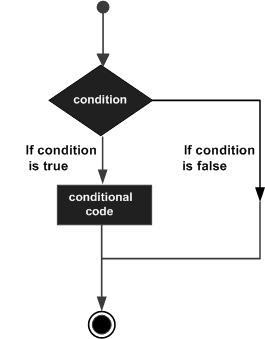
**JAVA DAY 4 WEEK 1 DECISION CONTROL**

When we need to execute a set of statements based on a condition then we need to use control flow statements. For example, if a number is greater than zero then we want to print “Positive Number” but if it is less than zero then we want to print “Negative Number”. In this case we have two print statements in the program, but only one print statement executes at a time based on the input value. We will see how to write such type of conditions in the java program using control statements.

Decision making structures have one or more conditions to be evaluated or tested by the program, along with a statement or statements that are to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false.

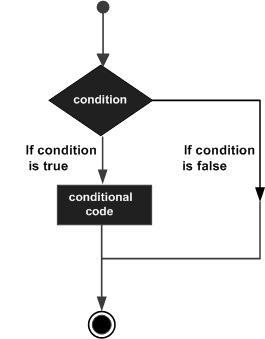
The general form of a typical decision-making structure used in java looks like the one below:



**decision-making structure**

**If Statement**

An**if statement** consists of a Boolean expression followed by one or more statements.



***If Statement***

**Syntax**

The java syntax for if statement is as follows:

if(Boolean\_expression) {

     Statements; // Statements will execute if the Boolean expression is true

   }

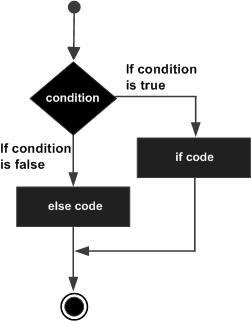
If the Boolean expression evaluates to **true** then the block of code inside the if statement (Statements) **will be executed.**

If the Boolean expression evaluates to **false**, then the block of code inside the if statement (Statements) **will not be executed**

**IF ELSE statement**

An if statement can be followed by an optional else statement, which executes when the Boolean expression is false. The diagram below shows the flow of an if else statement.

As you can see the are two code blocks to be executed, the block for if the condition is true and a block for when the condition false.



**Syntax**

The java syntax for **if else statement** is as follows:

if (num < 50) {

   System.out.println("num is less than 50”);

      }

  else {

   System.out.println("num is greater than or equal 50");

       }

From the above code segment

* if num is**less** than 50 then **num is less than 50** will be printed
* if num is **greater** than 50 then **num is greater than 50** will be printed

**Nested If Statement**

An if statement can be followed by an optional else if...else statement, which is very useful to test various conditions using single if...else if statement. You have as many else if as the number of you possible conditions

When using if, else if, else statements there are a few points to keep in mind.

* An if can have zero or one else's and it must come after any else if's.
* An if can have zero to many else if's and they must come before the else.
* Once an else if succeeds, none of the remaining else if's or else's will be tested.

**Syntax**

if( x == 10 ) {

   System.out.print("Value of X is 10");

}else if( x == 20 ) {

   System.out.print("Value of X is 20");

}else if( x == 30 ) {

   System.out.print("Value of X is 30");

}else {

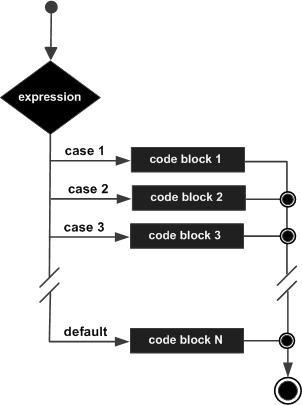
   System.out.print("This is else statement");

**SWEITCH STATEMENT**

**Switch statement**

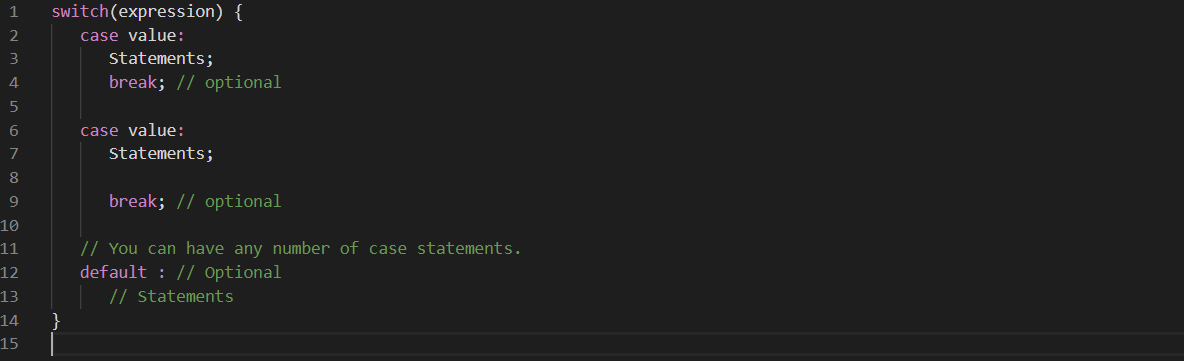
A switch statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case.

Switch case statement is used when we have number of options (or choices) and we may need to perform a different task for each choice.

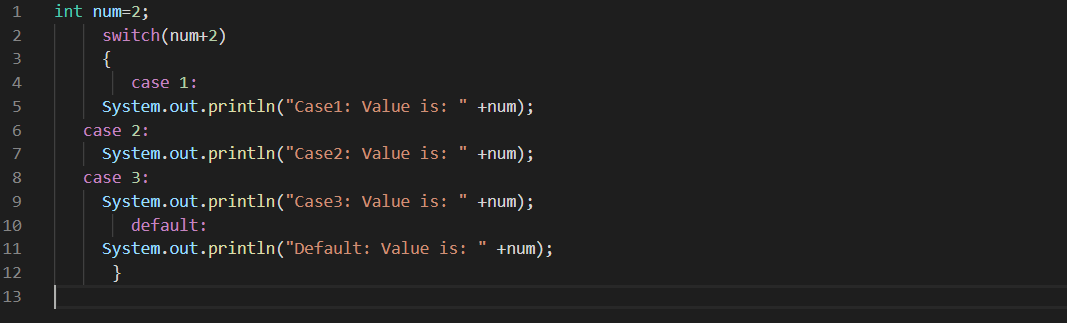


General rules to a switch statement are:

* The variable used in a switch statement can only be integers, convertible integers (byte, short, char), strings and enums.
* You can have any number of case statements within a switch. Each case is followed by the value to be compared to and a colon.
* The value for a case must be the same data type as the variable in the switch and it must be a constant or a literal.
* When the variable being switched on is equal to a case, the statements following that case will execute until a break statement is reached.
* When a break statement is reached, the switch terminates, and the flow of control jumps to the next line following the switch statement.
* Not every case needs to contain a break. If no break appears, the flow of control will fall through to subsequent cases until a break is reached.
* A switch statement can have an optional default case, which must appear at the end of the switch. The default case can be used for performing a task when none of the cases is true. No break is needed in the default case.



**Example**



**The tenary (?) operator**

The tenary operator only caters for two outcomes. Like the IF ELSE statement evaluates for the true or false only. It can only be used in place of if else statement.

**Syntax**

(expression1)? expression2: expression3;

Where expression1, expression2, and expression3 are expressions. Notice the use and placement of the colon.

* To determine the value of the whole expression, initially exp1 is evaluated.
* If the value of exp1 is true, then the value of Exp2 will be the value of the whole expression.
* If the value of exp1 is false, then Exp3 is evaluated and its value becomes the value of the entire expression.