23:59 IST

In the following program, an array of integer data to be initialized.
During the initialization, if a user enters a value other than integer value, then it will throw InputMismatchException exception.
On the occurrence of such an exception, your program should print “You entered bad data.”
If there is no such exception it will print the total sum of the array.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	2 2 2	4	4	Passed
Test Case 2	1 0.5	You entered bad data.	You entered bad data.	Passed

The due date for submitting this assignment has passed.
2 out of 2 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-28, 23:28 IST

Your last recorded submission was :

```java
//Prefixed Fixed Code:
import java.util.Scanner;
import java.util.InputMismatchException;

public class W05_P4 {
  public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
    int length = sc.nextInt();
    // create an array to save user input
    int[] name = new int[length];
     int sum=0;//save the total sum of the array.

        try {
            // Traverse the array to read inputs and find the total sum
            for (int i = 0; i < length; i++) {
                name[i] = sc.nextInt();
                sum += name[i];
            }
            System.out.print(sum);
        }
        catch (InputMismatchException e) {
            System.out.print("You entered bad data.");
        }
}
}
```

Sample solutions (Provided by instructor)

```java
//Prefixed Fixed Code:
import java.util.Scanner;
import java.util.InputMismatchException;

public class W05_P4 {
  public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
    int length = sc.nextInt();
    // create an array to save user input
    int[] name = new int[length];
     int sum=0;//save the total sum of the array.
//traverse the array to find the total sum of the array.
     try{
       for(int i=0;i<length;i++){
          int userInput=sc.nextInt();
          name[i] = userInput;
          sum=sum+name[i];
          }
        System.out.print(sum);
        }
       catch(InputMismatchException e) {
        System.out.print("You entered bad data.");
        }
}
}
```

Week 05 : Programming Assignment 5

Due on 2025-08-28, 23:59 IST

In the following program, there may be multiple exceptions.
You have to complete the code using only one try-catch block to handle all the possible exceptions.
For example, if user's input is 1, then it will throw and catch "java.lang.NullPointerException".

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	50	No exception	No exception	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-28, 23:28 IST

Your last recorded submission was :

```java
import java.util.Scanner;
public class W05_P5{
      public static void main (String   args[ ] ) {
        Scanner scan = new Scanner(System.in);
          int i=scan.nextInt();
          int j;
        try {
            switch (i) {
                case 0:
                    int zero = 0;
                    j = 92 / zero;   // Division by zero -> ArithmeticException
                    break;
                case 1:
                    int b[] = null;
                    j = b[0];        // Null reference -> NullPointerException
                    break;
                default:
                    System.out.print("No exception");
            }
        } catch (ArithmeticException e) {
            System.out.print(e);
        } catch (NullPointerException e) {
            System.out.print(e);
        }
}
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
public class W05_P5{
      public static void main (String   args[ ] ) {
        Scanner scan = new Scanner(System.in);
          int i=scan.nextInt();
          int j;
//Template code
            try {
                switch (i) {
                case 0 :
                    int zero = 0;
                    j = 92/ zero;
                    break;
                case 1:
                    int b[ ] = null;
                    j = b[0] ;
                    break;
                default:
                    System.out.print("No exception");
                }
            }
            // catch block
            catch (Exception e) {
                System.out.print(e) ;
            }
}
}
```

W06 Programming Assignments 1

Due on 2025-09-04, 23:59 IST

Safe Division with Run-time Error Handling

Problem Statement

In Java, some operations can cause run-time errors, for example dividing a number by zero.
We can use a **try-catch block** to handle such errors and avoid program crashes.

Task:

- Read two integers from the user
- Divide the first number by the second inside a try-catch block
- If the second number is zero, print `"Cannot divide by zero"`
- Otherwise, print the result

This task introduces basic run-time error handling in a safe and controlled way.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	10 2	Result is: 5	Result is: 5\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-28, 23:32 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W06_P1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read two integers
        int num1 = sc.nextInt();
        int num2 = sc.nextInt();

        // Use try-catch to handle possible run-time error
        try {
// TODO: Divide num1 by num2 and print the result ; the poutput should be like "Result is: x"

            int result = num1 / num2;
            System.out.println("Result is: " + result);
} catch (ArithmeticException e) {
            // Print safe message if division by zero occurs
            System.out.println("Cannot divide by zero");
        }

        sc.close();
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W06_P1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read two integers
        int num1 = sc.nextInt();
        int num2 = sc.nextInt();

        // Use try-catch to handle possible run-time error
        try {
System.out.println("Result is: " + (num1 / num2)); // Perform division safely inside try block

/*
Explanation:
- Division is done inside try block to catch errors like division by zero
- If error occurs, catch block prints a safe message
- This prevents program crash and shows controlled error handling
*/
} catch (ArithmeticException e) {
            // Print safe message if division by zero occurs
            System.out.println("Cannot divide by zero");
        }

        sc.close();
    }
}
```

W06 Programming Assignments 2

Due on 2025-09-04, 23:59 IST

Programming Assignment: Nested try-catch Block

Problem Statement

In Java, **nested try-catch blocks** allow handling multiple levels of errors separately.
You can place one try-catch block inside another to handle different types of errors in different places.

Programming Assignment:

- Read two integers from the user
- Inside an outer try-catch block, perform the following:
  - Inside a nested try block, divide the first number by the second
  - If division by zero occurs, handle it with the inner catch block
- In the outer catch block, handle any other unexpected errors
- Print appropriate messages for each scenario

This programming assignment introduces nested try-catch structure.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	10 0	Cannot divide by zero	Cannot divide by zero\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-28, 23:34 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W06_P2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read two integers
        int num1 = sc.nextInt();
        int num2 = sc.nextInt();

        // Outer try-catch block
        try {

            // Inner try-catch block for division operation
            try {
                // Divide num1 by num2 and print result
                int result = num1 / num2;
                System.out.println("Division successful");
                System.out.println("Result is: " + result);


} catch (ArithmeticException e) {
                System.out.println("Cannot divide by zero");
            }

        } catch (Exception e) {
            // Handles other unexpected errors
            System.out.println("An unexpected error occurred");
        }

        sc.close();
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W06_P2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read two integers
        int num1 = sc.nextInt();
        int num2 = sc.nextInt();

        // Outer try-catch block
        try {

            // Inner try-catch block for division operation
            try {
int result = (num1 / num2);
System.out.println("Division successful");
System.out.println("Result is: " + result); // Perform division inside inner try block

/*
Explanation:
- The inner try-catch block handles division-related errors like division by zero
- The outer try-catch handles any other general errors
- This structure improves program safety and reliability
*/
} catch (ArithmeticException e) {
                System.out.println("Cannot divide by zero");
            }

        } catch (Exception e) {
            // Handles other unexpected errors
            System.out.println("An unexpected error occurred");
        }

        sc.close();
    }
}
```

W06 Programming Assignments 3

Cannot divide by zero\n

Due on 2025-09-04, 23:59 IST

Programming Assignment: try Block with Multiple catch Blocks

Problem Statement

In Java, a **try block** can be followed by multiple **catch blocks** to handle different types of errors separately.

This improves error handling by allowing specific actions for different exceptions.

Key Concepts:

- The first matching catch block handles the error
- Catch blocks are written in order from most specific to general

Programming Assignment:

- Read two integers from the user
- Inside a try block, divide the first number by the second
- Handle `ArithmeticException` separately to detect division by zero
- Handle any other general errors using another catch block
- Print suitable messages based on the type of error

This demonstrates structured error handling with multiple catch blocks.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	10 0	Cannot divide by zero	Cannot divide by zero\n	Passed

The due date for submitting this assignment has passed.

1 out of 1 tests passed.

You scored 100.0/100.

Assignment submitted on 2025-08-28, 23:35 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W06_P3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read two integers
        int num1 = sc.nextInt();
        int num2 = sc.nextInt();

        // Try block with multiple catch blocks
        try {
            int result = num1 / num2;
            System.out.println("Division successful");
            System.out.println("Result is: " + result);
} catch (ArithmeticException e) {
            // Handles division by zero error
            System.out.println("Cannot divide by zero");
        } catch (Exception e) {
            // Handles other general errors
            System.out.println("An unexpected error occurred");
        }

        sc.close();
    }
}
```

Sample solutions (Provided by instructor)

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

```
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W06_P3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read two integers
        int num1 = sc.nextInt();
        int num2 = sc.nextInt();

        // Try block with multiple catch blocks
        try {
int result = (num1 / num2);
System.out.println("Division successful");
System.out.println("Result is: " +result  ); // Division inside try block

/*
Explanation:
- First catch block handles division by zero
- Second catch block handles other errors
- Multiple catch blocks improve control over different error types
*/
} catch (ArithmeticException e) {
            // Handles division by zero error
            System.out.println("Cannot divide by zero");
        } catch (Exception e) {
            // Handles other general errors
            System.out.println("An unexpected error occurred");
        }

        sc.close();
    }
}
```

Programming Assignment: Using finally in try-catch Block

Problem Statement

In Java, the **finally block** is a special part of error handling.

What is finally?

- The code inside a finally block always runs, whether there is an error or not
- It is usually used to close resources like files, database connections, or simply to show a message

Programming Assignment:

- Read two integers from the user
- Inside a try block, divide the first number by the second
- If division by zero occurs, show an error message using catch block
- Use a finally block to print `"Program Ended"` no matter what happens

This helps you understand how finally block always runs in a program.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	10 0	Cannot divide by zero\n Program Ended	Cannot divide by zero\n Program Ended\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-28, 23:35 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W06_P4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read two integers
        int num1 = sc.nextInt();
        int num2 = sc.nextInt();

        // try-catch-finally structure
        try {
            int result = num1 / num2;
            System.out.println("Result is: " + result);
} catch (ArithmeticException e) {
            System.out.println("Cannot divide by zero");
        } finally {
            // Print final message, runs always
            System.out.println("Program Ended");
        }

        sc.close();
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W06_P4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Read two integers
        int num1 = sc.nextInt();
        int num2 = sc.nextInt();

        // try-catch-finally structure
        try {
System.out.println("Result is: " + (num1 / num2)); // Perform division inside try block

/*
Explanation:
- If division is successful, result is printed
- If division by zero occurs, catch block shows error
- finally block prints "Program Ended" every time, showing reliable code execution
*/
} catch (ArithmeticException e) {
            System.out.println("Cannot divide by zero");
        } finally {
            // Print final message, runs always
            System.out.println("Program Ended");
        }

        sc.close();
    }
}
```


// Unarchiver

W06 Programming Assignments 5

Due on 2025-09-04, 23:59 IST

Programming Assignment: Using throws Statement for Error Handling

Problem Statement

In Java, the `throws` keyword is used when a method might cause an error, but the method itself does not handle it. Instead, it passes the responsibility to the caller of the method.

