

LAB CONTENT

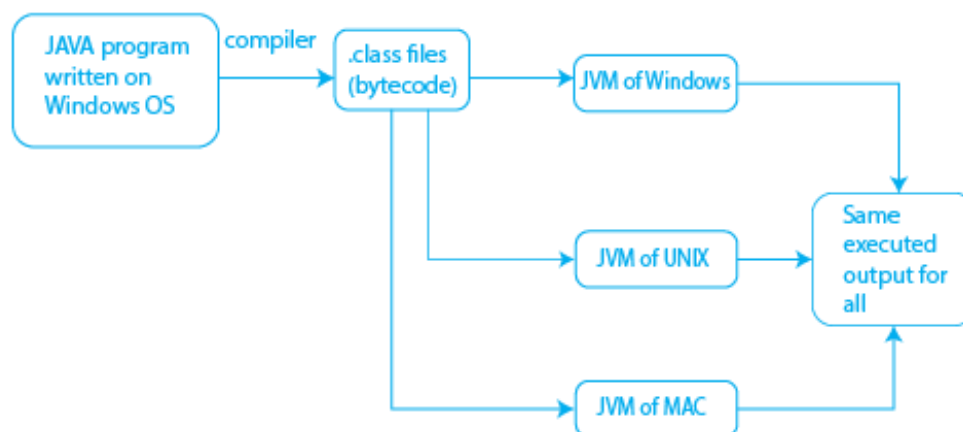
Module – Core Java

Introduction :



What is Java?

Java is a high-level, object-oriented programming language designed to have as few implementation dependencies as possible. It is widely used for building cross-platform applications, from web servers to mobile apps.

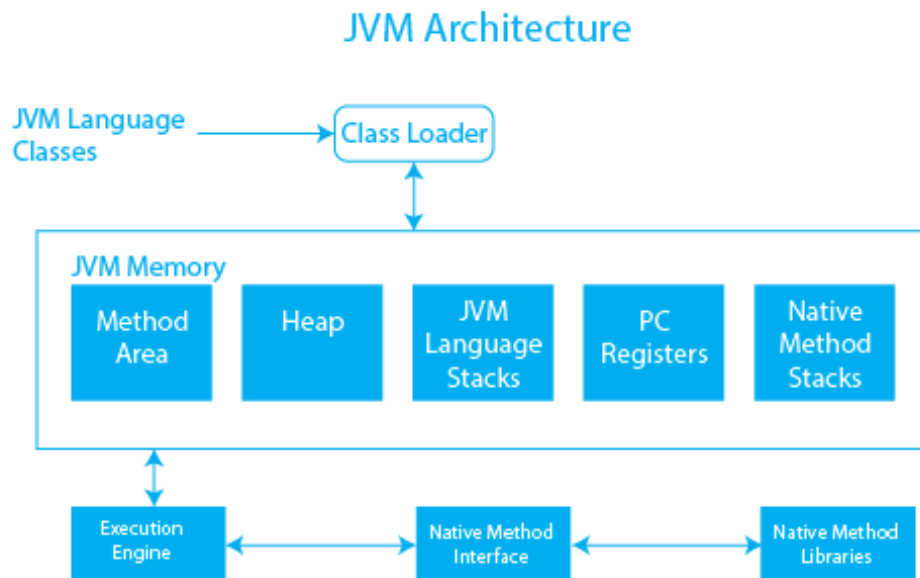


What is the Java Virtual Machine (JVM)?

The JVM is an engine that provides a runtime environment to execute Java bytecode. It is platform-independent, meaning Java code can run on any device that has a JVM installed.

What is the significance of the main method in Java?

The main method is the entry point of any Java application. It is where the program begins execution. The method signature is `public static void main(String[] args)`.



What is the difference between JDK, JRE, and JVM?

The JDK (Java Development Kit) includes tools for developing Java applications, such as the compiler and libraries. The JRE (Java Runtime Environment) provides the libraries and JVM needed to run Java applications. The JVM is the virtual machine that executes Java bytecode.

Installation Requirements for Java Programming

Java Development Kit (JDK)

Description: The JDK is required to develop Java applications. It includes the Java Runtime Environment (JRE), an interpreter/loader (Java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc), and other tools needed for Java development.

Download: Available from the Oracle website or OpenJDK.

Integrated Development Environment (IDE)

Description: An IDE is optional but highly recommended for a more efficient development process. Popular Java IDEs include:

Eclipse: Free, open-source IDE with extensive plugins.

IntelliJ IDEA: Available in both a free Community edition and a paid Ultimate edition.

NetBeans: Free, open-source IDE provided by Apache.

System Requirements

Operating System: Java can run on Windows, macOS, and various distributions of Linux.

Memory: Minimum 512 MB of RAM, 2 GB or more recommended.

Disk Space: At least 300 MB of free disk space for JDK installation.

Environment Variables

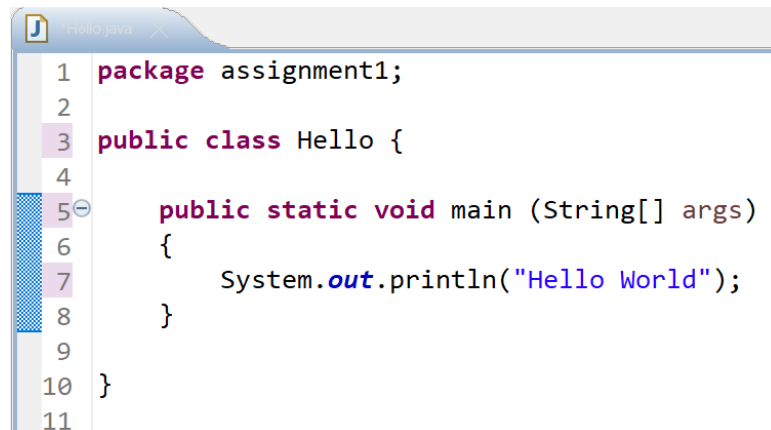
JAVA_HOME: Set this environment variable to the directory where the JDK is installed.

PATH: Add the bin directory of the JDK to the system PATH to allow running Java

Lab Assignment: 1

Question:

1. Write a Java Program to print Hello World using command line and Eclipse.



```
1 package assignment1;
2
3 public class Hello {
4
5     public static void main (String[] args)
6     {
7         System.out.println("Hello World");
8     }
9
10 }
11
```

Java Program: Hello

Class Declaration :

```
public class Hello
```

This line declares a public class named Hello.

The class is public, meaning it can be accessed from other classes.

Main Method :

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

This is the main method, the entry point of any Java application.

public: The method is accessible from anywhere.

static: The method belongs to the class, not an instance of the class. It can be called without creating an object of the class.

void: The method does not return any value.

String[] args: The method takes a single argument, an array of String objects, which can be used to pass command-line arguments.

Print Statement :

```
System.out.println("Hello World");
```

This line prints the text "Hello World" to the console.

Lab Assignment: 1

System.out: A standard output stream.

println: A method that prints a line of text to the console and then terminates the line.

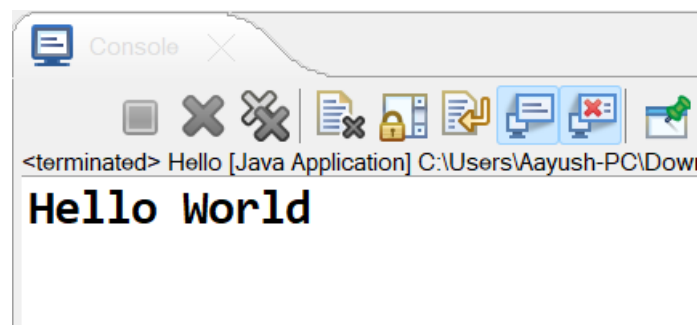
Overall Structure :

The program defines a single class named Hello.

The main method within this class is the entry point and is responsible for executing the program.

The program's primary function is to print "Hello World" to the console when executed.

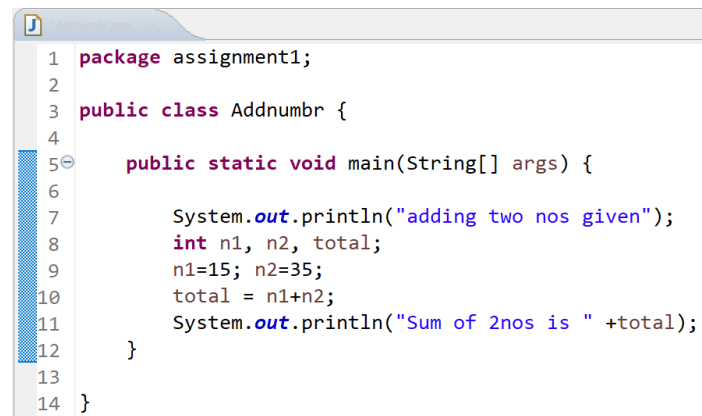
Output :



Lab Assignment: 1

Question:

2. Write a program to find SUM of a given digit.



```
1 package assignment1;
2
3 public class Addnumbr {
4
5     public static void main(String[] args) {
6
7         System.out.println("adding two nos given");
8         int n1, n2, total;
9         n1=15; n2=35;
10        total = n1+n2;
11        System.out.println("Sum of 2nos is " +total);
12    }
13
14 }
```

This Java program calculates the sum of two numbers and prints the result.

public class Addnumbr: Defines a public class named Addnumbr.

System.out.println("adding two nos given");: Prints the message "adding two nos given" to the console.

