

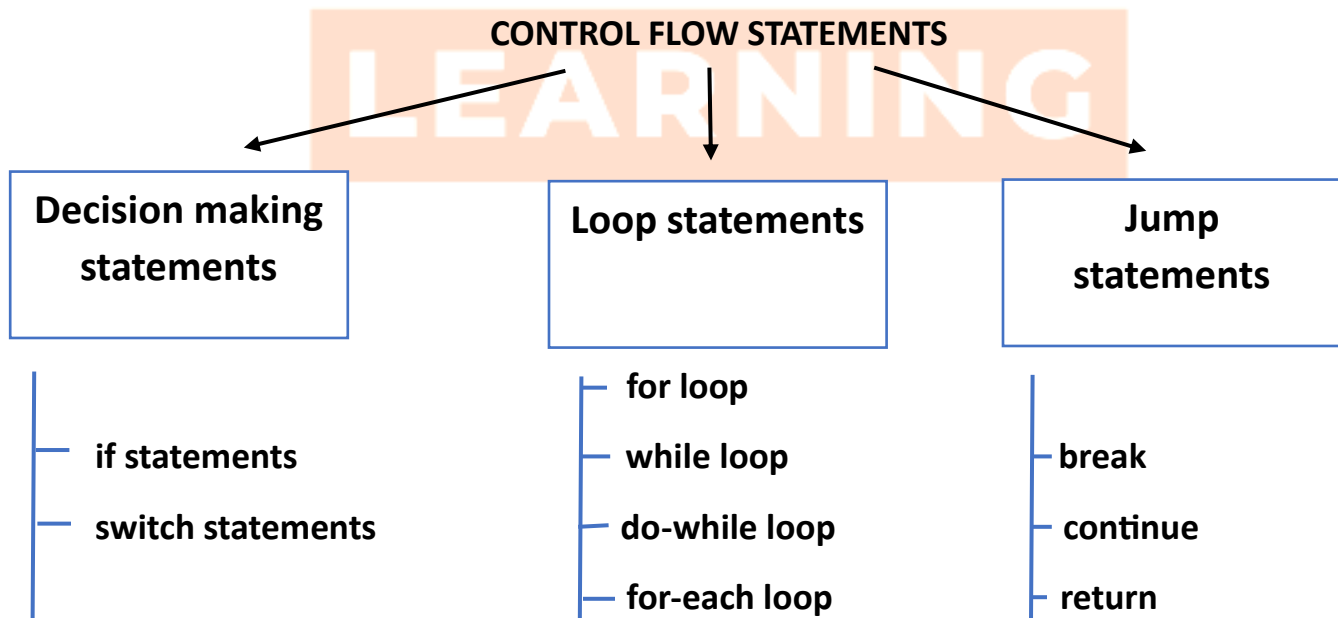
CORE JAVA

JAVA CONTROL STATEMENTS-

Java control statements are essential components of the language that allow developers to dictate the flow of program execution. These statements enable you to execute certain parts of your code based on specific conditions, loop through code, and handle exceptions.

Control flow statements have multiple purposes:

1. We use control flow statement if we want to execute a specific statement based on some condition. For example- Display user data if age is greater than 18.
2. Or, if we want to execute some statements repeatedly. For example- table of a number.
3. Or, to handle exceptions.



Decision making statements-

Decision-making statements in Java allow you to control the flow of execution based on conditions.

