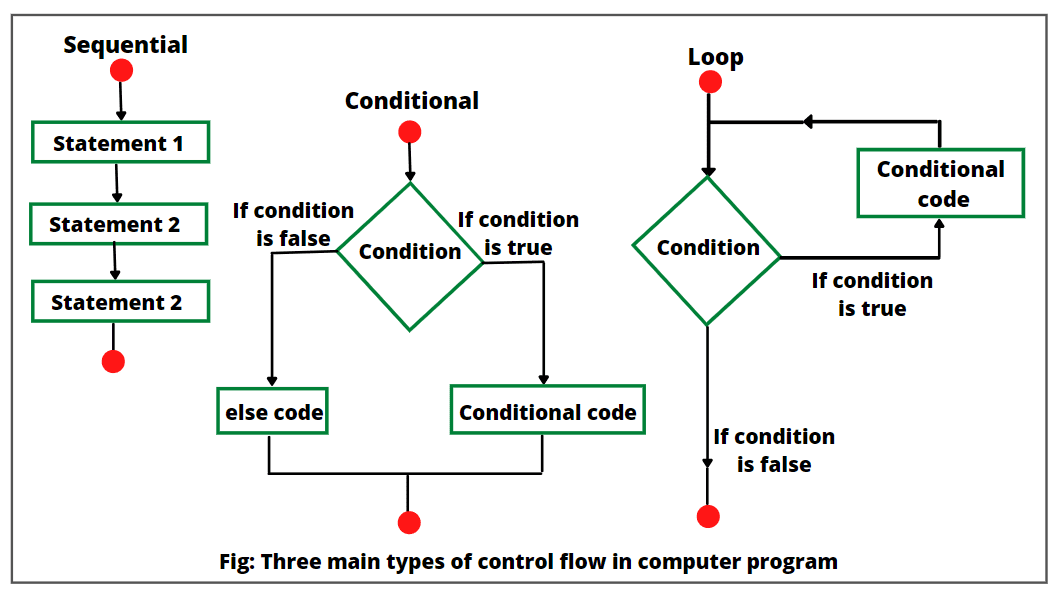
|  |  |
| --- | --- |
| **Name – Prerna Sunil Jadhav** | **SAP ID - 60004220127** |

**Experiment No - 02**

**AIM: TO IMPLEMENT JAVA CONTROL STATEMENTS AND LOOPS**

**THEORY:**

* Java compiler executes the code from top to bottom. The statements in the code are executed according to the order in which they appear.
* Java provides statements that can be used to control the flow of Java code. Such statements are called control flow statements.
* It is one of the fundamental features of Java, which provides a smooth flow of program.
* Java provides three types of control flow statements.
  + Decision Making statements
    - if statements
    - switch statement
  + Loop statements
    - do while loop
    - while loop
    - for loop
    - for-each loop
  + Jump statements
    - break statement
    - continue statement

**PROGRAM 1:** Write A Program to find roots of a Quadratic equation. Take care of imaginary values.

**THEORY:**

The standard form of a quadratic equation is: ax2 + bx + c = 0

Here, a, b, and c are real numbers and a can't be equal to 0.

We can calculate the root of a quadratic by using the formula: x = (-b ± √(b2-4ac)) / (2a)

The ± sign indicates that there will be two roots:

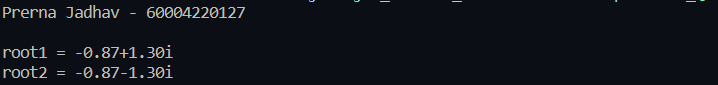
* root1 = (-b + √(b2-4ac)) / (2a)
* root1 = (-b - √(b2-4ac)) / (2a)

The term b2-4ac is known as the determinant of a quadratic equation. It specifies the nature of roots. That is,

* if determinant > 0, roots are real and different
* if determinant == 0, roots are real and equal
* if determinant < 0, roots are complex and different