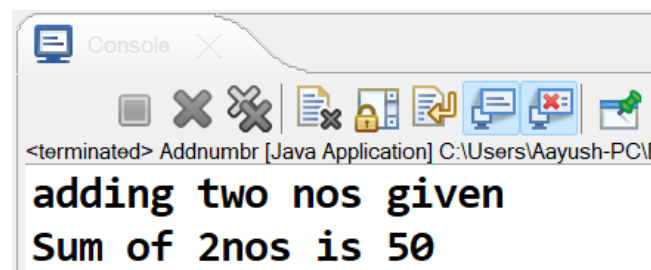
int n1, n2, total;: Declares three integer variables n1, n2, and total.

n1=15; n2=35;: Assigns the values 15 and 35 to n1 and n2 respectively.

total = n1+n2;: Calculates the sum of n1 and n2 and assigns it to total.

System.out.println("Sum of 2nos is " +total);: Prints the message "Sum of 2nos is " followed by the value of total to the console.

Output :

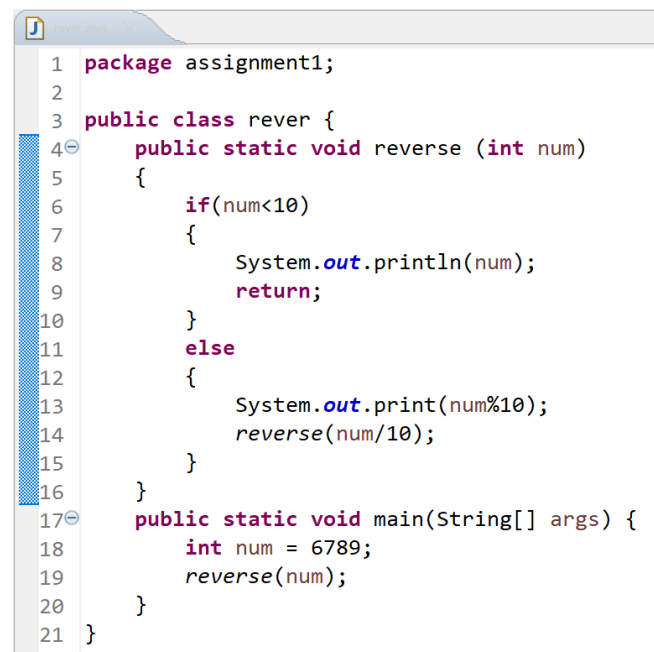


```
<terminated> Addnumbr [Java Application] C:\Users\Aayush-PC\I
adding two nos given
Sum of 2nos is 50
```

Lab Assignment: 1

Question:

3. Write a java program to reverse the given number.

A screenshot of a Java IDE window titled 'assignment1'. The code is as follows:

```
1 package assignment1;
2
3 public class rever {
4     public static void reverse (int num)
5     {
6         if(num<10)
7         {
8             System.out.println(num);
9             return;
10        }
11        else
12        {
13            System.out.print(num%10);
14            reverse(num/10);
15        }
16    }
17    public static void main(String[] args) {
18        int num = 6789;
19        reverse(num);
20    }
21 }
```

This Java program defines a class named Addnumbr which contains an inner class called rever. Inside the rever class, there's a method named reverse which takes an integer num as input and prints its digits in reverse order recursively.

Here's a brief explanation of the program:

The reverse method takes an integer num as input.

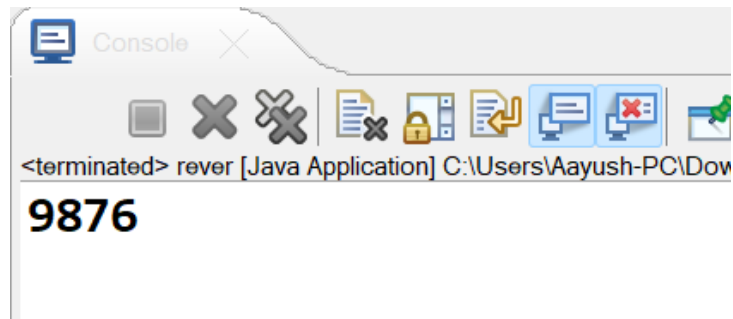
If num is less than 10 (i.e., a single digit number), it prints num and returns.

Otherwise, it prints the last digit of num (num%10) and recursively calls reverse on the remaining digits (num/10).

The main method initializes an integer variable num with the value 6789 and calls the reverse method on it to print its digits in reverse order.

So, when you run this program, it will print the digits of the number 6789 in reverse order: 9876.

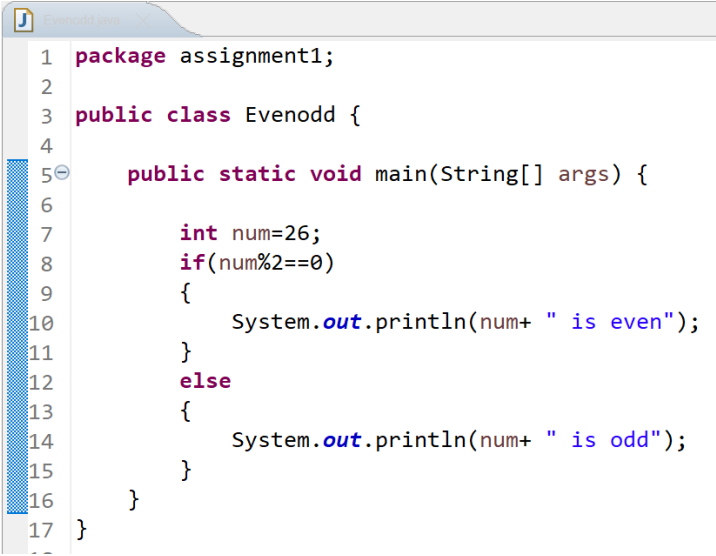
Output :



Lab Assignment: 1

Question:

3. Java Program to Check Whether a Number is Even or Odd



```
1 package assignment1;
2
3 public class Evenodd {
4
5     public static void main(String[] args) {
6
7         int num=26;
8         if(num%2==0)
9         {
10             System.out.println(num+ " is even");
11         }
12         else
13         {
14             System.out.println(num+ " is odd");
15         }
16     }
17 }
```

*This Java program determines whether a given number is even or odd.
Here's a breakdown:*

Package Declaration: The program is in the package named assignment1.

Class Declaration: The class is named Evenodd.

Main Method: The main method is the entry point of the program, where execution begins.

Variable Initialization: The integer variable num is initialized with the value 26.

Condition Check: It checks if num is divisible by 2 without any remainder, which indicates it's an even number. This is done using the condition `num % 2 == 0`.

Lab Assignment: 1

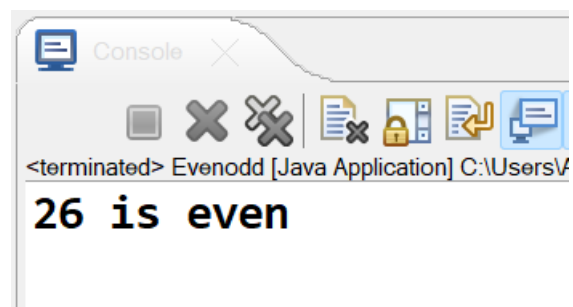
Printing Result:

If the condition is true (num is even), it prints num is even.

If the condition is false (num is odd), it prints num is odd.

Overall, the program determines whether the number stored in num is even or odd and prints the result accordingly.

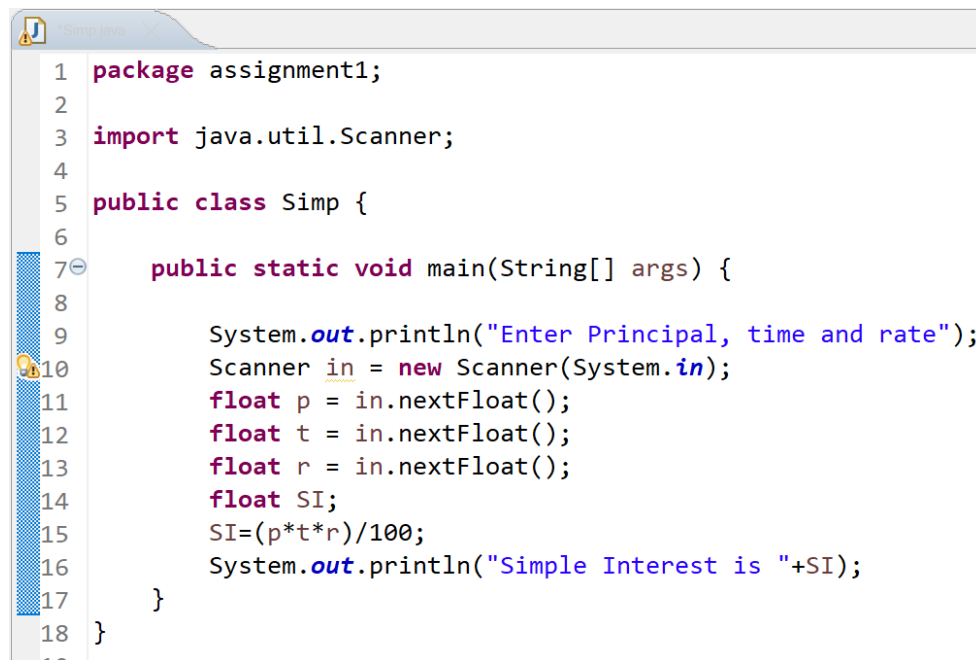
Output :



Lab Assignment: 1

Question:

5. Java Program to Calculate Simple Interest.

A screenshot of a Java IDE window titled "Simp.java". The code is as follows:

```
1 package assignment1;  
2  
3 import java.util.Scanner;  
4  
5 public class Simp {  
6  
7     public static void main(String[] args) {  
8  
9         System.out.println("Enter Principal, time and rate");  
10        Scanner in = new Scanner(System.in);  
11        float p = in.nextFloat();  
12        float t = in.nextFloat();  
13        float r = in.nextFloat();  
14        float SI;  
15        SI=(p*t*r)/100;  
16        System.out.println("Simple Interest is "+SI);  
17    }  
18 }
```

This Java program calculates the simple interest based on the provided principal amount, time period, and interest rate.