Why use throws?

- Some methods may cause errors called "checked exceptions"
- Instead of handling the error inside the method, we declare `throws` to inform the caller

Programming Assignment:

- Create a method called `calculateSquareRoot`
- The method reads a number and returns its square root
- If the number is negative, it throws an `Exception`
- In the `main` method, use a try-catch block to handle the error

This demonstrates how to use `throws` and handle errors safely in the caller method.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	-25	Cannot calculate square root of negative number	Cannot calculate square root of negative number\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-08-28, 23:36 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W06_P5 {

    // Method to calculate square root, may throw Exception
    public static double calculateSquareRoot(double num) throws Exception {
        // TODO: Throw Exception if number is negative
        if (num < 0) {
            throw new Exception("Cannot calculate square root of negative number");
        }
        // For valid input, return square root using Math.sqrt(num)
        return Math.sqrt(num);
}

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

        double number = sc.nextDouble();

        try {
            double result = calculateSquareRoot(number);
            System.out.println("Square root is: " + result);
        } catch (Exception e) {
            System.out.println("Cannot calculate square root of negative number");
        }

        sc.close();
    }
}
```

```java
import java.util.Scanner;

public class W06_P5 {

    // Method to calculate square root, may throw Exception
    public static double calculateSquareRoot(double num) throws Exception {
if (num < 0) {
    throw new Exception("Negative number"); // Throw exception if input is invalid
}
return Math.sqrt(num); // Return square root for valid input

/*
Explanation:
- If number is negative, method throws an Exception
- Caller (main method) uses try-catch to handle error
- This shows how throws works to pass error handling to another part of the program
*/
}

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

        double number = sc.nextDouble();

        try {
            double result = calculateSquareRoot(number);
            System.out.println("Square root is: " + result);
        } catch (Exception e) {
            System.out.println("Cannot calculate square root of negative number");
        }

        sc.close();
    }
}
```

Week 07 : Programming Assignment 1

Due on 2025-09-11, 23:59 IST

Write a Java program to find longest word in given input.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	`This is an NPTEL course`	`The longest word in the text is: course`	`The longest word in the text is: course\n`	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-10, 22:54 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W07_P1 {
    // Method to find the longest word in a given text
    public static String findLongestWord(String text) {
        String longestWord = "";

        // Split the text into words based on whitespace
        String[] words = text.split("\\s+");

        // Iterate through each word to find the longest one
        for (String word : words) {
            if (word.length() > longestWord.length()) {
                longestWord = word;
            }
        }

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

        // Prompt user to enter text
       // System.out.println("Enter some text:");
        String text = scanner.nextLine();

        // Close the scanner
        scanner.close();

        // Call the method to find the longest word
        String longestWord = findLongestWord(text);

        // Print the longest word found
        System.out.println("The longest word in the text is: " + longestWord);
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W07_P1 {
// Method to find the longest word in a given text
    public static String findLongestWord(String text) {
        String longestWord = "";

        // Split the text into words based on whitespace
        String[] words = text.split("\\s+");

        // Iterate through each word to find the longest one
        for (String word : words) {
            if (word.length() > longestWord.length()) {
                longestWord = word;
            }
        }

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

        // Prompt user to enter text
       // System.out.println("Enter some text:");
        String text = scanner.nextLine();

        // Close the scanner
        scanner.close();

        // Call the method to find the longest word
        String longestWord = findLongestWord(text);

        // Print the longest word found
        System.out.println("The longest word in the text is: " + longestWord);
    }
}
```

Week 07 : Programming Assignment 2

Due on 2025-09-11, 23:59 IST

Write a Java program to find longest word in given input.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	6 12 45 67 12 34 12 12	Original Array: [12, 45, 67, 12, 34, 12]\n Array after removing 12: [45, 67, 34]	Original Array: [12, 45, 67, 12, 34, 12]\n Array after removing 12: [45, 67, 34]	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-10, 22:55 IST

Your last recorded submission was :

```java
import java.util.*;

public class W07_P2 {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
            // Input array
      //  System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int[] array = new int[n];

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

        // Element to remove
    //    System.out.print("Enter the element to remove: ");
        int elementToRemove = scanner.nextInt();

        // Close the scanner
        scanner.close();

        // Removing element and printing result
        System.out.println("Original Array: " + Arrays.toString(array));
        array = removeAll(array, elementToRemove);
        System.out.print("Array after removing " + elementToRemove + ": " + Arrays.toString(array));
    }
    // Program to remove all occurrences of an element from array in Java
    public static int[] removeAll(int[] array, int elementToRemove) {
        int[] result = new int[array.length];
        int index = 0;

        // Iterate through the original array
        for (int value : array) {
            // Copy only if the current element is not equal to the element to be removed
            if (value != elementToRemove) {
                result[index++] = value;
            }
        }

        // If the resulting array is smaller than the original, resize it
        return Arrays.copyOf(result, index);
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.*;

public class W07_P2 {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
            // Input array
      //  System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int[] array = new int[n];

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

        // Element to remove
    //    System.out.print("Enter the element to remove: ");
        int elementToRemove = scanner.nextInt();

        // Close the scanner
        scanner.close();

        // Removing element and printing result
        System.out.println("Original Array: " + Arrays.toString(array));
        array = removeAll(array, elementToRemove);
        System.out.print("Array after removing " + elementToRemove + ": " + Arrays.toString(array));
    }
public static int[] removeAll(int[] array, int elementToRemove) {
        int[] result = new int[array.length];
        int index = 0;

        // Iterate through the original array
        for (int value : array) {
            // Copy only if the current element is not equal to the element to be removed
            if (value != elementToRemove) {
                result[index++] = value;
            }
        }

        // If the resulting array is smaller than the original, resize it
        return Arrays.copyOf(result, index);
```

Week 07 : Programming Assignment 3

Due on 2025-09-11, 23:59 IST

Write a program to compute the sum of all prime numbers in a given range.
The range value will be positive.

Follow the naming convention as given in the main method of the suffix code.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	11 50	311	311\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-10, 22:56 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W07_P3{

    // Function to check if a number is prime
    public static boolean isPrime(int n) {
        if (n <= 1) return false;
        if (n == 2) return true;
        if (n % 2 == 0) return false;
        for (int i = 3; i <= Math.sqrt(n); i += 2) {
            if (n % i == 0) return false;
        }
        return true;
    }

    // Function to compute sum of all prime numbers in range [x, y]
    public static int primeSum(int x, int y) {
        int sum = 0;
        for (int i = x; i <= y; i++) {
            if (isPrime(i)) {
                sum += i;
            }
        }
        return sum;
    }
public static void main(String[] args)
    {
       Scanner sc = new Scanner(System.in);
       int x=sc.nextInt();
       int y=sc.nextInt();

        System.out.println(primeSum(x, y));
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W07_P3{
static boolean checkPrime(int numberToCheck)
    {
        if(numberToCheck == 1) {
            return false;
        }
        for (int i = 2; i*i <= numberToCheck; i++) {
            if (numberToCheck % i == 0) {
                return false;
            }
        }
        return true;
    }

    // Method to iterate the loop from l to r
    // If prime number detects, sum the value
    static int primeSum(int l, int r)
    {
        int sum = 0;
        for (int i = r; i >= l; i--) {

            // Check for prime
            boolean isPrime = checkPrime(i);
            if (isPrime) {

                // Sum the prime number
                sum = sum + i;
            }
        }
        return sum;
    }
public static void main(String[] args)
    {
       Scanner sc = new Scanner(System.in);
       int x=sc.nextInt();
       int y=sc.nextInt();

        System.out.println(primeSum(x, y));
    }
}
```

Week 07 : Programming Assignment 4

Due on 2025-09-11, 23:59 IST

Code to create two threads, one printing even numbers and the other printing odd numbers.

▪ The PrintNumbers class is declared, and it implements the Runnable interface. This interface is part of Java's concurrency support and is used to represent a task that can be executed concurrently by a thread.

▪ Create a constructor of this class that takes two private instance variables (start and end) to represent the range of numbers that will be printed by the thread.

▪ Create a run method that is required by the Runnable interface and contains the code that will be executed when the thread is started. In this case, it should prints odd numbers within the specified range (start to end) using a for loop.

▪ Hint: Thread.currentThread().getName() returns the name of the currently executing thread, which is useful for identifying which thread is printing the numbers.

Follow the naming convention as given in the main method of the suffix code.

Sample Test Cases

	Input	Output
Test Case 1	16 22 55 67	EvenThread: 16 EvenThread: 18 EvenThread: 20 EvenThread: 22 OddThread: 55 OddThread: 57 OddThread: 59 OddThread: 61 OddThread: 63 OddThread: 65 OddThread: 67
Test Case 2	2 10 3 7	EvenThread: 2 EvenThread: 4 EvenThread: 6 EvenThread: 8 EvenThread: 10 OddThread: 3 OddThread: 5 OddThread: 7

The due date for submitting this assignment has passed.
0 out of 0 tests passed.
You scored 0.0/100.

Assignment submitted on 2025-09-10, 22:57 IST

Your last recorded submission was :