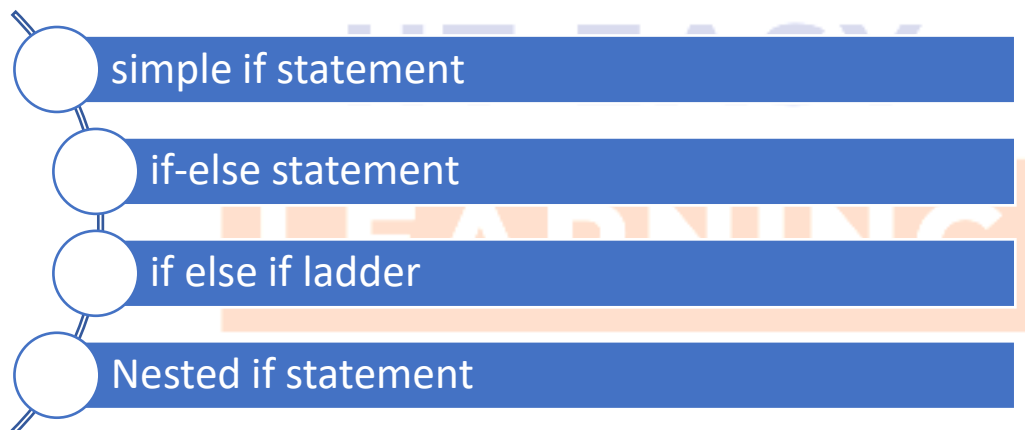
Basically, based on some condition your program will decide which statement to execute and which to not.

For example, we put a condition that if age is greater than 18 then print- “eligible to vote” and if less than 18 then print- “Not eligible to vote”.

There are two types of decision-making statements in Java-

1. if statements
2. switch statements

Types of if statements-

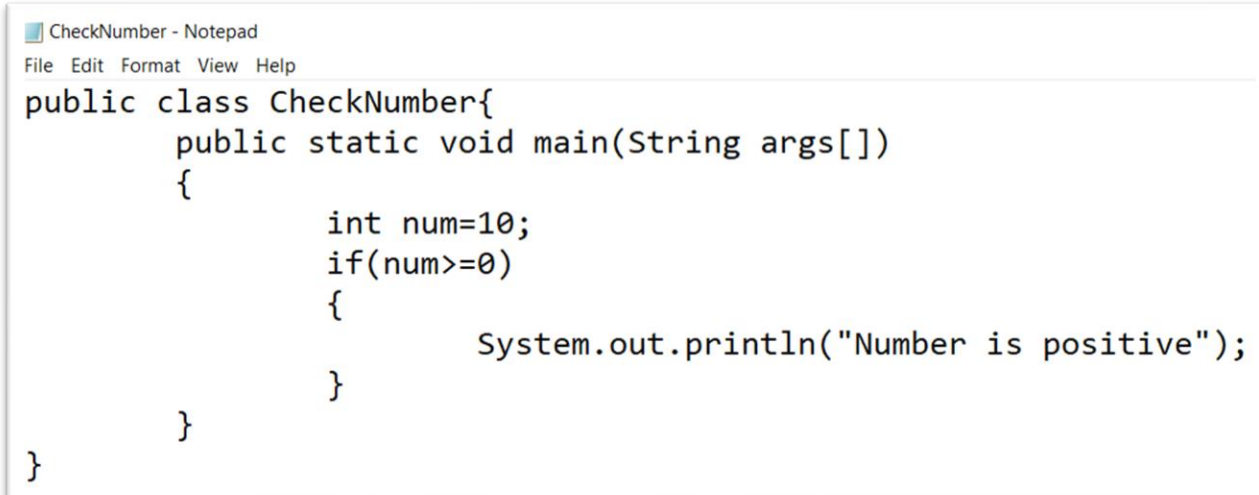


1.if statements-

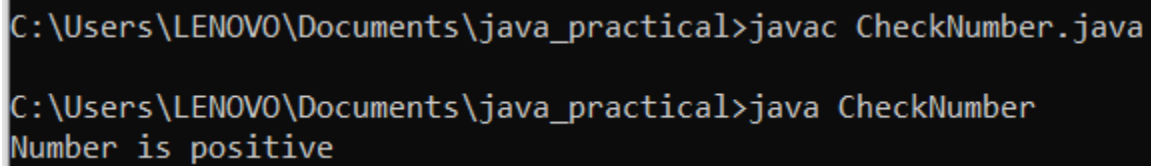
The if statement executes a block of code if a specified condition is true.