It begins by importing the Scanner class from the java.util package to facilitate user input.

The main method prompts the user to input the principal amount, time period, and interest rate.

It reads these inputs using the Scanner object in.

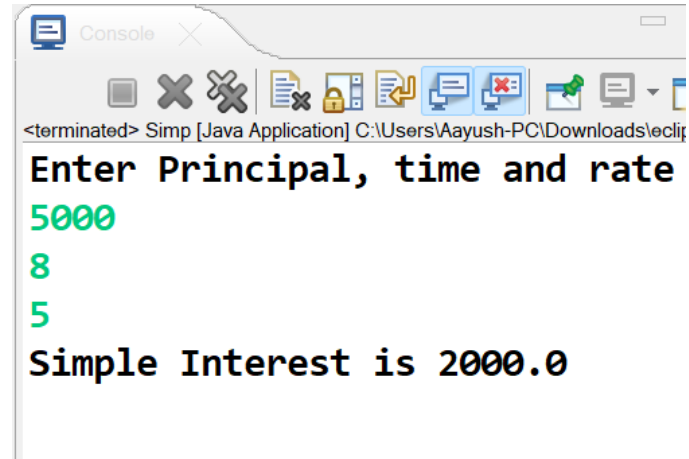
The program calculates the simple interest using the formula (principal * time * rate) / 100.

The result is stored in the variable SI.

Finally, it prints the calculated simple interest using System.out.println.

Overall, this program demonstrates basic input/output operations and arithmetic calculations in Java.

Output:



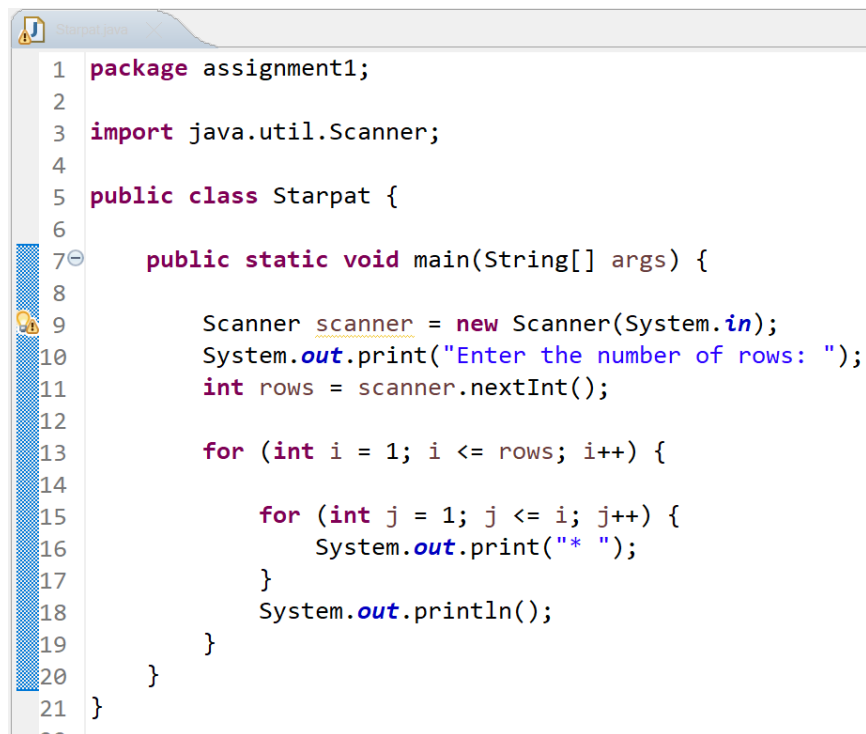
```
<terminated> Simp [Java Application] C:\Users\Aayush-PC\Downloads\ecliq
Enter Principal, time and rate
5000
8
5
Simple Interest is 2000.0
```

The screenshot shows a console window titled "Console" with a standard Windows taskbar at the top. The window contains the following text: a prompt "Enter Principal, time and rate", three lines of user input (5000, 8, and 5), and the final output "Simple Interest is 2000.0". The input values are displayed in green, while the prompt and output are in black. The window title bar includes standard minimize, maximize, and close buttons.

Lab Assignment: 1

Question:

6. Java Program to Print Right Triangle Star Pattern

A screenshot of a Java IDE window titled 'Starpat.java'. The code is as follows:

```
1 package assignment1;
2
3 import java.util.Scanner;
4
5 public class Starpat {
6
7     public static void main(String[] args) {
8
9         Scanner scanner = new Scanner(System.in);
10        System.out.print("Enter the number of rows: ");
11        int rows = scanner.nextInt();
12
13        for (int i = 1; i <= rows; i++) {
14
15            for (int j = 1; j <= i; j++) {
16                System.out.print("* ");
17            }
18            System.out.println();
19        }
20    }
21 }
```

This Java program prints a pattern of stars in the shape of a right triangle. Here's a brief explanation of how it works:

The program begins by importing the Scanner class from the java.util package to allow user input.

The Starpat class contains the main method, which serves as the entry point of the program.

Inside the main method:

It creates a Scanner object named scanner to read input from the console.

It prompts the user to enter the number of rows for the pattern.

It reads the user input as an integer and assigns it to the variable rows.

The program then enters a nested loop structure to generate the pattern:

The outer loop (for loop) iterates over each row from 1 to the specified number of rows (rows).

The inner loop (for loop) iterates over each column within the current row.

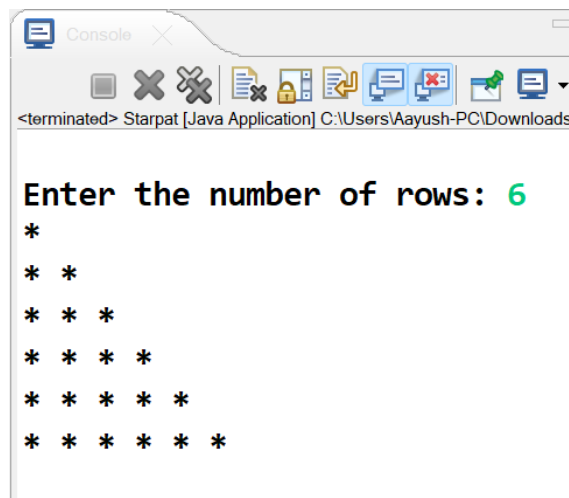
For each column, it prints a star followed by a space (*).

After printing all stars in the current row, it moves to the next line using `System.out.println()` to start a new row.

Once the loops complete execution, the program exits.

In summary, the program takes user input to determine the number of rows and then prints a right triangle pattern of stars, where each row contains an increasing number of stars.

Output :



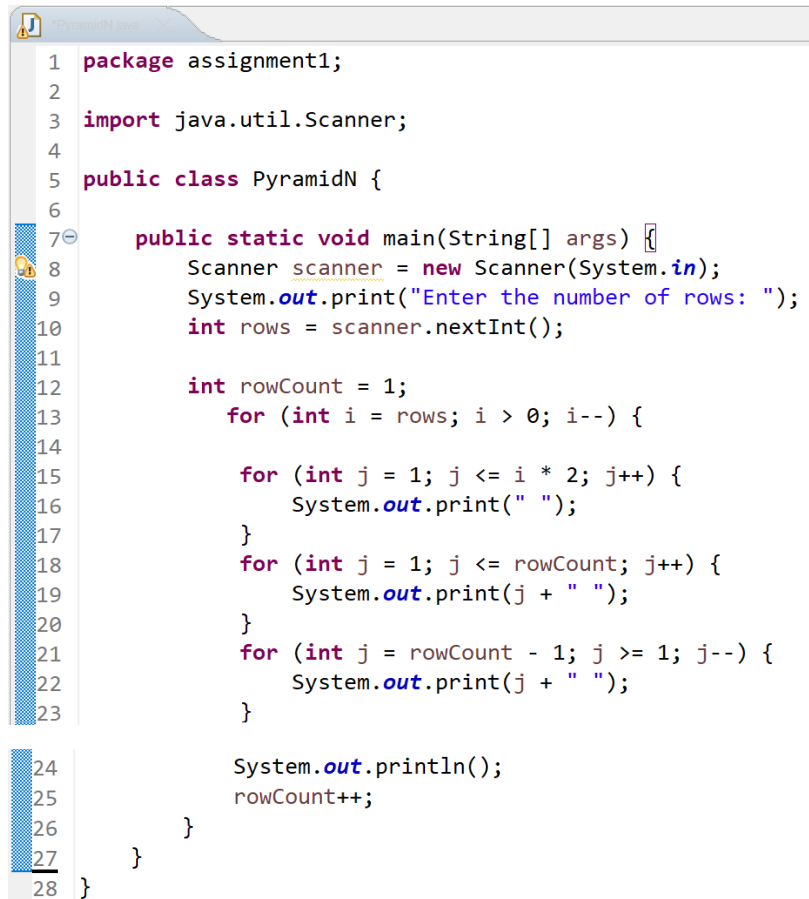
```
<terminated> Starpat [Java Application] C:\Users\Aayush-PC\Downloads

Enter the number of rows: 6
*
* *
* * *
* * * *
* * * * *
* * * * * *
```

Lab Assignment: 1

Question:

7. Java Program to Print Pyramid Number Pattern

A screenshot of a Java IDE window titled "Pyramid1.java". The code is as follows:

```
1 package assignment1;
2
3 import java.util.Scanner;
4
5 public class PyramidN {
6
7     public static void main(String[] args) {
8         Scanner scanner = new Scanner(System.in);
9         System.out.print("Enter the number of rows: ");
10        int rows = scanner.nextInt();
11
12        int rowCount = 1;
13        for (int i = rows; i > 0; i--) {
14
15            for (int j = 1; j <= i * 2; j++) {
16                System.out.print(" ");
17            }
18            for (int j = 1; j <= rowCount; j++) {
19                System.out.print(j + " ");
20            }
21            for (int j = rowCount - 1; j >= 1; j--) {
22                System.out.print(j + " ");
23            }
24
25            System.out.println();
26            rowCount++;
27        }
28    }
```

Input Prompt:

It prompts the user to enter the number of rows for the pyramid.