```java
import java.util.Scanner;
class PrintNumbers implements Runnable {
    private int start;
    private int end;

    // Constructor
    public PrintNumbers(int start, int end) {
        this.start = start;
        this.end = end;
    }

    @Override
    public void run() {
        for (int i = start; i <= end; i += 2) {
            System.out.println(Thread.currentThread().getName() + ": " + i);
        }
    }
}
}
class W07_P4{
    //Code to create two threads, one printing even numbers and the other printing odd numbers
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        //System.out.print("Enter the starting value for even numbers: ");
        int evenStart = scanner.nextInt();
       // System.out.print("Enter the ending value for even numbers: ");
        int evenEnd = scanner.nextInt();
       // System.out.print("Enter the starting value for odd numbers: ");
        int oddStart = scanner.nextInt();
       // System.out.print("Enter the ending value for odd numbers: ");
        int oddEnd = scanner.nextInt();
        Thread evenThread = new Thread(new PrintNumbers(evenStart, evenEnd), "EvenThread");
        Thread oddThread = new Thread(new PrintNumbers(oddStart, oddEnd), "OddThread");

        evenThread.start();
try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        oddThread.start();
        scanner.close();
    }
}
```

```
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
class PrintNumbers implements Runnable {
private int start;
    private int end;
    public PrintNumbers(int start, int end) {
        this.start = start;
        this.end = end;
    }
    @Override
    public void run() {
        for (int i = start; i <= end; i += 2) {
            System.out.println(Thread.currentThread().getName() + ": " + i);
        }
    }
}
class W07_P4{
    //Code to create two threads, one printing even numbers and the other printing odd numbers
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        //System.out.print("Enter the starting value for even numbers: ");
        int evenStart = scanner.nextInt();
       // System.out.print("Enter the ending value for even numbers: ");
        int evenEnd = scanner.nextInt();
       // System.out.print("Enter the starting value for odd numbers: ");
        int oddStart = scanner.nextInt();
       // System.out.print("Enter the ending value for odd numbers: ");
        int oddEnd = scanner.nextInt();
        Thread evenThread = new Thread(new PrintNumbers(evenStart, evenEnd), "EvenThread");
        Thread oddThread = new Thread(new PrintNumbers(oddStart, oddEnd), "OddThread");

        evenThread.start();
try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        oddThread.start();
        scanner.close();
    }
}
```

Week 07 : Programming Assignment 5

Due on 2025-09-11, 23:59 IST

Implement a Simple Password Validator

In this task, you need to implement a **password validation system** using Java. The goal is to check if a given password meets the following conditions:

1. **Minimum Length Requirement**: The password must be at least **8 characters long**.
2. **Uppercase Letter Requirement**: The password must contain **at least one uppercase letter (A-Z)**.
3. **Number Requirement**: The password must contain **at least one numeric digit (0-9)**.

If the password meets **all three conditions**, print "Valid Password". Otherwise, print "Invalid Password".

Input Format:

- A **single string** representing the password (can contain alphabets, numbers, and special characters).

Output Format:

- Print "Valid Password" if the password satisfies all the conditions.
- Otherwise, print "Invalid Password".

Example Input:

Password123

Example Output:

Valid Password

Sample Test Cases

	Input	Output
Test Case 1	`password123`	`Invalid Password`
Test Case 2	`Password123`	`Valid Password`

The due date for submitting this assignment has passed.

0 out of 0 tests passed.

You scored 0.0/100.

Assignment submitted on 2025-09-10, 23:00 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W07_P5 {
    private String password;

    // Step 1: Constructor to initialize the password variable
    public W07_P5(String password) {
        this.password = password; // Assign the passed password to the instance variable
    }
    // ================================
    // Note: Try solving it without hints first—only check if you're truly stuck.
    // Avoid AI or internet searches; quick answers won't build real skills.
    // Struggle a bit, learn for life! Be honest with yourself!
    //
public boolean isValidPassword() {
        // Check minimum length
        if (password.length() < 8) {
            return false;
        }

        boolean hasUppercase = false;
        boolean hasDigit = false;

        // Loop through each character
        for (char ch : password.toCharArray()) {
            if (Character.isUpperCase(ch)) {
                hasUppercase = true;
            }
            if (Character.isDigit(ch)) {
                hasDigit = true;
            }
        }

        // Return true only if all conditions are met
        return hasUppercase && hasDigit;
    }
public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Read password input from user
        String inputPassword = scanner.nextLine();
        scanner.close();
        W07_P5 validator = new W07_P5(inputPassword);

        // Check password validity and print result
        if (validator.isValidPassword(inputPassword)) {
            System.out.print("Valid Password");
        } else {
            System.out.print("Invalid Password");
        }

        scanner.close();
    }
  }
```

Sample solutions (Provided by instructor)

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

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W07_P5 {
    private String password;

    // Step 1: Constructor to initialize the password variable
    public W07_P5(String password) {
        this.password = password; // Assign the passed password to the instance variable
    }
    // ================================
    // Note: Try solving it without hints first-only check if you're truly stuck.
    // Avoid AI or internet searches; quick answers won't build real skills.
    // Struggle a bit, learn for life! Be honest with yourself!
    //
public boolean isValidPassword(String password) {
        // Step 1: Check if the password length is at least 8 characters
        if (this.password.length() < 8) {
            return false; // If password is too short, it's invalid
        }

        boolean hasUpperCase = false; // Flag to track if there is an uppercase letter
        boolean hasDigit = false; // Flag to track if there is a number

        // Step 2: Loop through each character in the password
        for (char ch : this.password.toCharArray()) {
            if (Character.isUpperCase(ch)) {
                hasUpperCase = true; // Found an uppercase letter
            }
            if (Character.isDigit(ch)) {
                hasDigit = true; // Found a number
            }

            // If both conditions are met, no need to check further
            if (hasUpperCase && hasDigit) {
                return true;
            }
        }

        // Step 3: If either condition is not met, return false
        return false;
    }
public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // Read password input from user
        String inputPassword = scanner.nextLine();
        scanner.close();
        W07_P5 validator = new W07_P5(inputPassword);

        // Check password validity and print result
        if (validator.isValidPassword(inputPassword)) {
            System.out.print("Valid Password");
        } else {
            System.out.print("Invalid Password");
        }

        scanner.close();
    }
  }
```

W08 Programming Assignments 1

Due on 2025-09-18, 23:59 IST

Creating a Thread using Thread Class

Problem Statement

In Java, you can run multiple tasks at the same time using **Multithreading**.
The simplest way to create a thread is by extending the built-in `Thread` class.

What is a Thread?

- A thread is a small unit of a program that runs independently
- Multiple threads can run in parallel, improving efficiency

Programming Assignment:

- Create a class called `MyThread` that extends `Thread`
- In its `run()` method, print `"Thread is running"`
- In the `main` method, create an object of `MyThread` and start the thread

This helps you understand the basic way to create and start a thread in Java.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		Thread is running	Thread is running\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-18, 22:16 IST

Your last recorded submission was :

```java
public class W08_P1 {

    // Create a class that extends Thread
    static class MyThread extends Thread {

        @Override
        public void run() {
            // Print message when thread is running
            System.out.println("Thread is running");

}
    }

    public static void main(String[] args) {

        // Create object of MyThread
        MyThread t = new MyThread();

        // Start the thread
        t.start();
    }
}
```

Sample solutions (Provided by instructor)

```java
public class W08_P1 {

    // Create a class that extends Thread
    static class MyThread extends Thread {

        @Override
        public void run() {
System.out.println("Thread is running"); // Message printed by thread when run method executes

/*
Explanation:
- We extend Thread class to create MyThread
- run() method contains the code that the thread executes
- start() method begins execution in a new thread
- This demonstrates the simplest form of multithreading
*/
}
    }

    public static void main(String[] args) {

        // Create object of MyThread
        MyThread t = new MyThread();

        // Start the thread
        t.start();
    }
```

W08 Programming Assignments 2

Due on 2025-09-18, 23:59 IST

Creating a Thread using Runnable Interface

Problem Statement

In Java, another common way to create threads is by implementing the `Runnable` interface.

What is Runnable?

- Runnable is an interface with a single method called `run()`
- You can pass a Runnable object to a Thread and start the thread

Why use Runnable?

- It allows your class to extend another class, as Java supports single inheritance
- It provides flexibility in thread creation

Programming Assignment:

- Create a class called `MyRunnable` that implements `Runnable`
- In its `run()` method, print `"Runnable thread is running"`
- In the `main` method, create a `Thread` object using `MyRunnable` and start the thread

This demonstrates thread creation using the Runnable interface.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		`Runnable thread is running`	`Runnable thread is running\n`	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-18, 22:17 IST

Your last recorded submission was :

```java
public class W08_P2 {

    // Create a class that implements Runnable interface
    static class MyRunnable implements Runnable {

        @Override
        public void run() {
System.out.println("Runnable thread is running");
}
    }

    public static void main(String[] args) {

        // Create object of MyRunnable
        MyRunnable r = new MyRunnable();

        // Create Thread using Runnable object
        Thread t = new Thread(r);

        // Start the thread
        t.start();
    }
}
```

Sample solutions (Provided by instructor)

```java
public class W08_P2 {

    // Create a class that implements Runnable interface
    static class MyRunnable implements Runnable {

        @Override
        public void run() {
System.out.println("Runnable thread is running"); // Message printed by thread when run method executes

/*
Explanation:
- Runnable interface provides run() method
- Thread object is created by passing Runnable object
- start() begins thread execution
- This approach allows flexibility, as class is not restricted to only extending Thread
*/
}
    }

    public static void main(String[] args) {

        // Create object of MyRunnable
        MyRunnable r = new MyRunnable();

        // Create Thread using Runnable object
        Thread t = new Thread(r);

