

Decision-making

Decision-making is about deciding the order of execution of statements based on certain conditions. When you write a computer program, regardless of the programming language, you often need to execute different set of statements depending on some satisfied condition. The process of determining the order in which statements execute in a program is called decision-making or flow of Control. Java language handles decision-making by supporting the following statements.

- *if* statement
- *if, else* statement
- *if, else if, else* statement
- *switch, case, default* statement
- **conditional operator** statement

❖ Decision making with *if* statement

1. *if* statements

The *if* statement may be implemented in different forms depending on the evaluation of a conditional expression return a value of **True**. The syntax for a simple *if* statement is as follows:

Syntax: For one statement without block or curly braces.

Flowchart of *if* statement

if (conditional expression)

Statement;

Syntax: For block or more than one statement.

if (conditional expression)

```
{ Statement1; Statement2;
  :
}
```

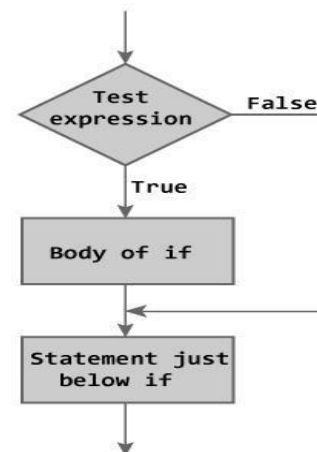


Figure: Flowchart of *if* Statement

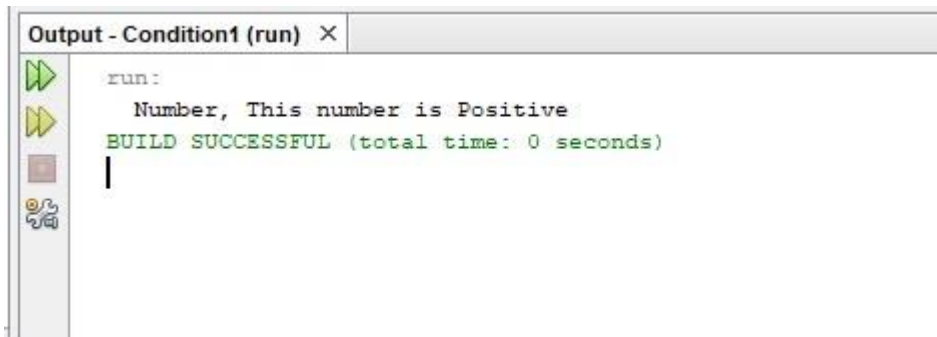
If the expression is evaluated and found to be true, the single statement following the "*if*" is executed. If false, the following statement is skipped. Here a compound statement composed of several statements bounded by braces can replace the single statement.

Example *if* condition is True than execute For one statements.

```
int Number = 2;
```

```
if ( Number>0 )
```

```
    System.out.println(Number+" This number is Positive");
```



```
Output - Condition1 (run) X
run:
    Number, This number is Positive
BUILD SUCCESSFUL (total time: 0 seconds)
```

Example *if* condition is true than execute for block or more than one statements.

```
int Roll=21;
```

```
if ( Roll==21)
```

```
{
```

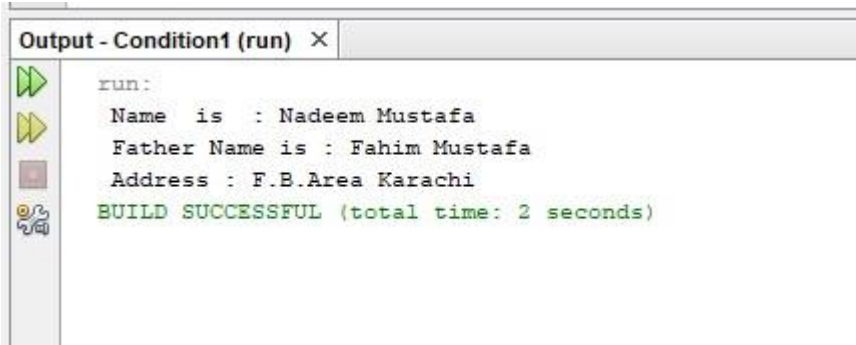
```
    System.out.println(" Name is : Nadeem Mustafa ");
```

```
    System.out.println(" Father Name is : Fahim Mustafa");
```

```
    System.out.println(" Address : F.B.Area Karachi");
```

```
}
```

Output



```
Output - Condition1 (run) X
run:
    Name is : Nadeem Mustafa
    Father Name is : Fahim Mustafa
    Address : F.B.Area Karachi
BUILD SUCCESSFUL (total time: 2 seconds)
```

2. if, else statement:

In this condition the programmer to write a single comparison, and then execute one of the two statements depending upon whether the test expression is true or false. The general form of the **if else** statement is.

