Decision Making Statement

A programming language uses control statements to control the flow of execution of program based on certain conditions. These  are used to cause the flow of execution to advance and branch based on changes to the state of a program.

**Java’s Selection statements:-**

**1.if statement**

**2.if-else statement**

**3.nested-if statement**

**4.if-else-if statement**

**5.switch-case statement**

**6.jump(break, continue, return) statement**

**1.If statement:-**

if statement is the most simple decision making statement. It is used to decide whether a certain statement or block of statements will be executed or not i.e. if a certain condition is true then a block of statement is executed otherwise not.

**Syntax:-**

if(condition)

{

/\* Statements will be executed if the condition(**Boolean expression**) is true.\*/

}

**Flow chart:-**



**// Example of if statement**

class Ifdemo

{

    public static void main(String args[])

    {

        int i = 10;

        if (i < 15)

{

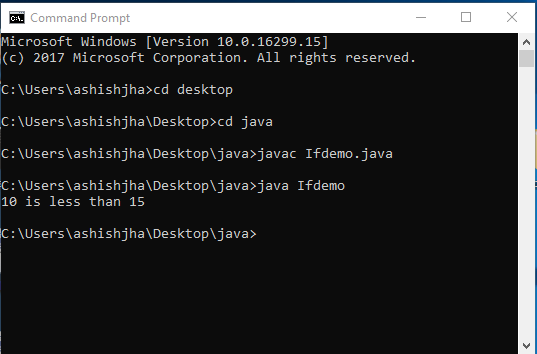
            System.out.println("10 is less than 15");

 }

}

}

**Output:-**



**2.if-else statement:-**

The if statement alone tells us that if a condition is true it will execute a block of statements and if the condition is false it won’t. But what if we want to do something else if the condition is false. Here comes the else statement. We can use the else statement with if statement to execute a block of code when the condition is false.

**Syntax:-**

if(condition)

{

/\* Statements will be executed if the condition(**Boolean expression**) is true.\*/

}

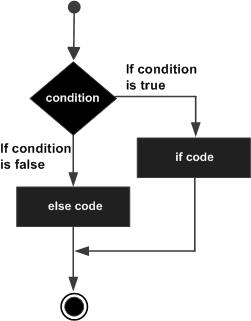
else

{

/\* Statements will be executed if the condition(**Boolean expression**) is false.\*/

}

**Flow Statement:-**



**// Example of if–else statement.**

class Ifelsedemo

{

    public static void main(String args[])

    {

        int i = 10;

        if (i < 15)

{

            System.out.println(i+” is smaller than 15");

}

        else

{

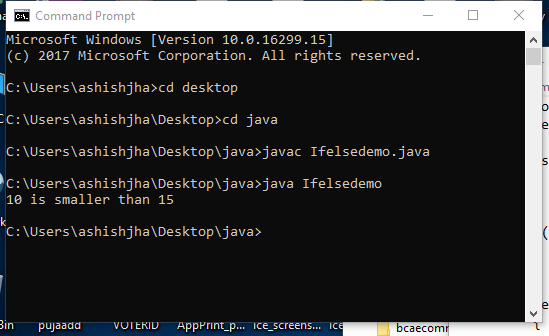
            System.out.println(i+” is greater than 15");

}

    }

}

**Output:-**



**3.nested-if statement :-**

A nested if is an if statement that is the target of another if or else. Nested if statements means an if statement inside an if statement. Yes, java allows us to nest if statements within if statements. i.e, we can place an if statement inside another if statement.

**Syntax:-**

If(condition1)

{

/\* Statements will be executed if the condition(**Boolean expression**) is true.\*/

if(condition2)

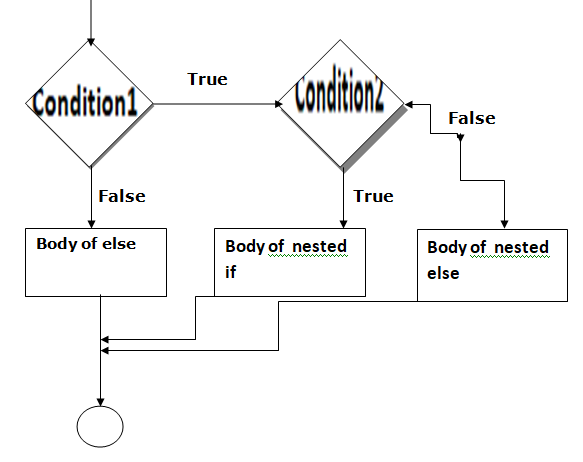
{

/\* Statements will be executed if the condition(**Boolean expression**)is true.\*/

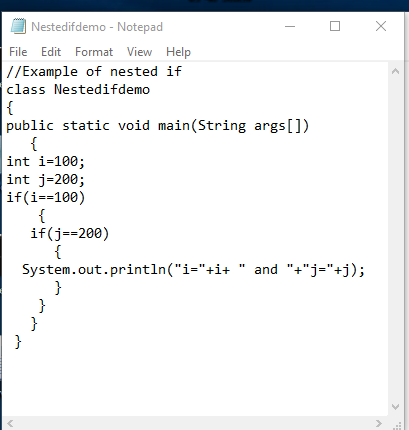
}

}

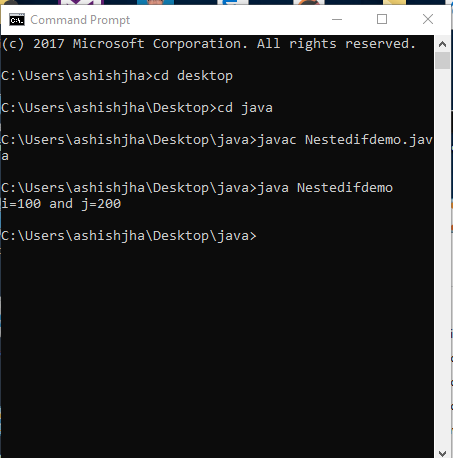
***Flow Chart:-***



**Example:-**



**Output:-**

****

**4.if-else-if statement:-**

The if statements are executed from the top down. As soon as one of the conditions controlling the if is true, the statement associated with that if is executed, and the rest of the ladder is bypassed. If none of the conditions is true, then the final else statement will be executed.

**Syntax:-**

if(condition1)

{

/\* Statements will be executed if the condition1(**Boolean expression**) is true.\*/

}

else if(condition2)

{

/\* Statements will be executed if the condition2(**Boolean expression**) is true.\*/

}

else if(condition3)

{

/\* Statements will be executed if the condition3(**Boolean expression**) is true.\*/

}

**.**

**.**

**.**

**.**

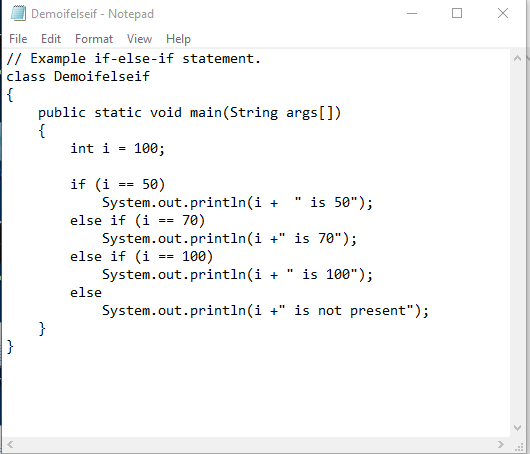
else

{

// Statement will be executed if above the condition is not true.

}

**Example:-**



**Output:-**