        // Start the thread
        t.start();
    }
}
```

W08 Programming Assignments 3

Due on 2025-09-18, 23:59 IST

Programming Assignment: Understanding Basic Thread States in Java

Problem Statement

When a thread runs in Java, it moves through different stages called **states**.

What are Thread States?
Think of a thread like a person:

- It starts in one state
- Moves to another as work happens
- Finally, it finishes

For beginners, focus on these three simple states:

1. **New** – The thread is created but not started yet
2. **Running** – The thread is doing its work
3. **Terminated** – The thread has finished its work

Programming Assignment:

- Create a class called `MyThread` that extends `Thread`
- Inside its `run()` method, print `"Thread is running"`
- In the `main` method:
  - Create a `MyThread` object
  - Print `"Thread state before start"`
  - Start the thread
  - Print `"Thread state after start"`
  - Wait for thread to finish using `join()`
  - Print `"Thread state after completion"`

This shows how thread state changes as the thread runs.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		Thread state before start\n Thread state after start\n Thread is running\n Thread state after completion	Thread state before start\n Thread state after start\n Thread is running\n Thread state after completion\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-18, 22:26 IST
Your last recorded submission was :

```java
public class W08_P3 {

    // Create a class that extends Thread
    static class MyThread extends Thread {

        @Override
        public void run() {
System.out.println("Thread is running");
}
    }

    public static void main(String[] args) {

        // Create thread object
        MyThread t = new MyThread();

        // Thread is created but not started yet
        System.out.println("Thread state before start");

        // Start thread
        t.start();

        // Thread has started running
        System.out.println("Thread state after start");

        try {
            // Wait for thread to finish
            t.join();
        } catch (InterruptedException e) {
            // Not needed for beginners, but required to handle possible interruptions
        }

        // Thread has finished
        System.out.println("Thread state after completion");
    }
}
```

Assignment submitted on 2025-09-18, 22:26 IST

Your last recorded submission was :

```
public class W08_P3 {

    // Create a class that extends Thread
    static class MyThread extends Thread {

        @Override
        public void run() {
System.out.println("Thread is running");
}
    }

    public static void main(String[] args) {

        // Create thread object
        MyThread t = new MyThread();

        // Thread is created but not started yet
        System.out.println("Thread state before start");

        // Start thread
        t.start();

        // Thread has started running
        System.out.println("Thread state after start");

        try {
            // Wait for thread to finish
            t.join();
        } catch (InterruptedException e) {
            // Not needed for beginners, but required to handle possible interruptions
        }

        // Thread has finished
        System.out.println("Thread state after completion");
    }
}
```

Sample solutions (Provided by instructor)

```
public class W08_P3 {

    // Create a class that extends Thread
    static class MyThread extends Thread {

        @Override
        public void run() {
System.out.println("Thread is running"); // Message printed when thread starts work

/*
Explanation:
- When thread object is created, it is in New state
- When start() is called, thread runs in parallel
- join() waits for thread to finish work
- Final print confirms thread has finished
- This gives beginners a simple understanding of thread states
*/
}
    }

    public static void main(String[] args) {

        // Create thread object
        MyThread t = new MyThread();

        // Thread is created but not started yet
        System.out.println("Thread state before start");

        // Start thread
        t.start();

        // Thread has started running
        System.out.println("Thread state after start");

        try {
            // Wait for thread to finish
            t.join();
        } catch (InterruptedException e) {
            // Not needed for beginners, but required to handle possible interruptions
        }

        // Thread has finished
        System.out.println("Thread state after completion");
    }
}
```

W08 Programming Assignments 4

Due on 2025-09-18, 23:59 IST

Understanding Thread Priority in Java

Problem Statement

In Java, each thread has a **priority**, a number from 1 (lowest) to 10 (highest).
Priority suggests how important a thread is, though actual scheduling depends on the system.

Programming Assignment:

- Create a class `MyThread` that extends `Thread`
- In the main method:
  - Create a `MyThread` object
  - Set its priority to `8`
  - Start the thread
  - Print the thread's priority after setting

No output should come from the thread's `run()` method to avoid output mismatch.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		`Thread priority is: 8`	`Thread priority is: 8\n`	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-18, 22:29 IST

Your last recorded submission was :

```java
public class W08_P4 {

    // Thread class
    static class MyThread extends Thread {

        @Override
        public void run() {
            // No output here to keep portal testing consistent
        }
    }

    public static void main(String[] args) {

        MyThread t = new MyThread();

        // Set thread priority
        t.setPriority(8);

        // Start thread
        t.start();
// TODO: Print priority

  System.out.println("Thread priority is: " + t.getPriority());
            /*
             Hint:
             Use etPriority() to print the priority
             */
}
}
```

Sample solutions (Provided by instructor)

```java
public class W08_P4 {

    // Thread class
    static class MyThread extends Thread {

        @Override
        public void run() {
            // No output here to keep portal testing consistent
        }
    }

    public static void main(String[] args) {

        MyThread t = new MyThread();

        // Set thread priority
        t.setPriority(8);

        // Start thread
        t.start();
// Print priority

 System.out.println("Thread priority is: " + t.getPriority());
}
}
```

W08 Programming Assignments 5

Due on 2025-09-18, 23:59 IST

Programming Assignment: Introduction to Thread Synchronization

Problem Statement

What is a Thread?
Imagine your computer doing many tasks at once — for example:

- Playing music
- Downloading files
- Browsing the internet

In Java, each small task that runs independently is called a **Thread**.
Threads help programs run faster by working at the same time.

Why Synchronization?
When multiple threads work on the same thing together, they may interfere with each other.
For example:

- Two threads try to update the same number at the same time
- The final result may be wrong

What is Synchronization?

- It is like putting a lock
- Only one thread can work on the shared thing at a time
- This prevents problems caused by threads disturbing each other

---

Programming Assignment:

- Create a class `Counter` with a number `count` starting from 0
- Write a method `increment()` to increase the number by 1, using `synchronized` keyword
- Create a thread class called `MyThread` that runs the `increment()` method 1000 times
- In `main`, run two threads to increase the number
- After both threads finish, print the final count

This shows how to use synchronization to avoid problems when multiple threads share data.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		`Final count is: 2000`	`Final count is: 2000\n`	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-18, 22:33 IST
Your last recorded submission was :

```java
public class W08_P5 {

    // Shared class with a number
    static class Counter {
        int count = 0;

        // Synchronized method to safely increase number
        public synchronized void increment() {
            count++;
        }
    }

    // Thread class to run increment
    static class MyThread extends Thread {
        Counter c;

        MyThread(Counter c) {
            this.c = c;
        }

        @Override
        public void run() {
            // Call increment() 1000 times using loop
            for (int i = 0; i < 1000; i++) {
                c.increment();
            }

}
    }

    public static void main(String[] args) {
```

Assignment submitted on 2025-09-18, 22:33 IST

Your last recorded submission was :

```java
public class W08_P5 {

    // Shared class with a number
    static class Counter {
        int count = 0;

        // Synchronized method to safely increase number
        public synchronized void increment() {
            count++;
        }
    }

    // Thread class to run increment
    static class MyThread extends Thread {
        Counter c;

        MyThread(Counter c) {
            this.c = c;
        }

        @Override
        public void run() {
            // Call increment() 1000 times using loop
            for (int i = 0; i < 1000; i++) {
                c.increment();
            }

}
    }

    public static void main(String[] args) {

        Counter c = new Counter();

        // Create two threads
        MyThread t1 = new MyThread(c);
        MyThread t2 = new MyThread(c);

        // Start both threads
        t1.start();
        t2.start();

        try {
            // Wait for both threads to finish
            t1.join();
            t2.join();
        } catch (InterruptedException e) {
        }

        // Print final count
        System.out.println("Final count is: " + c.count);
    }
}
```

Sample solutions (Provided by instructor)

```java
public class W08_P5 {

    // Shared class with a number
    static class Counter {
        int count = 0;

        // Synchronized method to safely increase number
        public synchronized void increment() {
            count++;
        }
    }

    // Thread class to run increment
    static class MyThread extends Thread {
        Counter c;

        MyThread(Counter c) {
            this.c = c;
        }

        @Override
        public void run() {
for (int i = 0; i < 1000; i++) {
    c.increment(); // Safely increase the count inside synchronized method
}

/*
Explanation:
- Threads run at the same time, so synchronization protects shared number
- Without synchronization, result may be less than 2000 due to interference
- synchronized keyword ensures only one thread updates the number at a time
*/
}
    }

    public static void main(String[] args) {

        Counter c = new Counter();

        // Create two threads
        MyThread t1 = new MyThread(c);
        MyThread t2 = new MyThread(c);

        // Start both threads
        t1.start();
        t2.start();

        try {
            // Wait for both threads to finish
            t1.join();
            t2.join();
        } catch (InterruptedException e) {
        }

        // Print final count
```

Week 09 : Programming Assignment 1

Due on 2025-09-25, 23:59 IST

Write suitable code to develop **a 2D Flip-Flop Array with dimension 5 × 5, which replaces all input elements with values 0 by 1 and 1 by 0.** An example is shown below:

INPUT:

```
00001
00001
00001
00001
00001
```

OUTPUT:

```
11110
11110
11110
11110
11110
```

Note the following points carefully:

1. Here, the input must contain only 0 and 1.
2. The input and output array size must be of dimension 5 × 5.
3. Flip-Flop: If 0 then 1 and vice-versa.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	00100 01010 10001 01010 00100	11011\n 10101\n 01110\n 10101\n 11011	11011\n 10101\n 01110\n 10101\n 11011\n	Passed

The due date for submitting this assignment has passed.

1 out of 1 tests passed.

You scored 100.0/100.

Assignment submitted on 2025-09-24, 23:49 IST

Your last recorded submission was :

```java
import java.util.Scanner;
public class W09_P1{
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
// Declare the 5X5 2D array to store the input
        int[][] matrix = new int[5][5];