Syntax-

```
if (condition) {  
    // code to be executed if condition is true  
}
```

Practical 1: To check if the given number is positive. (using simple if statement)

```
public class CheckNumber{
    public static void main(String args[])
    {
        int num=10;
        if(num>=0)
        {
            System.out.println("Number is positive");
        }
    }
}
```

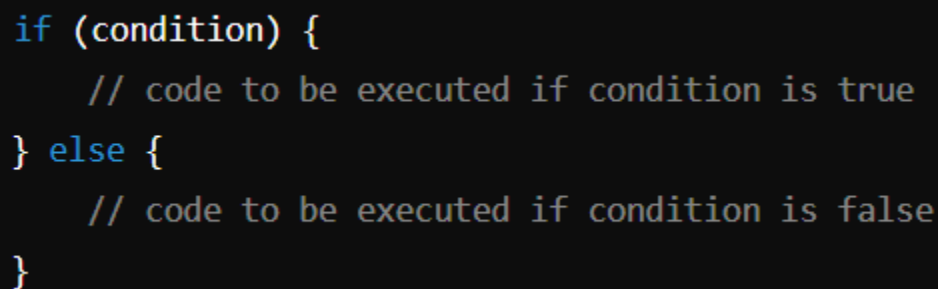
Output-

```
C:\Users\LENOVO\Documents\java_practical>javac CheckNumber.java

C:\Users\LENOVO\Documents\java_practical>java CheckNumber
Number is positive
```

b) if-else statements-

The if-else statement executes one block of code if a condition is true and another block if the condition is false.

Syntax-

```
if (condition) {
    // code to be executed if condition is true
} else {
    // code to be executed if condition is false
}
```

Practical 2: To check if a given number is positive or not (using if-else statement)

```
CheckNumber2 - Notepad
File Edit Format View Help
// Checking number is positive or not using if else statement
public class CheckNumber2
{
    public static void main(String args[])
    {
        int num=10;
        if(num>=0)
        {
            System.out.println("Number is positive");
        }
        else{
            System.out.println("Number is not positive");
        }
    }
}
```

Output-

```
C:\Users\LENOVO\Documents\java_practical>javac CheckNumber2.java
C:\Users\LENOVO\Documents\java_practical>
C:\Users\LENOVO\Documents\java_practical>java CheckNumber2
Number is positive
```

Practical 3: To check which number is greater out of the two numbers. (using if else statement)

```
CheckNumber3 - Notepad
File Edit Format View Help
// Checking which number is greater between two numbers using if else statement
public class CheckNumber3
{
    public static void main(String args[])
    {
        int num1=5;
        int num2=10;
        if(num1>num2)
        {
            System.out.println("Number 1 is greater");
        }
        else{
            System.out.println("Number 2 is greater");
        }
    }
}
```

Output-

```
C:\Users\LENOVO\Documents\java_practical>javac CheckNumber3.java
C:\Users\LENOVO\Documents\java_practical>java CheckNumber3
Number 2 is greater
```

Suppose in the above question both the numbers are equal having same value, then what will be the output?

```
CheckNumber3 - Notepad
File Edit Format View Help
// Checking which number is greater between two numbers using if else statement
public class CheckNumber3
{
    public static void main(String args[])
    {
        int num1=10;
        int num2=10;
        if(num1>num2)
        {
            System.out.println("Number 1 is greater");
        }
        else{
            System.out.println("Number 2 is greater");
        }
    }
}
```

OUTPUT-

```
C:\Users\LENOVO\Documents\java_practical>javac CheckNumber3.java  
C:\Users\LENOVO\Documents\java_practical>java CheckNumber3  
Number 2 is greater
```

Explanation-

- Since both num1 and num2 are equal to 10, the condition num1 > num2 is false.
- As the if condition is false, the code enters the else block.
- The statement System.out.println("Number 2 is greater"); is executed, printing "Number 2 is greater" to the console.