Pyramid Generation:

It uses nested loops to generate the pyramid pattern.

The outer loop (for loop) iterates through each row of the pyramid, decrementing from the user-inputted number of rows to 1.

The first inner loop (for loop) prints spaces to create the left indentation for each row.

Lab Assignment: 1

The second inner loop prints numbers in increasing order from 1 to the current row count.

The third inner loop prints numbers in decreasing order from the current row count minus 1 down to 1.

Output:

After each row is generated, a newline character is printed to move to the next row.

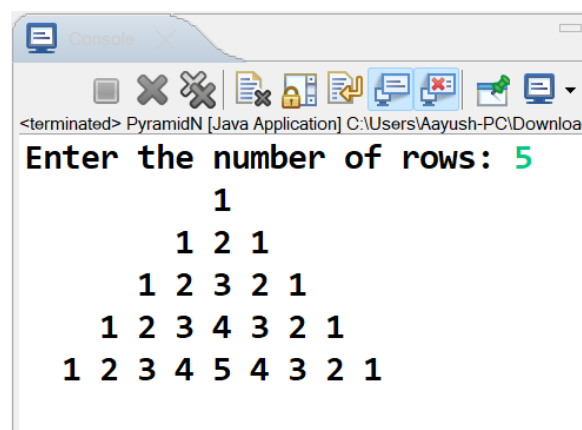
Incrementing rowCount:

The rowCount variable keeps track of the current row number being printed.

It increments after each row is printed to move to the next row.

This program effectively prints a pyramid pattern with numbers based on the user-inputted number of rows.

Output :

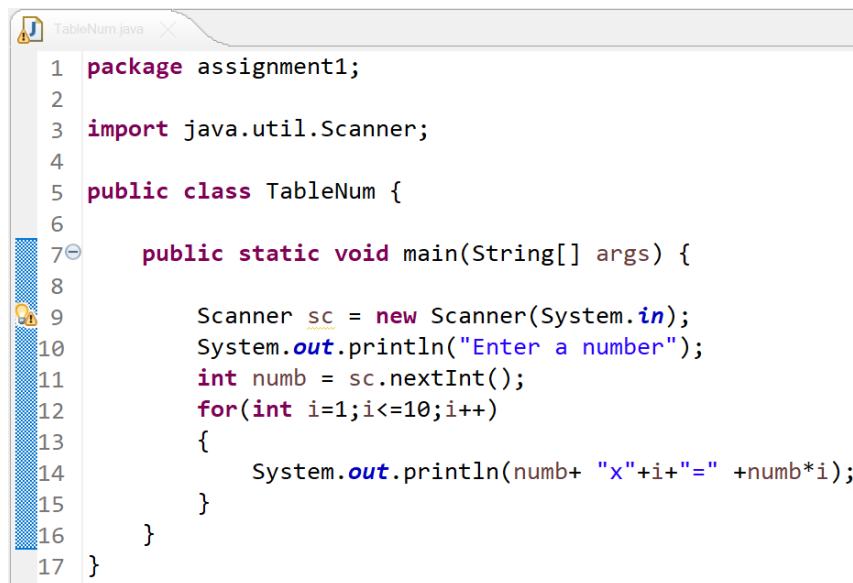


```
Console
<terminated> PyramidN [Java Application] C:\Users\Aayush-PC\Downloa
Enter the number of rows: 5
      1
    1 2 1
  1 2 3 2 1
1 2 3 4 3 2 1
1 2 3 4 5 4 3 2 1
```

Lab Assignment: 1

Question:

8. Write a java program to print Table of given Number



```
1 package assignment1;
2
3 import java.util.Scanner;
4
5 public class TableNum {
6
7     public static void main(String[] args) {
8
9         Scanner sc = new Scanner(System.in);
10        System.out.println("Enter a number");
11        int numb = sc.nextInt();
12        for(int i=1;i<=10;i++)
13        {
14            System.out.println(numb+ "x"+i+"=" +numb*i);
15        }
16    }
17 }
```

This Java program generates a multiplication table for a given number entered by the user. Here's a brief explanation:

It starts by importing the Scanner class from the java.util package to read input from the user.

The TableNum class is defined, which contains the main method, the entry point of the program.

Inside the main method:

An instance of Scanner is created to read input from the console.

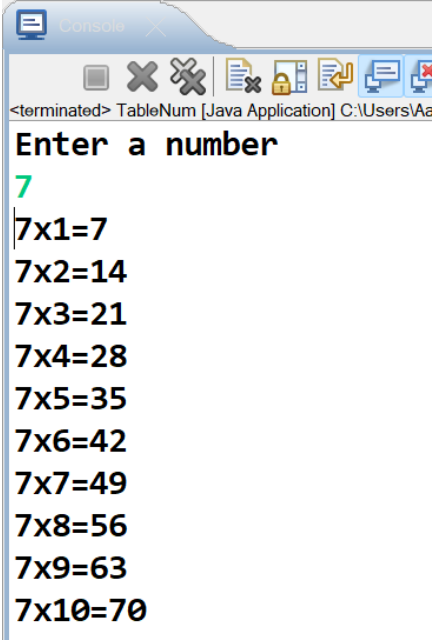
The program prompts the user to enter a number.

The entered number is stored in the variable numb.

A for loop is used to iterate from 1 to 10.

Inside the loop, the program prints the multiplication table for the entered number, where each line contains the number, the multiplier (from 1 to 10), and the result of the multiplication.

Output :

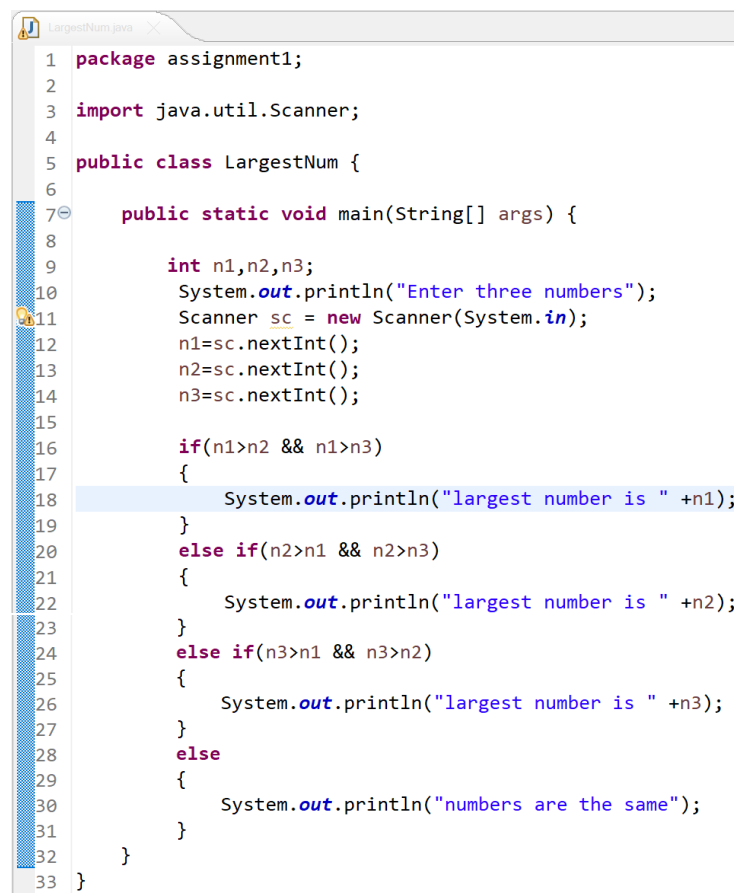


```
Console
<terminated> TableNum [Java Application] C:\Users\Aa
Enter a number
7
7x1=7
7x2=14
7x3=21
7x4=28
7x5=35
7x6=42
7x7=49
7x8=56
7x9=63
7x10=70
```

Lab Assignment: 1

Question:

9. Write Java Program to Find Largest of Three Numbers.

A screenshot of a Java IDE window titled 'LargestNum.java'. The code is as follows:

```
1 package assignment1;
2
3 import java.util.Scanner;
4
5 public class LargestNum {
6
7     public static void main(String[] args) {
8
9         int n1,n2,n3;
10        System.out.println("Enter three numbers");
11        Scanner sc = new Scanner(System.in);
12        n1=sc.nextInt();
13        n2=sc.nextInt();
14        n3=sc.nextInt();
15
16        if(n1>n2 && n1>n3)
17        {
18            System.out.println("largest number is " +n1);
19        }
20        else if(n2>n1 && n2>n3)
21        {
22            System.out.println("largest number is " +n2);
23        }
24        else if(n3>n1 && n3>n2)
25        {
26            System.out.println("largest number is " +n3);
27        }
28        else
29        {
30            System.out.println("numbers are the same");
31        }
32    }
33 }
```

Define the Class:

Define a public class named LargestNum.

Define the Main Method:

Create the main method, which serves as the entry point of the program.

Declare Variables:

Declare three integer variables (n1, n2, n3) to store the three numbers entered by the user.

Prompt User for Input:

Print a message asking the user to enter three numbers.

Lab Assignment: 1

Create a Scanner object to read input from the user.

Read three integers from the user and store them in the variables declared earlier.

Determine the Largest Number:

Use an if-else statement to compare the three numbers:

If the first number (n1) is greater than the other two, print that the largest number is n1.