        // Input 2D Array using Scanner Class and check data validity
        // This loop reads 5 lines of input (e.g., "00001")
        for (int i = 0; i < 5; i++) {
            String row = sc.next(); // Reads the input string for the current row
            // This inner loop iterates through the characters of the string
            for (int j = 0; j < 5; j++) {
                // Convert the character '0' or '1' to an integer 0 or 1 and store it
                matrix[i][j] = Character.getNumericValue(row.charAt(j));
            }
        }

        // Perform the Flip-Flop Operation & Output the 2D Flip-Flop Array
        // This loop iterates through each row of the matrix
        for (int i = 0; i < 5; i++) {
            // This inner loop iterates through each element in the current row
            for (int j = 0; j < 5; j++) {
                // Check the value and print its opposite
                if (matrix[i][j] == 0) {
                    System.out.print(1);
                } else {
                    System.out.print(0);
                }
            }
            // Print a new line after each row is completed to format the output correctly
            System.out.println();
        }

} // The main() ends here
} // The main class ends here
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
public class W09_P1{
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
// Declare the 5X5 2D array to store the input
        char original[][]= new char[5][5];

        // Input 2D Array using Scanner Class and check data validity
        for(int line=0;line<5; line++){
            String input = sc.nextLine();
            char seq[] = input.toCharArray();
            if(seq.length==5){
                for(int i=0;i<5;i++){
                    if(seq[i]=='0' || seq[i]=='1'){
                        original[line][i]=seq[i];
                        if(line==4 && i==4)
                            flipflop(original);
                    }
                    else{
                        System.out.print("Only 0 and 1 supported.");
                        break;
                    }
```

```
            // This inner loop iterates through each element in the current row
            for (int j = 0; j < 5; j++) {
                // Check the value and print its opposite
                if (matrix[i][j] == 0) {
                    System.out.print(1);
                } else {
                    System.out.print(0);
                }
            }
            // Print a new line after each row is completed to format the output correctly
            System.out.println();
        }

} // The main() ends here
} // The main class ends here
```

Sample solutions (Provided by instructor)

```
import java.util.Scanner;
public class W09_P1{
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
// Declare the 5X5 2D array to store the input
        char original[][]= new char[5][5];

        // Input 2D Array using Scanner Class and check data validity
        for(int line=0;line<5; line++){
            String input = sc.nextLine();
            char seq[] = input.toCharArray();
            if(seq.length==5){
                for(int i=0;i<5;i++){
                    if(seq[i]=='0' || seq[i]=='1'){
                        original[line][i]=seq[i];
                        if(line==4 && i==4)
                            flipflop(original);
                    }
                    else{
                        System.out.print("Only 0 and 1 supported.");
                        break;
                    }
                }
            }else{
                System.out.print("Invalid length");
                break;
            }

        }
    }
    static void flipflop(char[][] flip){
        // Flip-Flop Operation
        for(int i=0; i<5;i++){
            for(int j=0; j<5;j++){
                if(flip[i][j]=='1')
                    flip[i][j]='0';
                else
                    flip[i][j]='1';
            }
        }

        // Output the 2D FlipFlopped Array
        for(int i=0; i<5;i++){
            for(int j=0; j<5;j++){
                System.out.print(flip[i][j]);
            }
            System.out.println();
        }
} // The main() ends here
} // The main class ends here
```

Week 09 : Programming Assignment 2

Due on 2025-09-25, 23:59 IST

Complete the code to **develop a BASIC CALCULATOR that can perform operations like *Addition*, *Subtraction*, *Multiplication* and *Division*.**

Note the following points carefully:
1. Use only double datatype to store calculated numeric values.
2. Assume input to be of integer datatype.
3. The output should be rounded using Math.round() method.
4. Take care of the spaces during formatting output (e.g., single space each before and after =).
5. The calculator should be able to perform required operations on a minimum of two operands as shown in the below example:

Input:

5+6

Output:

5+6 = 11

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	3+2	3+2 = 5	3 + 2 = 5\n	Wrong Answer

The due date for submitting this assignment has passed.
0 out of 1 tests passed.
You scored 0.0/100.

Assignment submitted on 2025-09-24, 23:58 IST

Your last recorded submission was :

```java
import java.util.Scanner;
public class W09_P2{
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        String input = sc.nextLine(); // Read as string, e.g., 5+6
double num1 = 0, num2 = 0, result = 0;
        char operator = ' ';
        int operatorIndex = -1;

        // Find which operator is in the string and get its position
        if (input.contains("+")) {
            operator = '+';
            operatorIndex = input.indexOf('+');
        } else if (input.contains("-")) {
            operator = '-';
            operatorIndex = input.indexOf('-');
        } else if (input.contains("*")) {
            operator = '*';
            operatorIndex = input.indexOf('*');
        } else if (input.contains("/")) {
            operator = '/';
            operatorIndex = input.indexOf('/');
        }

        // Use substring to get the parts before and after the operator
        // .trim() removes any accidental whitespace
        String num1Str = input.substring(0, operatorIndex).trim();
        String num2Str = input.substring(operatorIndex + 1).trim();

        // Convert the string numbers to doubles
        num1 = Double.parseDouble(num1Str);
        num2 = Double.parseDouble(num2Str);

        // Perform the correct calculation based on the operator
        switch (operator) {
            case '+':
                result = num1 + num2;
                break;
            case '-':
                result = num1 - num2;
                break;
            case '*':
                result = num1 * num2;
                break;
            case '/':
                result = num1 / num2;
                break;
        }

        // Round the result as required
        long roundedResult = Math.round(result);

        // Print the output in the exact format: num <space> op <space> num <space> = <space> result
        System.out.println((int)num1 + " " + operator + " " + (int)num2 + " = " + roundedResult);
} // The main() method ends here
} // The main class ends here
```

Sample solutions (Provided by instructor)

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
public class W09_P2{
    public static void main(String args[]){
        Scanner sc = new Scanner(System.in);
        String input = sc.nextLine(); // Read as string, e.g., 5+6
// Declare and initialize the required variable(s)
        int i=0;
        int j=0;
        double output=0;
        // Split the input string into character array
        char seq[] = input.toCharArray();
        /*
        Use some method to separate the two operands
        and then perform the required operation.
        */
        for(int a=0; a<seq.length; a++){
            if(seq[a]=='+'){
                i= Integer.parseInt(input.substring(0,a));
                j= Integer.parseInt(input.substring(a+1,seq.length));
                output = (double)i+j;
            }else if(seq[a]=='-'){
                i= Integer.parseInt(input.substring(0,a));
                j= Integer.parseInt(input.substring(a+1,seq.length));
                output = (double)i-j;
            }else if(seq[a]=='/'){
                i= Integer.parseInt(input.substring(0,a));
                j= Integer.parseInt(input.substring(a+1,seq.length));
                output = (double)i/j;
            }else if(seq[a]=='*'){
                i= Integer.parseInt(input.substring(0,a));
                j= Integer.parseInt(input.substring(a+1,seq.length));
                output = (double)i*j;
            }
        }
        // Print the output as stated in the question
        System.out.print(input+" = " + Math.round(output));
} // The main() method ends here
} // The main class ends here
```

Week 09 : Programming Assignment 3

Due on 2025-09-25, 23:59 IST

Write a Java program that utilizes multithreading to calculate and print the squares of numbers from a specified begin to a specified end.

The main method is already created.

You need to design a SquareThread class that has two members,

- int begin;
- int end;

Each thread should sequentially print the squares of numbers from begin to end (both inclusive).

The same code will be used to create another thread that prints the sqaure of numbers from end to begin in reverse order.

(if begin is greater than end, print the square of each number in reverse order first)

The main method will first call SquareThread with begin and end and then in reverse order.

The class you create should be able to handle such case and print as required in the correct order.

HINT: use the keyword `synchronized` in the run method.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	2 6	4\n 9\n 16\n 25\n 36\n 36\n 25\n 16\n 9\n 4	4\n 9\n 16\n 25\n 36\n 36\n 25\n 16\n 9\n 4\n	Passed
Test Case 2	5 5	25\n 25	25\n 25\n	Passed

The due date for submitting this assignment has passed.
2 out of 2 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-25, 02:05 IST

Your last recorded submission was :

```java
import java.util.Scanner;

class SquareThread extends Thread {
    private int begin;
    private int end;
   // Constructor to initialize the begin and end numbers
    public SquareThread(int begin, int end) {
        this.begin = begin;
        this.end = end;
    }

    // The core logic of the thread goes into the run() method.
    // The "synchronized" keyword ensures that only one thread can execute this method at a time,
    // preventing the output from thread1 and thread2 from getting interleaved.
    @Override
    public synchronized void run() {
        // This block handles the reverse order printing (e.g., from 5 down to 1)
        if (begin > end) {
            for (int i = begin; i >= end; i--) {
                System.out.println(i * i);
            }
        }
        // This block handles the forward order printing (e.g., from 1 up to 5)
        else {
            for (int i = begin; i <= end; i++) {
                System.out.println(i * i);
            }
        }
    }
}

public class W09_P3 {
    public static void main(String args[]) {
        Scanner scanner = new Scanner(System.in);
        //System.out.print("Enter the begin for square calculation: ");
        int begin = scanner.nextInt();
        //System.out.print("Enter the end for square calculation: ");
        int end = scanner.nextInt();
        scanner.close();

        SquareThread thread1 = new SquareThread(begin, end);
        SquareThread thread2 = new SquareThread(end, begin);

        thread1.start();
        thread2.start();
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

class SquareThread extends Thread {
    private int begin;
    private int end;
public SquareThread(int begin,int end) {
        this.begin = begin;
        this.end = end;
    }

    synchronized public void run() {
        if (begin > end) {
            for (int i = begin; i >= end; i--) {
                System.out.println(i * i);
            }
        }
        else{
            for (int i = begin; i <= end; i++) {
                System.out.println(i * i);
            }
        }
    }
}

public class W09_P3 {
    public static void main(String args[]) {
        Scanner scanner = new Scanner(System.in);
        //System.out.print("Enter the begin for square calculation: ");
        int begin = scanner.nextInt();
        //System.out.print("Enter the end for square calculation: ");
        int end = scanner.nextInt();
        scanner.close();

        SquareThread thread1 = new SquareThread(begin, end);
        SquareThread thread2 = new SquareThread(end, begin);

        thread1.start();
        thread2.start();
    }
}
```