To handle this kind of situation we use if-else ladder where one more condition is added to check if the numbers are equal.

c) if-else if statements or if-else if ladder -

The if-else-if ladder is used to test multiple conditions. When a condition is true, the corresponding block of code is executed, and the rest of the ladder is skipped. If a condition is false, then the next condition is checked and this process continues until all conditions are checked and if none of the condition satisfies then the else block is executed.

Syntax-

```
if (condition1) {  
    // code to be executed if condition1 is true  
} else if (condition2) {  
    // code to be executed if condition2 is true  
} else if (condition3) {  
    // code to be executed if condition3 is true  
} else {  
    // code to be executed if all conditions are false  
}
```

Practical 4: Check if a number is positive, negative or zero. (using if else if ladder)

```
CheckNumber4 - Notepad
File Edit Format View Help
// Check a number if it is positive, negative or zero
public class CheckNumber4
{
    public static void main(String args[])
    {
        int num=0;
        if(num>0)
        {
            System.out.println("Number is positive");
        }
        else if(num==0)
        {
            System.out.println("Number is equal to 0");
        }
        else{
            System.out.println("Number is negative");
        }
    }
}
```

Output-

```
C:\Users\LENOVO\Documents\java_practical>java CheckNumber4
Number is equal to 0
```

Practical 5: Check between two numbers if they are equal or one is greater than the other.

```
CheckNumber5 - Notepad
File Edit Format View Help
// Check between two numbers if they are equal or one is greater than the other.
public class CheckNumber5
{
    public static void main(String args[])
    {
        int a=10;
        int b=10;
        if(a>b)
        {
            System.out.println("a is greater than b");
        }
        else if(b>a)
        {
            System.out.println("b is greater than a");
        }
        else{
            System.out.println("a and b are equal");
        }
    }
}
```

Output-

```
C:\Users\LENOVO\Documents\java_practical>javac CheckNumber5.java

C:\Users\LENOVO\Documents\java_practical>java CheckNumber5
a and b are equal
```


d) Nested-if statements-

In Java, you can use nested if statements to test multiple conditions within each other. This means placing one if statement inside another if or else statement. Nested if statements allow you to handle more complex decision-making scenarios.

Syntax-

```
if (condition1) {  
    // code to be executed if condition1 is true  
    if (condition2) {  
        // code to be executed if condition2 is also true  
    } else {  
        // code to be executed if condition2 is false  
    }  
} else {  
    // code to be executed if condition1 is false  
    if (condition3) {  
        // code to be executed if condition3 is true  
    } else {  
        // code to be executed if condition3 is false  
    }  
}
```

A logo consisting of a white capital letter 'G' centered within an orange square.

Practical 6: To check if a number is not zero, if it is not zero then check if it is positive or negative or else print if it is equal to zero.

```
CheckNumber6 - Notepad
File Edit Format View Help
//nested if-else statement
public class CheckNumber6
{
    public static void main(String args[])
    {
        int a=10;
        if(a!=0)
        {
            if(a>0)
            {
                System.out.println("Number is positive");
            }
            else{
                System.out.println("Number is negative");
            }
        }
        else{
            System.out.println("Number is equal to 0");
        }
    }
}
```

Output-

```
C:\Users\LENOVO\Documents\java_practical>javac CheckNumber6.java
C:\Users\LENOVO\Documents\java_practical>java CheckNumber6
Number is positive
```

Nested if statements are particularly useful when you need to make decisions based on multiple conditions that depend on each other.

Example- Validating complex user input (e.g., if a username exists, then check if the password is correct).

Join our growing community of tech enthusiasts! Subscribe to our YouTube channel, where we simplify programming languages in the easiest way possible. Gain a deeper understanding with our clear explanations and receive exclusive notes to enhance your learning journey. Don't miss out on valuable insights – hit that subscribe button now and embark on a programming adventure with us!

Subscribe to our channel:

<https://www.youtube.com/@HTEASYLEARNING>

HT EASY

LEARNING