If the second number (n2) is greater than the other two, print that the largest number is n2.

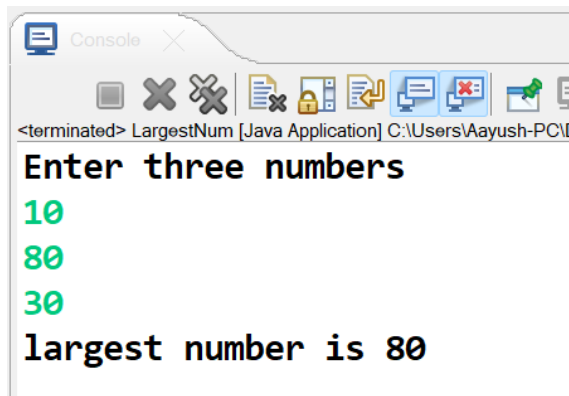
If the third number (n3) is greater than the other two, print that the largest number is n3.

If none of the conditions above are true, print that the numbers are the same (this handles the case where there's a tie).

End of Main Method and Class

Close the main method and the class to complete the program.

Output:



```
<terminated> LargestNum [Java Application] C:\Users\Aayush-PC\I
Enter three numbers
10
80
30
largest number is 80
```

Lab Assignment: 1

Question:

10. Java Program to Make a Simple Calculator Using switch...case

```
1 package assignment1;
2
3 import java.util.Scanner;
4
5 public class SimCalculator {
6
7     public static void main(String[] args) {
8         int op, n1, n2;
9         System.out.println(" 1 - add \n 2 - subtract "
10                             + "\n 3 - multiply \n 4 - divide \n");
11
12         System.out.println("choose operator");
13         Scanner sc = new Scanner(System.in);
14         op = sc.nextInt();
15         System.out.println("enter 1st no");
16         n1 = sc.nextInt();
17         System.out.println("enter 2nd no");
18         n2 = sc.nextInt();
19         int result=0;
20
21         switch(op)
22         {
23             case 1:
24                 result = n1+n2;
25                 break;
26             case 2:
27                 result = n1-n2;
28                 break;
29             case 3:
30                 result = n1*n2;
31                 break;
32             case 4:
33                 result = n1/n2;
34                 break;
35             default:
36                 System.out.println("invalid oprtr");
37         }
38         System.out.println("result is:"+result);
39     }
40 }
```

Class Declaration:

The class SimCalculator is defined.

Main Method:

The main method is the entry point of the program.

Variable Declaration:

Three integer variables op, n1, and n2 are declared.

Lab Assignment: 1

User Instructions:

Instructions are printed to the console, listing the operations (add, subtract, multiply, divide).

User Input:

A Scanner object is created to read user input.

The user is prompted to choose an operator (1-4) and enter two numbers (n1 and n2).

Variable Initialization:

The result variable is initialized to 0.

Switch Statement:

Based on the user's choice (op), a switch statement executes the corresponding arithmetic operation:

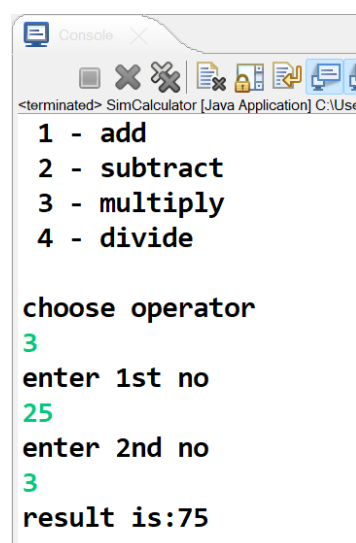
- 1 for addition
- 2 for subtraction
- 3 for multiplication
- 4 for division

If the user enters an invalid operator, a message is printed.

Output Result:

The result of the chosen operation is printed to the console.

Output:



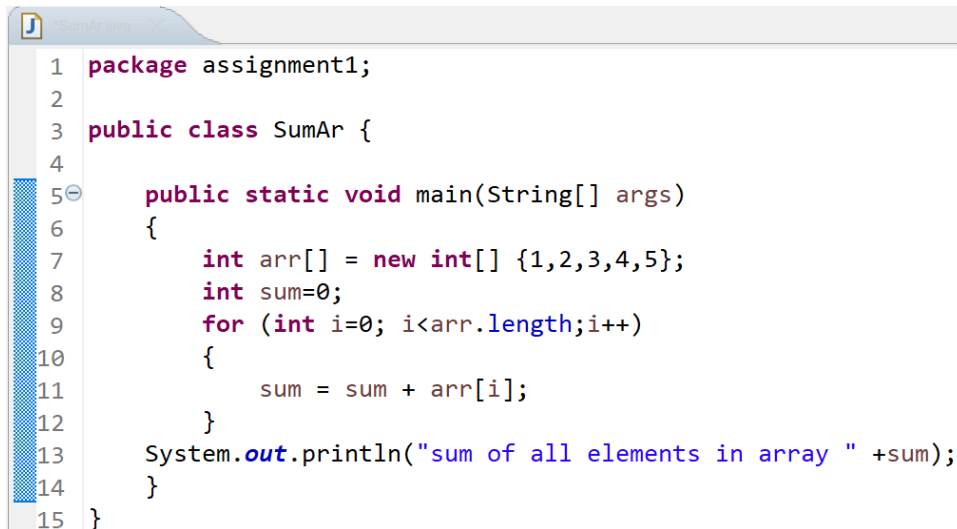
```
Console X
<terminated> SimCalculator [Java Application] C:\Use
1 - add
2 - subtract
3 - multiply
4 - divide

choose operator
3
enter 1st no
25
enter 2nd no
3
result is:75
```

Lab Assignment: 1

Question:

11. Write a Java program to sum values of an array.



```
1 package assignment1;
2
3 public class SumAr {
4
5     public static void main(String[] args)
6     {
7         int arr[] = new int[] {1,2,3,4,5};
8         int sum=0;
9         for (int i=0; i<arr.length;i++)
10        {
11            sum = sum + arr[i];
12        }
13        System.out.println("sum of all elements in array " +sum);
14    }
15 }
```

Class Definition:

A public class named SumAr is defined.

Main Method:

The main method is the entry point of the program.

Array Initialization:

An integer array arr is initialized with values {1, 2, 3, 4, 5}.

Sum Variable:

An integer variable sum is initialized to 0 to store the sum of array elements.

For Loop:

A for loop iterates over each element of the array.

In each iteration, the current array element is added to sum.

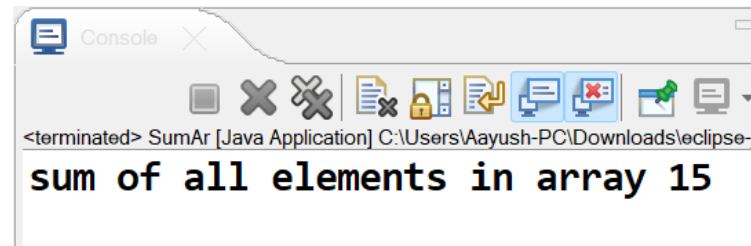
Print Statement:

Lab Assignment: 1

After the loop, the program prints the total sum of the array elements.

The program calculates and prints the sum of all elements in the given array.

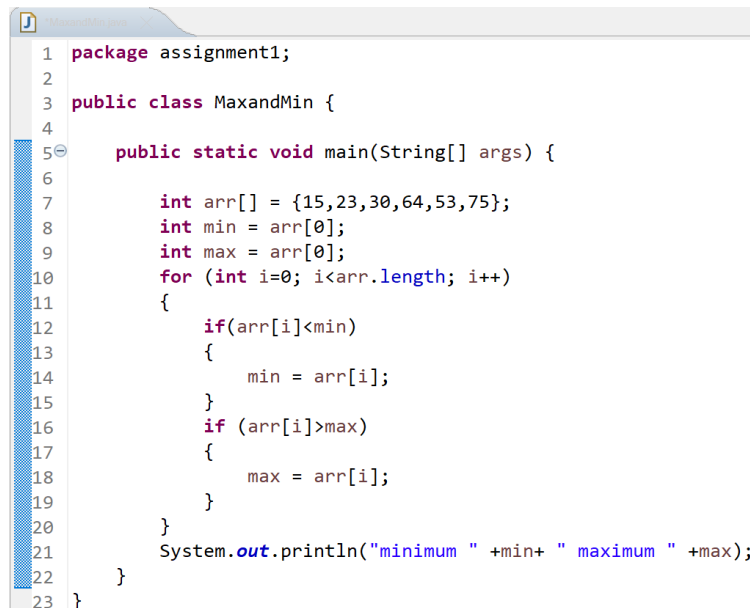
Output:



Lab Assignment: 1

Question:

12. Write a Java program to find the maximum and minimum value of an array.



```
1 package assignment1;
2
3 public class MaxandMin {
4
5     public static void main(String[] args) {
6
7         int arr[] = {15,23,30,64,53,75};
8         int min = arr[0];
9         int max = arr[0];
10        for (int i=0; i<arr.length; i++)
11        {
12            if(arr[i]<min)
13            {
14                min = arr[i];
15            }
16            if (arr[i]>max)
17            {
18                max = arr[i];
19            }
20        }
21        System.out.println("minimum " +min+ " maximum " +max);
22    }
23 }
```

Class Definition:

The class MaxandMin is defined as public, making it accessible from other classes.

Main Method:

The main method is the entry point of the program.

Array Initialization:

An integer array arr is initialized with the values {15, 23, 30, 64, 53, 75}.

Initial Min and Max:

Two integer variables, min and max, are initialized with the first element of the array (arr[0]).

Loop Through Array:

A for loop iterates through each element of the array:

If the current element is less than min, min is updated to this element.