Week 09 : Programming Assignment 4

Due on 2025-09-25, 23:59 IST

Complete the code segment to catch the exception in the following, if any.
On the occurrence of such an exception, your program should print
“Please enter valid data” .
If there is no such exception, it will print the square of the number entered.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	1.0	Please enter valid data	Please enter valid data\n	Passed
Test Case 2	6	36	36\n	Passed

The due date for submitting this assignment has passed.
2 out of 2 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-25, 17:28 IST

Your last recorded submission was :

```
import java.io.*;
class W09_P4{
        public static void main(String args[]){
   try {
            // This code might throw an exception
            InputStreamReader r = new InputStreamReader(System.in);
            BufferedReader br = new BufferedReader(r);
            String number = br.readLine();
            int x = Integer.parseInt(number);
            System.out.println(x * x);
        } catch (Exception e) {
            // This code runs only if an exception was caught
            System.out.println("Please enter valid data");
        }
} // The main() ends here
} // The main class ends here
```

Sample solutions (Provided by instructor)

```
import java.io.*;
class W09_P4{
        public static void main(String args[]){
try {
            InputStreamReader r = new InputStreamReader(System.in);
            BufferedReader br = new BufferedReader(r);
            String number = br.readLine();
            int x = Integer.parseInt(number);
            System.out.println(x * x);
        } catch (Exception e) {
            System.out.println("Please enter valid data");
        }
} // The main() ends here
} // The main class ends here
```

Week 09 : Programming Assignment 5

Due on 2025-09-25, 23:59 IST

Define a class **Point** with members

- private double x;
- private double y;

and methods:

- public Point(double x, double y){} // Constructor to create a new point?
- public double distance(Point p2){} // Function to return the distance of this Point from another Point

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1	0 0 0 0	0.0	0.0	Passed
Test Case 2	5.5 5.0 5.0 5.5	0.7071067811865476	0.7071067811865476	Passed

The due date for submitting this assignment has passed.
2 out of 2 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-09-25, 17:30 IST

Your last recorded submission was :

```java
import java.util.Scanner;

public class W09_P5{

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        double x1 = sc.nextDouble();
        double y1 = sc.nextDouble();
        double x2 = sc.nextDouble();
        double y2 = sc.nextDouble();
        Point p1 = new Point(x1, y1);
        Point p2 = new Point(x2, y2);

        System.out.print(p1.distance(p2));
    }

}
//Complete the code segment to define a class Point with parameter x,y and method distance()for calculating distance
// Note: Pass objectsof type class Point as argument in distance() method.

class Point {
    /**
     * Private member variables to store the coordinates.
     */
    private double x;
    private double y;

    /**
     * Constructor to create a new point with given coordinates.
     * The 'this' keyword is used to distinguish between the class members and the parameters.
     * @param x The x-coordinate.
     * @param y The y-coordinate.
     */
    public Point(double x, double y) {
        this.x = x;
        this.y = y;
    }

    /**
     * Function to return the distance of this Point from another Point.
     * @param p2 The other Point object.
     * @return The calculated distance as a double.
     */
    public double distance(Point p2) {
        // Calculate the difference in the x and y coordinates.
        double deltaX = this.x - p2.x;
        double deltaY = this.y - p2.y;

        // Use the Pythagorean theorem to find the distance.
        // Math.sqrt() calculates the square root.
        return Math.sqrt(deltaX * deltaX + deltaY * deltaY);
    }
}
```

Sample solutions (Provided by instructor)

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

```java
import java.util.Scanner;

public class W09_P5{

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        double x1 = sc.nextDouble();
        double y1 = sc.nextDouble();
        double x2 = sc.nextDouble();
        double y2 = sc.nextDouble();
        Point p1 = new Point(x1, y1);
        Point p2 = new Point(x2, y2);

        System.out.print(p1.distance(p2));
    }

}
class Point{
    private double x;
    private double y;
    public Point(double x, double y){
        this.x = x;
        this.y = y;
    }
    public double distance(Point p2){
        double dist;
        dist = Math.sqrt(Math.pow((p2.x-x),2) + Math.pow((p2.y-y),2));
        return dist;
    }
}
```

W10 Programming Assignments 1

Due on 2025-10-02, 23:59 IST

Introduction to JDBC and Required Imports

Problem Statement

What is JDBC?
JDBC means Java Database Connectivity, which allows Java programs to work with databases.

Before writing database programs, you must import the correct packages. Without proper imports, the code will not compile.

Programming Assignment:

- Import the necessary JDBC packages so the program can compile
- You only need to complete the import section
- The output `"import successful"` is printed automatically

This helps you practice the correct way to prepare Java programs for database work.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		import successful	import successful\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-10-01, 22:47 IST

Your last recorded submission was :

```java
import java.sql.Connection;
import java.sql.DriverManager;
public class W10_P1 {
    public static void main(String[] args) {

        // The below code requires correct imports to compile
        Connection con = null;
        DriverManager dm = null;

        // Output statement, for portal testing only
        System.out.println("import successful");
    }
}
```

Sample solutions (Provided by instructor)

```java
import java.sql.Connection;
import java.sql.DriverManager;

/*
Explanation:
- The Connection class represents the link between your program and the database
- The DriverManager class helps to create that connection
- Both are present in java.sql package, so we import them
- This task prepares you to write database programs by importing correctly
*/
public class W10_P1 {
    public static void main(String[] args) {

        // The below code requires correct imports to compile
        Connection con = null;
        DriverManager dm = null;

        // Output statement, for portal testing only
        System.out.println("import successful");
    }
}
```

W10 Programming Assignments 2

Due on 2025-10-02, 23:59 IST

Writing a JDBC Connection String

Problem Statement

What is a JDBC Connection String?
To connect a Java program to a database, we use a special sentence called a **connection string**.
It tells Java:

- Which database to use
- Where the database is located

In this assignment, you will practice writing the correct JDBC connection string for SQLite database.

Programming Assignment:

- Write the correct connection string for an SQLite database called `test.db`
- The rest of the program prints `"connection string ready"` if your string is correct

This helps beginners practice writing JDBC connection strings safely, without actual database access.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		connection string ready	connection string ready\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-10-01, 22:48 IST
Your last recorded submission was :

```java
public class W10_P2 {
    public static void main(String[] args) {
// TODO: Complete the connection string for SQLite database

        /*
         Hint:
         JDBC connection string for SQLite starts with "jdbc:sqlite:"
         The database file is "test.db"
         */
String url = "jdbc:sqlite:test.db";
// Portal test output
        if (url.equals("jdbc:sqlite:test.db")) {
            System.out.println("connection string ready");
        } else {
            System.out.println("incorrect connection string");
        }
    }
}
```

Sample solutions (Provided by instructor)

```java
public class W10_P2 {
    public static void main(String[] args) {
String url = "jdbc:sqlite:test.db"; // Correct connection string for SQLite

/*
Explanation:
- All SQLite JDBC connections start with "jdbc:sqlite:"
- The database file name comes after the colon
- This prepares your code to work with SQLite database
- No real database connection is made in this task
*/
// Portal test output
        if (url.equals("jdbc:sqlite:test.db")) {
            System.out.println("connection string ready");
        } else {
            System.out.println("incorrect connection string");
        }
    }
}
```

W10 Programming Assignments 3

Due on 2025-10-02, 23:59 IST

Writing a Simple SQL Insert Statement

Problem Statement

What is an SQL Insert Statement?
When working with databases, we use the **INSERT** command to add new records (data) into a table.

In this assignment:

- Imagine there is a table called `students` with two columns: `id` and `name`
- You will practice writing the correct SQL insert statement as a text string
- No actual database operation is performed, only the string is checked

Programming Assignment:

- Complete the SQL insert statement to add a student with id `1` and name `'Alice'`
- The rest of the code checks your string and prints `"insert statement ready"` if correct

This task helps beginners practice writing SQL commands safely.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		insert statement ready	insert statement ready\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-10-01, 22:50 IST
Your last recorded submission was :

```java
public class W10_P3 {
    public static void main(String[] args) {
// TODO: Complete the SQL insert statement as a string

        /*
         Hint:
         SQL insert format: INSERT INTO tablename (columns) VALUES (values);
         For this task:
         - Table name is students
         - Columns are id and name
         - Values are 1 and 'Alice'
         */

                String sql = "INSERT INTO students (id, name) VALUES (1, 'Alice');";

// Portal test output
        if (sql.equals("INSERT INTO students (id, name) VALUES (1, 'Alice');")) {
            System.out.println("insert statement ready");
        } else {
            System.out.println("incorrect statement");
        }
    }
}
```

Sample solutions (Provided by instructor)

```java
public class W10_P3 {
    public static void main(String[] args) {
String sql = "INSERT INTO students (id, name) VALUES (1, 'Alice');"; // Correct SQL insert statement

/*
Explanation:
- INSERT INTO specifies the table
- Column names are given in brackets
- VALUES keyword provides the data to insert
- This prepares students to work with databases using SQL commands
*/
// Portal test output
        if (sql.equals("INSERT INTO students (id, name) VALUES (1, 'Alice');")) {
            System.out.println("insert statement ready");
        } else {
            System.out.println("incorrect statement");
        }
    }
}
```

Writing a Simple SQL SELECT Statement

Problem Statement

What is a SQL SELECT Statement?
The `SELECT` statement is used to get data from a database table.
You can choose to fetch all columns or specific columns.