Flowchart of**Syntax:**

```
if( expression )
    statement1 ;
else
    statement2 ;
```

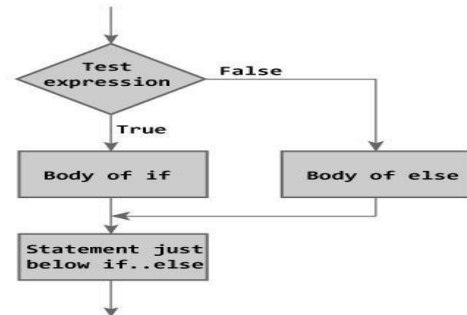


Figure: Flowchart of if...else Statement

Here also expression in parentheses must evaluate to (a Boolean) true or false. Typically you are testing something to see if it's true, and then running a code block (one or more statements) if it is true, and another block of code if it isn't. The statement1 or statement2 can be either simple or compound statement.

The following program demonstrates a legal if else statement:

```
int number=0; //Number stored@ "number"
if( number == 0 ) //Checks whether number you entered is equal to 0 or Not.
    System.out.println("Given number is Zero "); //if YES then enters into if-body
else //If not then enters into else -body
    System.out.println("Given number is not zero ");
```

```

Output - Condition1 (run) x
run:
  Number, This number is Positive
BUILD SUCCESSFUL (total time: 0 seconds)
  
```

3. if .. else if.. else:

This brings up the other **if-else** construct; **if, else, if, else**. This construct is useful where two or more alternatives are available for selection.

Syntax:

```
if (condition)
    statement 1;
else if (condition)
    statement 2;
else if(condition)
    statement 3;
else
    statement 4;
```

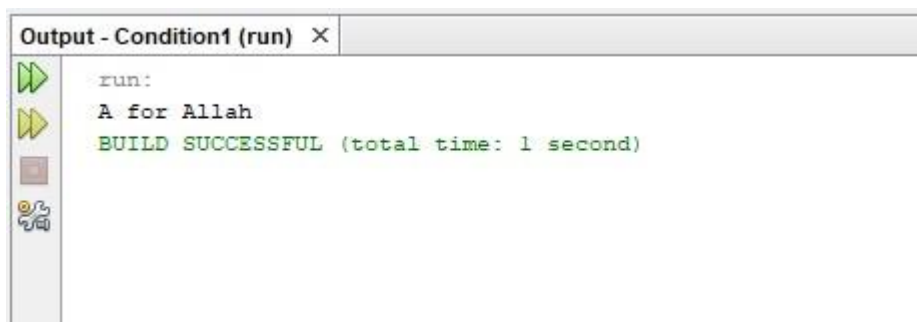
The various conditions are evaluated one by one starting from top to bottom, on reaching a condition evaluating to true the statement group associated with it are executed and skip other statements. If none of expression is evaluate to true, then the statement or group of statement associated with the final **else** is executed.

The following program demonstrates a legal **if-else if-else** statement:

```
String alphabet = "A";
if( alphabet == "A" || alphabet == "a" )
    System.out.println("A for Allah ");
else if( alphabet == "B" || alphabet == "b" )
    System.out.println(" B for Bissmillah ");
else if( alphabet == "C" || alphabet == "c" )
    System.out.println(" C for Captain ");
else

    System.out.println ("Unknown Character");
```

Output will be depending on character value in alphabet variable: -



```
Output - Condition1 (run) X
run:
A for Allah
BUILD SUCCESSFUL (total time: 1 second)
```

✚ Difference between **if** and **else – if** is that :

If Statement	If Else if Statement
<pre>int per = 72; if(per >=80) System.out.println("Grade A+ "); if (per>=70) System.out.println("Grade A "); if (per>=60) System.out.println("Grade B "); if (per>=50) System.out.println("Grade C ");</pre>	<pre>int per = 67; if(per >=80) System.out.println("Grade A+ "); else if (per>=70) System.out.println("Grade A "); else if (per>=60) System.out.println("Grade B "); else if (per>=50) System.out.println("Grade C ");</pre>
<p>Output: Grade A Grade B Grade C</p> <p>// it will checks all conditions and execute all conditions that are true//</p>	<p>Output: Grade B</p> <p>// It stop when the condition is true (i.e. not check further) and then exit from the if – structure//</p>

Nested If Expression:

If there is another structure within **if** structure that is called nested **if** statement.