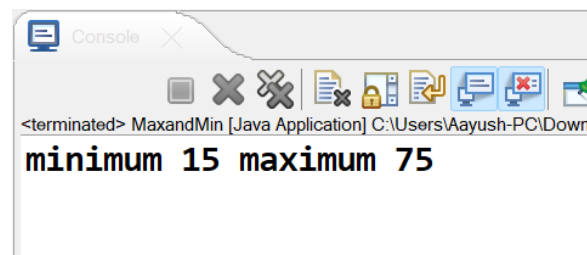
Lab Assignment: 1

If the current element is greater than max, max is updated to this element.

Print Results:

The program prints the minimum and maximum values found in the array.

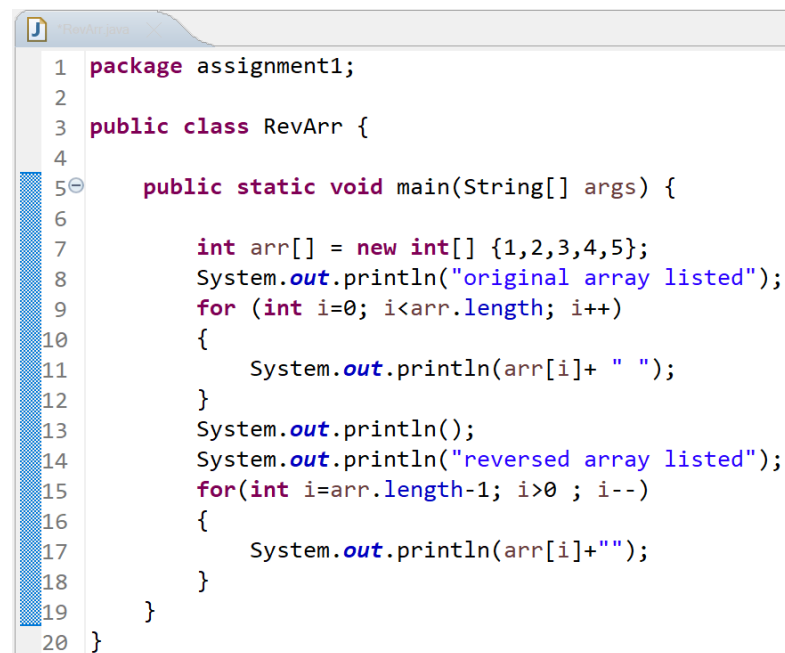
Output :



Lab Assignment: 1

Question:

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



```
1 package assignment1;
2
3 public class RevArr {
4
5     public static void main(String[] args) {
6
7         int arr[] = new int[] {1,2,3,4,5};
8         System.out.println("original array listed");
9         for (int i=0; i<arr.length; i++)
10        {
11            System.out.println(arr[i]+ " ");
12        }
13        System.out.println();
14        System.out.println("reversed array listed");
15        for(int i=arr.length-1; i>0 ; i--)
16        {
17            System.out.println(arr[i]+"");
18        }
19    }
20 }
```

Class Definition:

Defines a public class named RevArr.

Main Method:

Contains the main method, which is the entry point of the program.

Array Initialization:

Initializes an integer array arr with the elements {1, 2, 3, 4, 5}.

Print Original Array:

Prints a message indicating the original array.

Uses a for loop to iterate over the array elements from the beginning to the end and prints each element.

Print Reversed Array:

Prints a message indicating the reversed array.

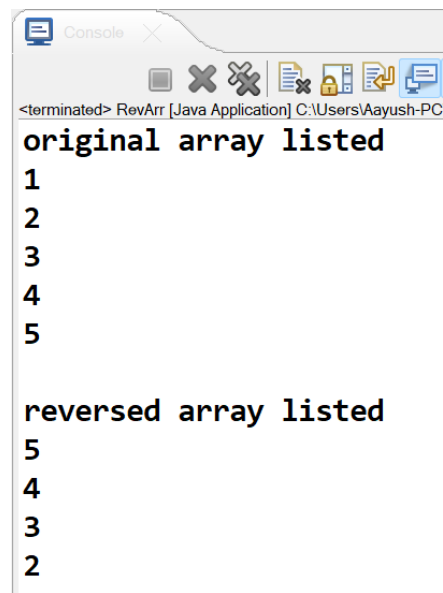
Uses a for loop to iterate over the array elements from the end to the beginning and prints each element.

Lab Assignment: 1

Summary:

The program initializes an array, prints it in its original order, then prints it in reverse order

Output:

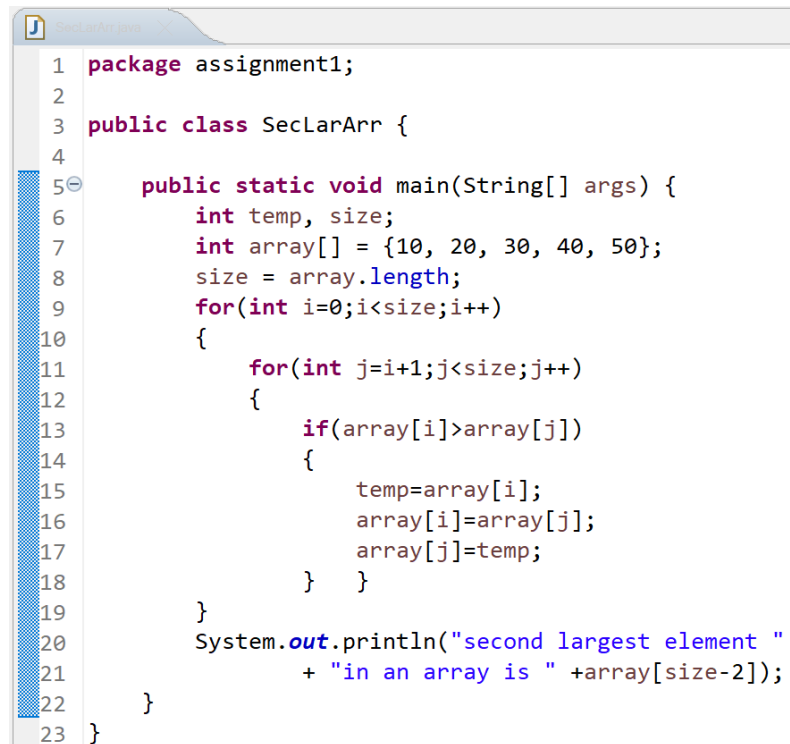


```
<terminated> RevArr [Java Application] C:\Users\Aayush-PC  
original array listed  
1  
2  
3  
4  
5  
  
reversed array listed  
5  
4  
3  
2
```

Lab Assignment: 1

Question:

14. Write a Java program to find the second largest element in an array.



```
1 package assignment1;
2
3 public class SecLarArr {
4
5     public static void main(String[] args) {
6         int temp, size;
7         int array[] = {10, 20, 30, 40, 50};
8         size = array.length;
9         for(int i=0;i<size;i++)
10        {
11            for(int j=i+1;j<size;j++)
12            {
13                if(array[i]>array[j])
14                {
15                    temp=array[i];
16                    array[i]=array[j];
17                    array[j]=temp;
18                }
19            }
20            System.out.println("second largest element "
21                               + "in an array is " +array[size-2]);
22        }
23    }
```

The class SecLarArr is defined.

Main Method:

The main method is the entry point of the program.

Variable Initialization:

Declare variables temp and size.

Initialize the array array with the elements {10, 20, 30, 40, 50}.

Set size to the length of the array.

Sorting the Array:

Use a nested for loop to sort the array in ascending order.

The outer loop runs from the start to the end of the array.

Lab Assignment: 1

The inner loop compares each element with the subsequent elements.

If the current element is greater than the next element, they are swapped.

Print the Second Largest Element:

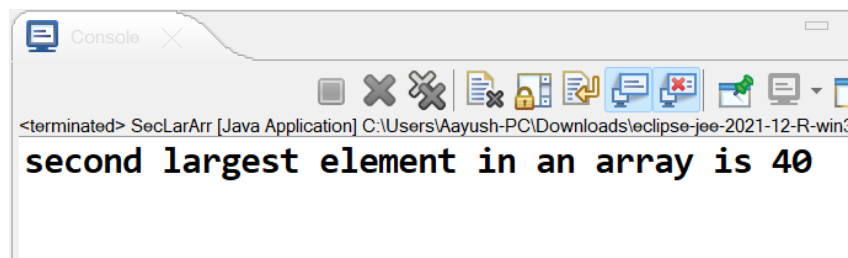
After sorting, the second largest element is at the position size-2.

Print the second largest element.

Summary:

The program sorts the array in ascending order and then prints the second largest element, which is the second last element in the sorted array.

Output:

A screenshot of an Eclipse IDE console window. The window title is "Console" with a close button. Below the title bar is a toolbar with icons for running, debugging, and other IDE functions. The console text shows a terminated Java application: "<terminated> SecLarArr [Java Application] C:\Users\Aayush-PC\Downloads\eclipse-jee-2021-12-R-win...". The output of the program is displayed in a large, bold, black font: "second largest element in an array is 40".

```
<terminated> SecLarArr [Java Application] C:\Users\Aayush-PC\Downloads\eclipse-jee-2021-12-R-win...
second largest element in an array is 40
```

Lab Assignment: 2

Question:

1. Create a class called Emp with data members empno, empname, designation, dept and salary and methods as readEmpData() (to read values to data members) and displayEmpData() (to display data members values to the screen) create an employee instance and display its information.

```
1 package assignment2;
2
3 import java.util.Scanner;
4
5 public class Emp {
6     int empno;
7     String empname, designation, dept;
8     double salary;
9
10    public void readEmpData() {
11        Scanner sc = new Scanner(System.in);
12        System.out.println("Enter Employee Number:");
13        empno = sc.nextInt();
14        System.out.println("Enter Employee Name:");
15        empname = sc.next();
16        System.out.println("Enter Designation:");
17        designation = sc.next();
18        System.out.println("Enter Department:");
19        dept = sc.next();
20        System.out.println("Enter Salary:");
21        salary = sc.nextDouble();
22    }
23
24    public void displayEmpData() {
25        System.out.println("Employee Number: " + empno);
26        System.out.println("Employee Name: " + empname);
27        System.out.println("Designation: " + designation);
28        System.out.println("Department: " + dept);
29        System.out.println("Salary: " + salary);
30    }
31
32    public static void main(String[] args) {
33        Emp employee = new Emp();
34        employee.readEmpData();
35        employee.displayEmpData();
36    }
37 }
```

Defining the Emp Class:

The Emp class is defined, which represents an employee. It contains instance variables to store employee data (empno, empname, designation, dept, salary), as well as methods to read input (readEmpData()) and display data (displayEmpData()).