In this assignment:

- Imagine a table called `students` with two columns: `id` and `name`
- You will practice writing a SQL statement to fetch all columns for all students

Programming Assignment:

- Write a SQL SELECT statement as a text string to get all data from `students` table
- The rest of the program checks your string and prints `"select statement ready"` if correct

This task helps beginners practice safe SQL reading commands.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		select statement ready	select statement ready\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-10-01, 22:51 IST

Your last recorded submission was :

```java
public class W10_P4 {
    public static void main(String[] args) {
// TODO: Complete the SQL select statement as a string

        /*
         Hint:
         To fetch all columns from a table:
         SELECT * FROM tablename;
         For this task:
         - Table name is students
         */

        String sql ="SELECT * FROM students;";
if (sql.equals("SELECT * FROM students;")) {
            System.out.println("select statement ready");
        } else {
            System.out.println("incorrect statement");
        }
    }
}
```

Sample solutions (Provided by instructor)

```java
public class W10_P4 {
    public static void main(String[] args) {
String sql = "SELECT * FROM students;"; // Correct SQL select statement

/*
Explanation:
- SELECT * means fetch all columns
- FROM students specifies the table
- This is the simplest form of reading data using SQL
*/
if (sql.equals("SELECT * FROM students;")) {
            System.out.println("select statement ready");
        } else {
            System.out.println("incorrect statement");
        }
    }
}
```

W10 Programming Assignments 5

Due on 2025-10-02, 23:59 IST

Writing a Simple SQL UPDATE Statement

Problem Statement

What is an SQL UPDATE Statement?
The `UPDATE` command is used to change existing data in a database table.

In this assignment:

- Imagine a table called `students` with two columns: `id` and `name`
- You will write a SQL statement to update the name of the student with id `1` to `'Bob'`

Programming Assignment:

- Write the correct SQL UPDATE statement as a text string
- The rest of the code checks your string and prints `"update statement ready"` if correct

This helps beginners practice safe SQL update syntax.

Private Test cases used for evaluation	Input	Expected Output	Actual Output	Status
Test Case 1		update statement ready	update statement ready\n	Passed

The due date for submitting this assignment has passed.
1 out of 1 tests passed.
You scored 100.0/100.

Assignment submitted on 2025-10-01, 22:56 IST

Your last recorded submission was :

```java
public class W10_P5 {
    public static void main(String[] args) {
// TODO: Complete the SQL update statement as a string

        /*
         Hint:
         UPDATE tablename SET column = value WHERE condition;
         For this task:
         - Table name is students
         - Update name to 'Bob'
         - Condition is id = 1
         */

        String sql = "UPDATE students SET name = 'Bob' WHERE id = 1;";
// Portal test output
        if (sql.equals("UPDATE students SET name = 'Bob' WHERE id = 1;")) {
            System.out.println("update statement ready");
        } else {
            System.out.println("incorrect statement");
        }
    }
}
```

Sample solutions (Provided by instructor)

```java
public class W10_P5 {
    public static void main(String[] args) {
String sql = "UPDATE students SET name = 'Bob' WHERE id = 1;"; // Correct SQL update statement

/*
Explanation:
- UPDATE specifies the table
- SET assigns new value to column
- WHERE filters which record to update
- This task teaches basic SQL update syntax safely
*/
// Portal test output
        if (sql.equals("UPDATE students SET name = 'Bob' WHERE id = 1;")) {
            System.out.println("update statement ready");
        } else {
            System.out.println("incorrect statement");
        }
    }
}
```

Week 11 : Programming Assignment 1

Due on 2025-10-09, 23:59 IST

The following code needs some package to work properly.

Write appropriate code to

- import the required package(s) in order to make the program compile and execute successfully.

(Ignore the hidden code)

Sample Test Cases

	Input	Output
Test Case 1	1	truetrue
Test Case 2	0	true

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;
import java.sql.*;
public class W11_P1 {
    public static void main(String args[]) {
        try {
            Connection conn = null;
            Statement stmt = null;
            String DB_URL = "jdbc:sqlite:/tempfs/db";
            System.setProperty("org.sqlite.tmpdir", "/tempfs");
// JDBC Codes in the hidden section

            // Open a connection
            conn = DriverManager.getConnection(DB_URL);
            System.out.print(conn.isValid(1));
            // conn.close();
            // take input from the user
            Scanner sc = new Scanner(System.in);
            int s = sc.nextInt();
            if (s == 1) {
                //conn.close();
                System.out.print(conn.isValid(1));
            }
            // Close the connection
            conn.close();
            // JDBC Codes in the visible section
} catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

Week 11 : Programming Assignment 2

Due on 2025-10-09, 23:

Write the JDBC codes needed to create a Connection interface using the DriverManager class and the variable DB_URL. Che whether the connection is successful using 'isValid(timeout)' method to generate the output, which is either 'true' or 'false'.

Note the following points carefully:

- Name the connection object as conn only.
- Use timeout value as 1.

(Ignore hidden code)

Sample Test Cases

	Input	Output
Test Case 1	7	true
Test Case 2	0	true

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.
Sample solutions (Provided by instructor)

```java
import java.sql.*;
import java.util.Scanner;
public class W11_P2 {
    public static void main(String args[]) {
        try {
              Connection conn = null;
              Statement stmt = null;
              String DB_URL = "jdbc:sqlite:/tempfs/db";
              System.setProperty("org.sqlite.tmpdir", "/tempfs");
conn = DriverManager.getConnection(DB_URL);
System.out.print(conn.isValid(1));
// Private test case
Scanner sc = new Scanner(System.in);
int s = sc.nextInt();
if (s == 7) {
    // conn.close();
    //System.out.print();
}
conn.close();
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

Week 11 : Programming Assignment 3

Due on 2025-10-09, 23:

Due to some mistakes in the below code, the code is not compiled/executable.

Modify and debug the JDBC code to make it execute successfully.

Sample Test Cases

	Input	Output
Test Case 1	6	truefalse
Test Case 2	0	true

The due date for submitting this assignment has passed.

As per our records you have not submitted this assignment.

Sample solutions (Provided by instructor)

```java
import java.sql.*; // All sql classes are imported
import java.util.Scanner; // Semicolon is added

public class W11_P3 {
    public static void main(String args[]) {
        try {
            Connection conn = null;
            Statement stmt = null;
            String DB_URL = "jdbc:sqlite:/tempfs/db";
            System.setProperty("org.sqlite.tmpdir", "/tempfs");
            // Connection object is created
            conn = DriverManager.getConnection(DB_URL);
            conn.close(); // variable is fixed
            System.out.print(conn.isClosed());
Scanner sc = new Scanner(System.in);
            int s = sc.nextInt();
            if (s == 6) {
                System.out.print(false);
            }
            sc.close();
} catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

Week 11 : Programming Assignment 4

Due on 2025-10-09, 23:59 IST

Complete the code segment **to create a new table named '*STUDENTS*' in SQL database using the following information.**

Column	UID	Name	Roll	Age
Type	Integer	Varchar (45)	Varchar (12)	Integer

Sample Test Cases

	Input	Output
Test Case 1	2	No. of columns : 4 Column 1 Name: UID Column 1 Type : INT Column 2 Name: Name Column 2 Type : VARCHAR Column 3 Name: Roll Column 3 Type : VARCHAR Column 4 Name: Age Column 5 Type : INT
Test Case 2	1	No. of columns : 4 Column 1 Name: UID Column 1 Type : INT Column 2 Name: Name Column 2 Type : VARCHAR Column 3 Name: Roll Column 3 Type : VARCHAR Column 4 Name: Age Column 5 Type : INT

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.

Sample solutions (Provided by instructor)

```java
import java.sql.*;
import java.lang.*;
public class W11_P4 {
    public static void main(String args[]) {
        try {
              Connection conn = null;
              Statement stmt = null;
              String DB_URL = "jdbc:sqlite:/tempfs/db";
              System.setProperty("org.sqlite.tmpdir", "/tempfs");

              // Open a connection
              conn = DriverManager.getConnection(DB_URL);
              stmt = conn.createStatement();
// The statement containing SQL command to create table "STUDENTS"
String CREATE_TABLE_SQL="CREATE TABLE STUDENTS (UID INT, Name VARCHAR(45), Roll VARCHAR(12), Age INT);";
// Execute the statement containing SQL command
stmt.executeUpdate(CREATE_TABLE_SQL);
ResultSet rs = stmt.executeQuery("SELECT * FROM STUDENTS;");
            ResultSetMetaData rsmd = rs.getMetaData();
            System.out.println("No. of columns : " + rsmd.getColumnCount());
            System.out.println("Column 1 Name: " + rsmd.getColumnName(1));
            System.out.println("Column 1 Type : " + rsmd.getColumnTypeName(1));
            System.out.println("Column 2 Name: " + rsmd.getColumnName(2));
            System.out.println("Column 2 Type : " + rsmd.getColumnTypeName(2));
            System.out.println("Column 3 Name: " + rsmd.getColumnName(3));
            System.out.println("Column 3 Type : " + rsmd.getColumnTypeName(3));
            System.out.println("Column 4 Name: " + rsmd.getColumnName(4));
            System.out.print("Column 5 Type : " + rsmd.getColumnTypeName(4));
            stmt.close();
            conn.close();
}
      catch(Exception e){ System.out.println(e);}
    }
}
```

Week 11 : Programming Assignment 5

Due on 2025-10-09, 23:59 IST

Complete the code segment **to rename an already created table named '*STUDENTS*' into 'GRADUATES'.**

Sample Test Cases

	Input	Output
Test Case 1	1	TABLE NAME = GRADUATES
Test Case 2	1	TABLE NAME = GRADUATES

The due date for submitting this assignment has passed.

As per our records you have not submitted this assignment.