Syntax:

```
if( expression )
{
    if( expression1 )
    {
        statement block1;
    }
    else
    {
        statement block2;
    }
} ← Outer if ends here
else
{
    statement block3;
}
```

← Outer if
← Inner If

if 'expression' is false the '**statement-block3**' will be *executed*, otherwise it continues to perform the test for 'expression 1' . *If the 'expression 1' is true* the '**statement-block1**' is *executed* otherwise 'statement-block2' is executed.

Example

The **if-else** statement can also use to test for Nested conditions. The following example uses two conditions so that **if** the first test fails, we want to perform a second test before deciding what to do:

```
int n;
```

```
n=91;
```

```
if ( (n % 2) != 0)
```

```
    if ( n>10)
```

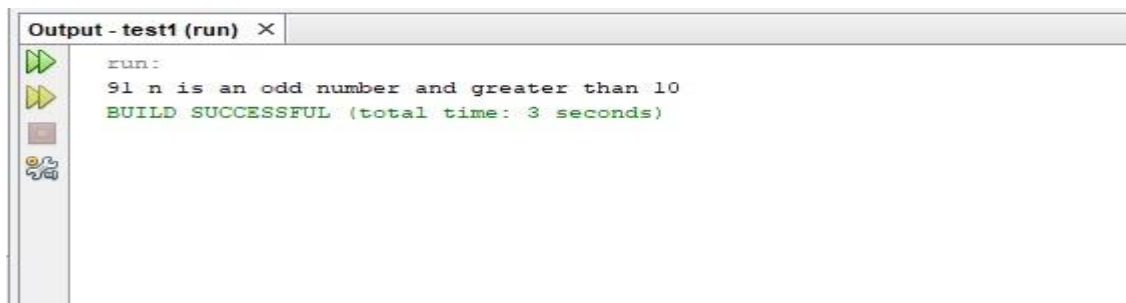
```
        System.out.println(n+" n is an odd number and greater than 10 ");
```

```
    else
```

```
        System.out.println(n+" n is an odd number and less than 10 ");
```

```
else
```

```
    System.out.println(n+" n is an even number");
```



```
Output - test1 (run) x
run:
91 n is an odd number and greater than 10
BUILD SUCCESSFUL (total time: 3 seconds)
```

4. Switch statement

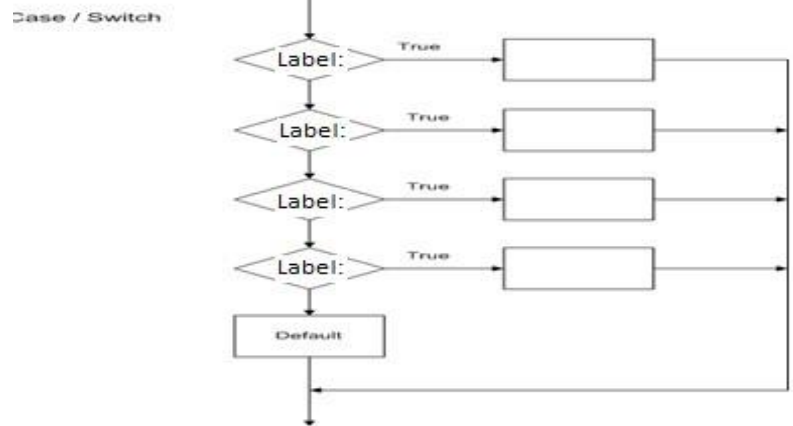
Switch statement is used to solve multiple option type. Another condition JAVA statement that is used for controlling program flow is the **switch** statement. The **switch** statement controls program flow by executing a **specific** set of statement depending on the value of an expression. A **switch** statement consists of the following components: the keyword **switch**, an expression, an opening brace, one or more **case** statements, a **default** label, and a closing brace. A **case** statement consists of a **case** label, the executable statements, and the keyword **break** use for exit the **switch** block. The syntax for the **switch** statement is as follows:

Syntax:`switch(expression)`

```

{
  case label-1:
    statement-1;      break;
  case label-2:
    statement-2;      break;
  case label-3:
    statement-3;      break;
  default:
    statement-otherwise-default;
}