Lab Assignment: 2

readEmpData() Method:

This method prompts the user to enter employee data using `System.out.println()` statements and reads the input using a `Scanner` object (`sc`). It reads the employee number as an integer, employee name, designation, department, and salary as double.

displayEmpData() Method:

This method simply prints out the employee data using `System.out.println()` statements.

Main Method:

The `main()` method is the entry point of the program.

It creates an object of the `Emp` class named `employee`.

It then calls the `readEmpData()` method to input employee data for the `employee` object.

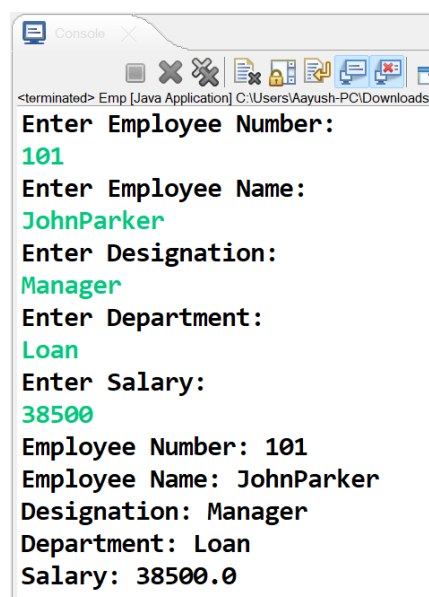
Finally, it calls the `displayEmpData()` method to display the entered employee data.

User Interaction:

When the program runs, it prompts the user to input employee data.

After the user enters the data, it displays the entered data back to the user.

Output:



```
<terminated> Emp [Java Application] C:\Users\Aayush-PC\Downloads\
Enter Employee Number:
101
Enter Employee Name:
JohnParker
Enter Designation:
Manager
Enter Department:
Loan
Enter Salary:
38500
Employee Number: 101
Employee Name: JohnParker
Designation: Manager
Department: Loan
Salary: 38500.0
```

Lab Assignment: 2

Question:

2. Create a class Electricity bill with data members as customer number, customer name, units consumed and methods as follows:

1. readData() - to read the values of data members.
2. showData - to display the customer details
3. computeBill() - to calculate and return electricity charges to be paid.

number of units	charges
< = 100	Rs. 1.20
for the next 200 units	Rs. 2.00
for the next 300 units	Rs. 3.00
for more	Rs. 5.00

ex: input = 320 units output = $100 \times 1.20 + 200 \times 2.00 + 20 \times 3.00 =$ Rs. 580

Read customer object values, calculate electricity bill and display the values.

```
1 package assignment2;
2
3 import java.util.Scanner;
4
5 public class ElectricityBill {
6     private int customerNumber;
7     private String customerName;
8     private int unitsConsumed;
9
10    public void readData() {
11        Scanner scanner = new Scanner(System.in);
12        System.out.print("Enter customer number: ");
13        customerNumber = scanner.nextInt();
14        scanner.nextLine(); // Consume newline
15        System.out.print("Enter customer name: ");
16        customerName = scanner.nextLine();
17        System.out.print("Enter units consumed: ");
18        unitsConsumed = scanner.nextInt();
19    }
20    public void showData() {
21        System.out.println("Customer Number: " + customerNumber);
22        System.out.println("Customer Name: " + customerName);
23        System.out.println("Units Consumed: " + unitsConsumed);
24    }
25    public double computeBill() {
26        double bill = 0;
27        if (unitsConsumed <= 100) {
28            bill = unitsConsumed * 1.20;
29        } else if (unitsConsumed <= 300) {
30            bill = 100 * 1.20 + (unitsConsumed - 100) * 2.00;
31        } else if (unitsConsumed <= 600) {
32            bill = 100 * 1.20 + 200 * 2.00
33                + (unitsConsumed - 300) * 3.00;
34        } else {
35            bill = 100 * 1.20 + 200 * 2.00 + 300 * 3.00
36                + (unitsConsumed - 600) * 5.00;
37        }
38        return bill;
39    }
}
```


Lab Assignment: 2

```
40 public static void main(String[] args) {  
41     ElectricityBill customer = new ElectricityBill();  
42     customer.readData();  
43     System.out.println("\nCustomer Details:");  
44     customer.showData();  
45     double billAmount = customer.computeBill();  
46     System.out.println("Electricity Bill: Rs. " + billAmount);  
47 }  
48 }
```

Defining the ElectricityBill Class:

The ElectricityBill class is defined, which contains fields for customerNumber, customerName, and unitsConsumed.

It also contains methods:

readData(): Reads customer details from the user input.

showData(): Displays the customer details.

computeBill(): Computes the electricity bill based on the units consumed.

Reading Customer Data (readData() Method):

Inside the readData() method, the program prompts the user to enter the customer number, name, and units consumed using Scanner.

User inputs are stored in the corresponding fields of the ElectricityBill object.

Displaying Customer Details (showData() Method):

The showData() method displays the customer details such as customer number, name, and units consumed.

Computing Electricity Bill (computeBill() Method):

The computeBill() method calculates the electricity bill based on the units consumed according to the specified rates:

For the first 100 units: Rs. 1.20 per unit.

For the next 200 units: Rs. 2.00 per unit.

For the next 300 units: Rs. 3.00 per unit.

For any additional units beyond 600: Rs. 5.00 per unit.

Main Method:

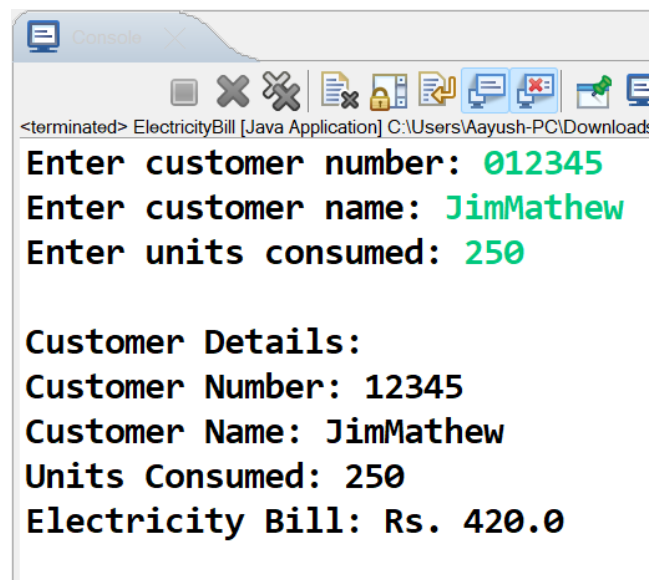
In the main() method, an instance of the ElectricityBill class is created.

Lab Assignment: 2

The readData() method is called to input customer details.
Customer details are displayed using the showData() method.
The electricity bill amount is computed using the computeBill() method and displayed to the user.

This program allows users to input customer details and calculates the electricity bill based on the units consumed, displaying both the customer details and the computed bill amount.

Output :



```
<terminated> ElectricityBill [Java Application] C:\Users\Aayush-PC\Download:
Enter customer number: 012345
Enter customer name: JimMathew
Enter units consumed: 250

Customer Details:
Customer Number: 12345
Customer Name: JimMathew
Units Consumed: 250
Electricity Bill: Rs. 420.0
```

Lab Assignment: 2

Question:

3) Write a Java program to create Student class with member variable as id, name, mark and result. Use method to initialize the value of name, id and marks. Write a member function to find the result and display the student details with result.

```
1 package assignment2;
2
3 import java.util.Scanner;
4
5 public class Student {
6     int id;
7     String name;
8     double mark;
9     String result;
10    public void initialize(int id, String name, double mark) {
11        this.id = id;
12        this.name = name;
13        this.mark = mark;
14    }
15    public void findResult() {
16        if (mark >= 40) {
17            result = "Pass";
18        } else {
19            result = "Fail";
20        }
21    }
22    public void displayDetails() {
23        System.out.println("Student ID: " + id);
24        System.out.println("Student Name: " + name);
25        System.out.println("Student Mark: " + mark);
26        System.out.println("Result: " + result);
27    }
28    public static void main(String[] args) {
29        Scanner scanner = new Scanner(System.in);
30        System.out.println("Enter student ID:");
31        int id = scanner.nextInt();
32        scanner.nextLine();
33
34        System.out.println("Enter student name:");
35        String name = scanner.nextLine();
36
37        System.out.println("Enter student mark:");
38        double mark = scanner.nextDouble();
39
40        Student student = new Student();
41        student.initialize(id, name, mark);
42        student.findResult();
43        student.displayDetails();
44    }
45 }
```

Execution Flow:

The program starts by defining the Student class.

It declares instance variables id, name, mark, and result.

The initialize method sets the values of id, name, and mark.

Lab Assignment: 2

The findResult method determines whether the student has passed or failed based on the mark.

The displayDetails method prints the student details along with the result.

In the main method:

It creates a Scanner object to read input from the user.