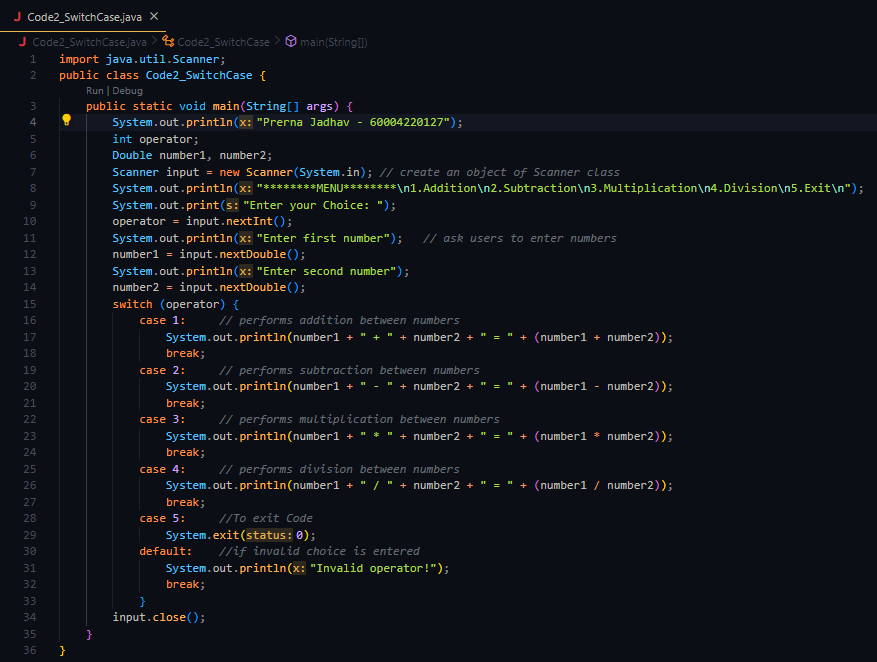
**CODE:**

**OUTPUT:**

****

**PROGRAM 2:** Write a menu driven program using switch case to perform mathematical operations.

**CODE:**

****

**OUTPUT:**

**Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated**

**THEORY:**

Here, we have used the Scanner class to take 3 inputs from the user.

operator - specifies the operation to be performed

number1/number2 - operands to perform operation on

Since the operator matches the case 3, so the corresponding codes are executed.

System.out.println(number + " \* " + number2 + " = " +( number1 \* number2));

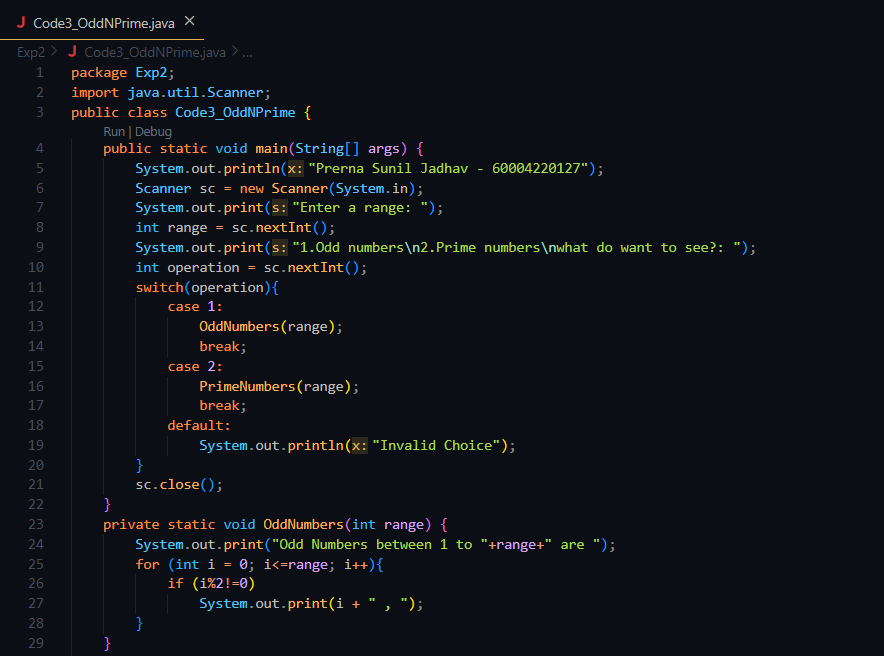
break;

These statements compute the product of two numbers and print the output. Finally, the break statement ends the switch statement.