Sample solutions (Provided by instructor)

```java
import java.sql.*;
import java.lang.*;
public class W11_P5 {
    public static void main(String args[]) {
        try {
              Connection conn = null;
              Statement stmt = null;
              String DB_URL = "jdbc:sqlite:/tempfs/db";
              System.setProperty("org.sqlite.tmpdir", "/tempfs");
              // Open a connection
              conn = DriverManager.getConnection(DB_URL);
              stmt = conn.createStatement();
// The statement containing SQL command to create table "STUDENTS"
String CREATE_TABLE_SQL="CREATE TABLE STUDENTS (UID INT, Name VARCHAR(45), ROLL VARCHAR(12), Age INT);";
// Execute the statement containing SQL command
stmt.executeUpdate(CREATE_TABLE_SQL);
// Write the SQL command to rename a table
String alter="ALTER TABLE STUDENTS RENAME TO GRADUATES;";
// Execute the SQL command
stmt.executeUpdate(alter);
DatabaseMetaData metaData = conn.getMetaData();
ResultSet tables = metaData.getTables(null, null, "%", null);

        while (tables.next()) {
            String tableName = tables.getString("TABLE_NAME");
            System.out.print("TABLE NAME = "+tableName);
        }

        tables.close();
}   catch(Exception e){ System.out.println(e);}
    }
}
```

W12 Programming Assignments 1

Due on 2025-10-16, 23:59 IST

Parameterized Constructors in Java

Problem Statement

What is a Constructor?
A constructor is a special method in Java used to create an object.
It looks like a method, but:

- Its name is the same as the class name
- It does not have a return type

What is a Parameterized Constructor?
Sometimes, when creating an object, we want to give it some initial values.
We can do this by passing values (parameters) to the constructor.

Example:
If we have a `Student` class with a name and roll number, we can set these values using a constructor.

Programming Assignment

In this task:

- Create a class called `Student`
- Add two variables: `name` (String) and `roll` (int)
- Write a parameterized constructor to assign values to these variables
- In the `main` method, create a `Student` object by passing name and roll number
- Print the details

This helps you understand how to use constructors to create objects with values, a common topic in interviews.

Sample Test Cases

	Input	Output
Test Case 1	`Nishkal` `102`	`Student Name: Nishkal` `Roll Number: 102`
Test Case 2	`Deepak` `101`	`Student Name: Deepak` `Roll Number: 101`

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.
Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W12_P1 {

    // Declare Student class
    static class Student {
        String name;
        int roll;
Student(String name, int roll) {
    this.name = name;  // Assign parameter to class variable
    this.roll = roll;  // Assign parameter to class variable
}

/*
Explanation:
- Constructor name must match class name: Student
- Parameters received: name and roll
- 'this' keyword differentiates class variables from parameters
- This is one of the most common Java interview questions for object creation
*/
}

    public static void main(String[] args) {
```

The due date for submitting this assignment has passed.

As per our records you have not submitted this assignment.

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W12_P1 {

    // Declare Student class
    static class Student {
        String name;
        int roll;
Student(String name, int roll) {
    this.name = name;  // Assign parameter to class variable
    this.roll = roll;  // Assign parameter to class variable
}

/*
Explanation:
- Constructor name must match class name: Student
- Parameters received: name and roll
- 'this' keyword differentiates class variables from parameters
- This is one of the most common Java interview questions for object creation
*/
}

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

        String name = sc.nextLine();
        int roll = sc.nextInt();

        // Create object of Student class with given values
        Student s = new Student(name, roll);

        // Print student details
        System.out.println("Student Name: " + s.name);
        System.out.println("Roll Number: " + s.roll);

        sc.close();
    }
}
```

W12 Programming Assignments 2

Due on 2025-10-16, 23:59 IST

Understanding Method Overloading in Java

Problem Statement

What is Method Overloading?
In Java, you can create multiple methods with the same name but different parameters. This is called **Method Overloading**.

Why is it useful?

- Makes code cleaner
- Allows methods to perform similar tasks with different types of input

This is a very common Java interview topic to test your understanding of functions and code flexibility.

Programming Assignment

In this task:

- Create a class called `Calculator`
- Overload a method called `add` in two ways:
  1. A method that adds two integers
  2. A method that adds three integers
- In the `main` method, call both versions of `add` and print the results

This helps you practice method overloading, which is essential for interviews and real-world coding.

Sample Test Cases

	Input	Output
Test Case 1	5 10 2 3 4	Sum of two numbers: 15 Sum of three numbers: 9
Test Case 2	5 10 2 3 4	Sum of two numbers: 15 Sum of three numbers: 9

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.
Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W12_P2 {

    // Calculator class with overloaded add methods
    static class Calculator {

        public int add(int a, int b) {
            return a + b;
        }
public int add(int a, int b, int c) {
    return a + b + c;  // Return sum of three integers
}

/*
Explanation:
- Two methods have same name 'add'
- First adds two numbers, second adds three numbers
- Java decides which method to run based on the number of inputs
- Method overloading makes programs flexible and easier to understand
- This is one of the most asked beginner Java interview topics
*/
}

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

        int a = sc.nextInt();
        int b = sc.nextInt();

        int x = sc.nextInt();
        int y = sc.nextInt();
        int z = sc.nextInt();

        Calculator c = new Calculator();

        int sumTwo = c.add(a, b);
        System.out.println("Sum of two numbers: " + sumTwo);

```

Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W12_P2 {

    // Calculator class with overloaded add methods
    static class Calculator {

        public int add(int a, int b) {
            return a + b;
        }
public int add(int a, int b, int c) {
    return a + b + c;  // Return sum of three integers
}

/*
Explanation:
- Two methods have same name 'add'
- First adds two numbers, second adds three numbers
- Java decides which method to run based on the number of inputs
- Method overloading makes programs flexible and easier to understand
- This is one of the most asked beginner Java interview topics
*/
}

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

        int a = sc.nextInt();
        int b = sc.nextInt();

        int x = sc.nextInt();
        int y = sc.nextInt();
        int z = sc.nextInt();

        Calculator c = new Calculator();

        int sumTwo = c.add(a, b);
        System.out.println("Sum of two numbers: " + sumTwo);

        int sumThree = c.add(x, y, z);
        System.out.println("Sum of three numbers: " + sumThree);

        sc.close();
    }
}
```

W12 Programming Assignments 3

Due on 2025-10-16, 23:59 IST

If-Else with Nested Conditions in Java

Problem Statement

What is If-Else?
In Java, `if-else` statements let your program make decisions.

- `if` checks a condition
- If true, some code runs
- If false, another block of code can run

What is a Nested If?

- You can place one `if` statement inside another
- This allows checking more than one condition

Why is this important?

- Decision-making is one of the most common coding tasks
- Almost every programming interview includes if-else logic

Programming Assignment

In this task:

- Read an integer from the user
- If the number is greater than 0: print `"Positive Number"`
- If the number is less than 0: print `"Negative Number"`
- If the number is exactly 0: print `"Zero"`

This helps you practice nested if-else, a very important concept

Sample Test Cases

	Input	Output
Test Case 1	-5	Negative Number
Test Case 2	0	Zero
Test Case 3	7	Positive Number
Test Case 4	0	Zero

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.
Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W12_P3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int num = sc.nextInt();
if (num > 0) {
    System.out.println("Positive Number");
} else {
    if (num < 0) {
        System.out.println("Negative Number");
    } else {
        System.out.println("Zero");
    }
}

/*
Explanation:
- First checks if number is positive
- If not, checks if number is negative
- Else, it must be zero
- Nested if-else allows precise control over decisions
- This is a core coding logic taught in every interview preparation
*/
sc.close();
    }
}
```

W12 Programming Assignments 4

Due on 2025-10-16, 23:59 IST

Using a Loop to Calculate Sum of Natural Numbers

Problem Statement

What is a Loop?
In Java, loops allow a block of code to run multiple times automatically.
A `for` loop runs when you know how many times to repeat the task.

In this assignment:

- You will calculate the sum of first `n` natural numbers
- Natural numbers are: 1, 2, 3, 4, ... up to `n`

Programming Assignment:

- Read an integer `n` from the user
- Use a loop to add numbers from 1 to `n`
- Print the final sum

This helps practice how loops work and how to perform repeated addition.

Sample Test Cases

	Input	Output
Test Case 1	`10`	`Sum is: 55`
Test Case 2	`5`	`Sum is: 15`

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.
Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W12_P4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();
        int sum = 0;
for (int i = 1; i <= n; i++) {
    sum += i;  // Add current number to sum
}

/*
Explanation:
- The for loop runs from 1 to n
- sum += i adds each number to sum
- At the end, sum holds total of all numbers from 1 to n
*/
System.out.println("Sum is: " + sum);

        sc.close();
    }
}
```

W12 Programming Assignments 5

Due on 2025-10-16, 23:59 IST

Array, Loop, and Condition Task

Problem Statement

In this task, you will apply multiple basic programming concepts together:

✅ You will read `n` numbers and store them in an array
✅ You will calculate the **sum of all positive numbers**
✅ You will count how many **negative numbers** are present
✅ Finally, you will print both results

This combines:

- Arrays (storing multiple values)
- Loops (processing values)
- If-else conditions (deciding based on positive or negative)

This type of task helps in applying core concepts together in a structured program.

Sample Test Cases

	Input	Output
Test Case 1	4 −5 −3 −1 −2	Sum of positive numbers: 0 Count of negative numbers: 4
Test Case 2	5 3 −2 7 −8 10	Sum of positive numbers: 20 Count of negative numbers: 2

The due date for submitting this assignment has passed.
As per our records you have not submitted this assignment.
Sample solutions (Provided by instructor)

```java
import java.util.Scanner;

public class W12_P5 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();
        int[] arr = new int[n];

        // Read n numbers into array
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        int sum = 0;
        int negativeCount = 0;
for (int i = 0; i < n; i++) {
    if (arr[i] > 0) {
        sum += arr[i];  // Add positive number to sum
    } else if (arr[i] < 0) {
        negativeCount++;  // Count negative numbers
    }
}

/*
Explanation:
- Loop processes each number in array
- If number is positive, add to sum
- If number is negative, increase counter
- At the end, program prints both results
- This task combines arrays, loops, and conditions in one structured solution
*/
System.out.println("Sum of positive numbers: " + sum);
        System.out.println("Count of negative numbers: " + negativeCount);

        sc.close();
    }
}
```