It prompts the user to enter student details: ID, name, and mark.

It reads the input values using the Scanner object.

It creates a new Student object.

It initializes the student details using the initialize method.

It finds the result using the findResult method.

It displays the student details along with the result using the displayDetails method.

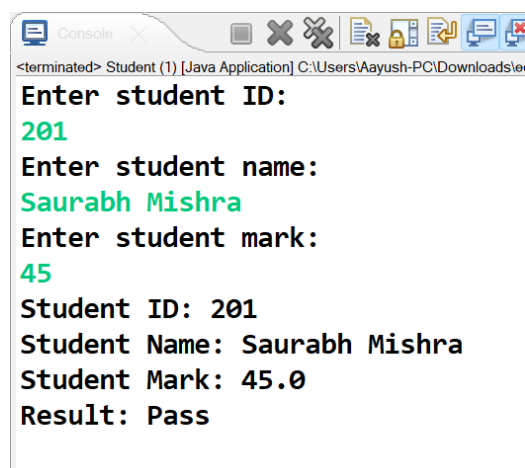
Input and Output:

The program prompts the user to enter the student ID, name, and mark.

It then calculates whether the student has passed or failed based on the mark.

Finally, it displays the entered details along with the result (Pass/Fail) to the user.

Output:



```
<terminated> Student (1) [Java Application] C:\Users\Aayush-PC\Downloads\ler
Enter student ID:
201
Enter student name:
Saurabh Mishra
Enter student mark:
45
Student ID: 201
Student Name: Saurabh Mishra
Student Mark: 45.0
Result: Pass
```

Lab Assignment: 2

Question:

4) Write a Java program that creates a account classs with instance variable accno,accname,amount and instance method withdraw, deposit, and interest. Create object of account class test all methods.

```
1 package assignment2;
2
3 import java.util.Scanner;
4
5 public class Account {
6     int accNo;
7     String accName;
8     double amount;
9     public Account(int accNo, String accName, double amount) {
10         this.accNo = accNo;
11         this.accName = accName;
12         this.amount = amount;
13     }
14     public void withdraw(double withdrawalAmount) {
15         if (withdrawalAmount > 0 && withdrawalAmount <= amount) {
16             amount -= withdrawalAmount;
17             System.out.println("Withdrawal successful. "
18                 + "Current balance: " + amount);
19         } else {
20             System.out.println("Invalid withdrawal amount "
21                 + "or insufficient balance.");
22         }
23     }
24     public void deposit(double depositAmount) {
25         if (depositAmount > 0) {
26             amount += depositAmount;
27             System.out.println("Deposit successful. "
28                 + "Current balance: " + amount);
29         } else {
30             System.out.println("Invalid deposit amount.");
31         }
32     }
33     public double calculateInterest(double rate) {
34         return amount * rate / 100;
35     }
36     public static void main(String[] args) {
37         Scanner scanner = new Scanner(System.in);
38
39         System.out.println("Enter account number:");
40         int accNo = scanner.nextInt();
41         scanner.nextLine(); // Consume newline
42
43         System.out.println("Enter account name:");
44         String accName = scanner.nextLine();
45
46         System.out.println("Enter initial amount:");
47         double amount = scanner.nextDouble();
48
49         Account account = new Account(accNo, accName, amount);
50
51         System.out.println("Enter withdrawal amount:");
52         double withdrawAmount = scanner.nextDouble();
53         account.withdraw(withdrawAmount);
54
55         System.out.println("Enter deposit amount:");
56         double depositAmount = scanner.nextDouble();
57         account.deposit(depositAmount);
58
59         System.out.println("Enter interest rate:");
60         double interestRate = scanner.nextDouble();
61         double interest = account.calculateInterest(interestRate);
62         System.out.println("Interest earned: " + interest);
63         scanner.close();
64     }
65 }
```

Lab Assignment: 2

Class Definition: The Account class is defined with attributes (accNo, accName, amount) and methods (withdraw, deposit, calculateInterest). Constructor initializes the account attributes.

Main Method:

The main method initializes a Scanner object to read user input.

It prompts the user to enter account details such as account number, account name, and initial amount.

It creates an Account object with the provided details.

Then, it prompts the user to enter withdrawal and deposit amounts, followed by the interest rate.

It calls methods to perform withdrawal, deposit, and interest calculation based on user input.

Finally, it prints the calculated interest earned.

User Input:

The program waits for user input for account details, withdrawal amount, deposit amount, and interest rate.

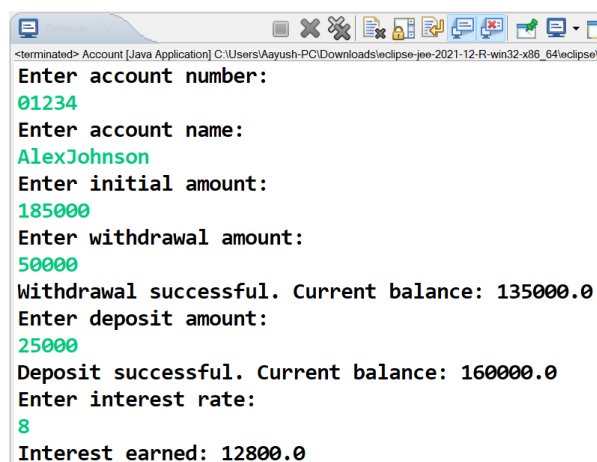
Output:

After each withdrawal or deposit operation, it prints the current balance.

At the end, it prints the interest earned.

Closing Scanner: The program closes the Scanner object to release system resources after completing the input reading process.

Output:



```
<terminated> Account [Java Application] C:\Users\Aayush-PC\Downloads\eclipse-jee-2021-12-R-win32-x86_64\eclipse.exe
Enter account number:
01234
Enter account name:
AlexJohnson
Enter initial amount:
185000
Enter withdrawal amount:
50000
Withdrawal successful. Current balance: 135000.0
Enter deposit amount:
25000
Deposit successful. Current balance: 160000.0
Enter interest rate:
8
Interest earned: 12800.0
```

Lab Assignment: 2

Question:

5. Write a Java program to create a class called player with name, age, country Name, total run as instance member. Create 5 player objects and write instance method to display the details of Player having more than 5000 as total run.

```
1 package assignment2;
2
3 import java.util.ArrayList;
4
5 public class Player {
6     private String name;
7     private int age;
8     private String countryName;
9     private int totalRun;
10    public Player(String name, int age, String countryName, int totalRun) {
11        this.name = name;
12        this.age = age;
13        this.countryName = countryName;
14        this.totalRun = totalRun;
15    }
16    public void displayPlayerDetails() {
17        System.out.println("Name: " + name);
18        System.out.println("Age: " + age);
19        System.out.println("Country: " + countryName);
20        System.out.println("Total Runs: " + totalRun);
21        System.out.println();
22    }
23    public static void main(String[] args) {
24
25        Player player1 = new Player("R Ponting", 40, "AUS ", 11736);
26        Player player2 = new Player("D Warner 2", 34, "AUS", 6817);
27        Player player3 = new Player("S Waugh 3", 36, "AUS", 4800);
28        Player player4 = new Player("M Clarke 4", 35, "AUS", 4655);
29        Player player5 = new Player("S Smith 5", 34, "AUS", 13386);
30
31        ArrayList<Player> players = new ArrayList<>();
32        players.add(player1);
33        players.add(player2);
34        players.add(player3);
35        players.add(player4);
36        players.add(player5);
37
38        System.out.println("Players with more than 5000 total runs:");
39        for (Player player : players) {
40            if (player.totalRun > 5000) {
41                player.displayPlayerDetails();
42            }
43        }
44    }
45 }
```

Class Definition: The Player class is defined with attributes for name, age, country name, and total runs. It includes a constructor to initialize these attributes and a method displayPlayerDetails to print player details.

Main Method: The main method is the entry point of the program.

Five Player objects are created with specified attributes.

Lab Assignment: 2

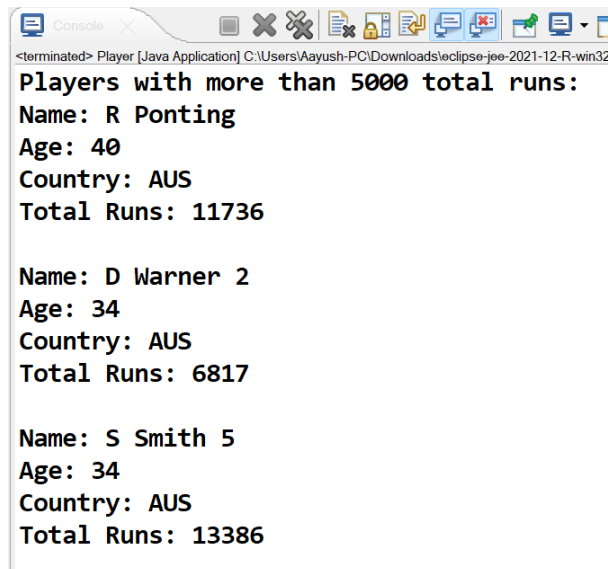
An ArrayList is used to store these Player objects.

The program prints a message indicating it will display players with more than 5000 total runs.

A loop iterates over the ArrayList and calls displayPlayerDetails for each player whose total runs exceed 5000.

Output: The program will output the details of players who have scored more than 5000 runs, displaying their name, age, country, and total runs.

Output :



```
<terminated> Player [Java Application] C:\Users\Aayush-PC\Downloads\eclipse-jee-2021-12-R-win32
Players with more than 5000 total runs:
Name: R Ponting
Age: 40
Country: AUS
Total Runs: 11736

Name: D Warner 2
Age: 34
Country: AUS
Total Runs: 6817

Name: S Smith 5
Age: 34
Country: AUS
Total Runs: 13386
```