```

Flow Chart of Switch Case

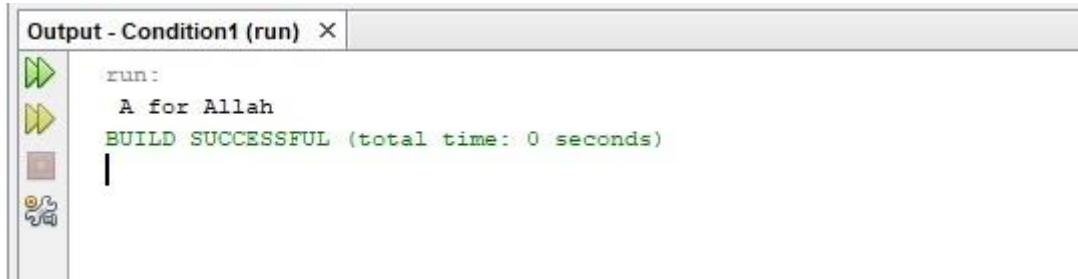
A **case** label consists of the keyword **case**, followed by a literal value or variable name it could be only character, **integer**, short, long and string value, followed by a colon. JAVA compares the value returned from the **switch** statement expression to the literal value or value of the variable named following the **case** keyword. Float, double and Boolean value are not allowed. If a match is found, the statement following the **case** label statement executes.

Example:

```

char val='A';
switch( val)
{
  case 'A':
    System.out.println (" A for Allah ");
    break;
  case 'B':
    System.out.println (" B for Bissmillah ");
    break;
  case 'C':
    System.out.println (" C for Captain ");
    break;
  case 'D':
    System.out.println (" D for Donner ");
    break;
  default:
    System.out.println(" Unknown Value");
}

```



```
Output - Condition1 (run) X
run:
  A for Allah
BUILD SUCCESSFUL (total time: 0 seconds)
```

The **expression** in **switch case** evaluates to return an **integral** value, which is then compared to the values in **different cases**, where it matches that block of code is executed, **if** there is no match, then default block is executed.

Points to Remember

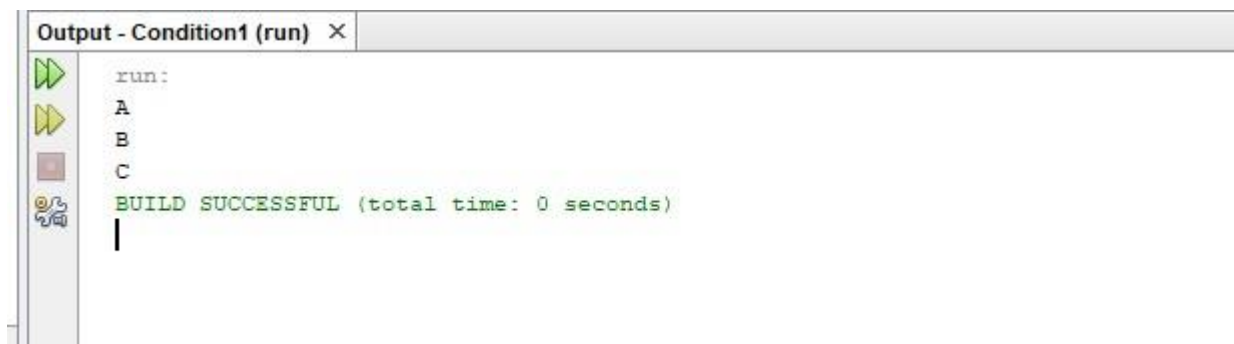
- ✓ It isn't necessary to use **break** after each block, but **if** you do not use it, all the consecutive block of codes will get executed after the matching block.
- ✓ Switch **case** is valid only for "equals to" condition, it doesn't work for any other relational operators.

Example:

```
int i = 1;
switch( i)
{
    case 1:                // compiler will check either i= 1?
        System.out.println("A");    // if i=1, it will print "A"
                                    //No break

    case 2:
        System.out.println("B");    // No break

    case 3:
        System.out.println("C");
        break;
}
```



```
Output - Condition1 (run) X
run:
  A
  B
  C
BUILD SUCCESSFUL (total time: 0 seconds)
```


Explanation:

1. The output was supposed to be only **A** because only the first **case** matches, but as there is no **break** statement after the block, the next blocks are executed, until the cursor encounters a **break**.
2. **default** case can be placed anywhere in the **switch case**. Even if we don't include the default **case switch** statement works.