Similarly, for different operators, different cases are executed.

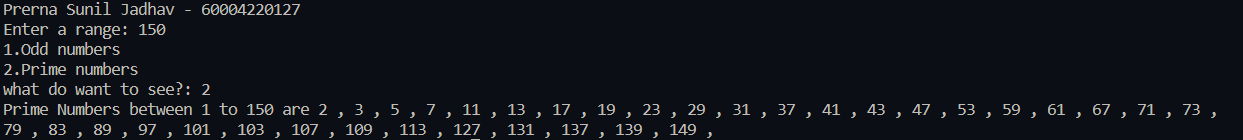
**PROGRAM 3:** Write A Program to display odd numbers from given range/ prime numbers from given range.

**CODE**



A screenshot of a computer

Description automatically generated with medium confidence

**A screenshot of a computer

Description automatically generated with medium confidenceOUTPUT:**

**THEORY:**

For Odd:

* Firstly, consider the given number N as input.
* Then apply a for loop in order to iterate the numbers from 1 to N.
* At last, check if each number is a odd number using the modulus and if it’s a odd number then print it

For Prime:

* Firstly, consider the given number N as input.
* Then apply a for loop in order to iterate the numbers from 1 to N.
* At last, check if each number is a prime number by checking if that number is divisible by any other number other than 1 and itself and if it’s a prime number then print it

**PROGRAM4:** Write A Program to display default value of primitive data types

**CODE**

**Text

Description automatically generated**

**OUTPUT:**

**A screenshot of a computer

Description automatically generated**

**THEORY:**

Primitive types are the Java data types used for data manipulation, for example, int, char, float, double, boolean, etc.

* byte: An 8-bit signed two’s complement integer (128 – 127)
* short: A 16-bit signed two’s complement integer (-32768 – 32767)
* int: A 32-bit signed two’s complement integer (-2,147,483,648 - 2,147,483,647)
* long: A 64-bit two’s complement integer (-9,223,372,036,854,775,808 - 9,223,372,036,854,775,807)
* char: A single 16-bit Unicode character. (‘\u0000’ (or 0) - ‘\uffff’)
* float: A single-precision 32-bit IEEE 754 floating point (1.4E-45 - 3.4028235E38)
* double: A double-precision 64-bit IEEE 754 floating point (4.9E-324 - 1.7976931348623157E308)
* boolean: Possible values, TRUE and FALSE.

**PROGRAM5A:** Write A Program to display the following patterns:

**1**

**2 1**

**1 2 3**

**4 3 2 1**

**1 2 3 4 5**

**6 5 4 3 2 1**

**1 2 3 4 5 6 7**

**CODE**



**A screenshot of a computer

Description automatically generated with medium confidenceOUTPUT:**

**THEORY:**

Here we started a loop till n, here its 7 And will check for every iteration of n if its even if its even reverse the order to print else print sequentially.

**PROGRAM5B:** Write A Program to display the following patterns:

A

C B

F E D

J I K G

**CODE:**

**Text

Description automatically generated**

**OUTPUT:**

**A picture containing shape

Description automatically generated**

**THEORY:**

Here we Initially assume the character to be printed 64 i.e., character before A and will add 1, 2, 3 ... in our Ascii value as our row increments and print the sequence in reverse by decrementing the duplicate value by 1 in inner for loop.

**CONCLUSION:**

Primitive data types are the building blocks of data manipulation.

For statement consumes the initialization, condition, and increment/decrement in one line thereby providing a shorter, easy to debug

structure of looping.

If a loop exists inside the body of another loop, it's called a nested loop.

That is why nested loops are also called as “loop inside loop“.

The Java switch statement only works with:

* Primitive data types: byte, short, char, and int
* Enumerated types
* String Class
* Wrapper Classes: Character, Byte, Short, and Integer.