5. The Conditional (? :) Operator

We have covered **conditional operator ? :** in the previous condition expression which can be used to replace **if...else** statements. It has the following general form:

Syntax:

(Exp1) ?Exp2 : Exp3;

Or

(Condition)? True : False;

Where Exp1, Exp2, and Exp3 are expressions. Notice the use and placement of the colon. The value of a ? expression is determined like:

- Exp1 is evaluated. If it is true, then Exp2 is evaluated and becomes the value of the entire? Expression.
- If Exp1 is false, then Exp3 is evaluated and its value becomes the value of the expression.

Example:

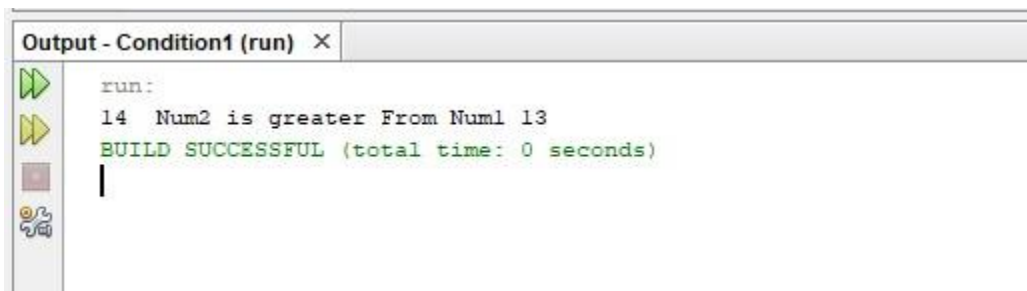
```
int Num1 =13;
```

```
int Num2 = 14;
```

```
String Result;
```

```
Result = ( Num1> Num2)? (Num1)+" Num1 is greater From Num2 "+Num2: +(Num2)+" Num2 is greater From Num1 "+ Num1;
```

```
System.out.println(Result);
```



```
Output - Condition1 (run) X
run:
14 Num2 is greater From Num1 13
BUILD SUCCESSFUL (total time: 0 seconds)
```

It will check **if (Num1> Num2)** then it will evaluate *Expression 1* i.e. , but since **Num1** is not greater than from **Num2** here so it will evaluate *expression 2* i.e.

Points to Remember

- ✓ It is also called **“Ternary Operator”** as it works on three operands.

Exercise

Theory Questions

1. Decision-making structures cannot be nested. True or false with give any example.
2. How do you perform more than one statement when a condition is true?
3. Correct the syntax error line by line.
 - 1) If x > 25
 - 2) {
 - 3) Y = x;
 - 4) else
 - 5) %y= z
 - 6) }
4. What function of **break** keyword/statement and where we can use it?
5. What function of **default** keyword/statement and where we can use it?

Practical Questions

1. Write a simple program to construct a **Calculator** that can perform **Basic Operation** Mathematical operation as well.
2. Write a simple program to check an alphabet entered by user is a **Vowel** or not using **if statement**, **if else if else statement** and **switch case statements**.

if Statement
if else if Statement
Switch Case

3. Write a code to check an integer number entered by user is **Even** or **Odd** using **Conditional operator statement**.
4. Write a program to input subject marks Math, English and Physics then to prepare marks sheet of a student that will show the Obtain marks, percentage and grade.
5. Write a program that input a number, then report whether the number is in the range from 1 to 100. Otherwise the number above to 100.
6. The following is supposed to cause an action or print message, whenever x is 5 and y is 9.

```

if ( x >= y )
    System.out.println("x is greater than y");
else
    System.out.println("y is greater than x ");

```

7. Write a **switch** statement that output messages indicating what day has been numerically input (for example 1-> Monday, 2-> Tuesday, 3-> Wednesday,...)

Output

Enter Any Number [1-7]: 3 ← Enter

Today is Wednesday

Objective MCQ's

1. You can exit a switch statement using a(n) ____
 - a) break
 - b) end
 - c) quit
 - d) complete
2. When the value return by a switch statement expression does not match a case label, the statements within the ____ label execute.
 - a) Exception
 - b) Else
 - c) Error
 - d) Default
3. In a simple if statement with no else, what happens if the condition following the if is false?
 - a) The program searches for the last else in the program.
 - b) Nothing, or control falls through to the statements following the if.
 - c) The body of the if statement is executed.
 - d) The program as a whole is executed.
4. The conditional operator statement that

```
int x=0 ;
System.out.println(( x ==0 )? "x equal to zero" : "x not equal to zero");
```

 - a) Is incorrect syntax
 - b) Is correct syntax, but x equal to zero will be print
 - c) Cause a run time error
 - d) Has no effect on the program?

5. What value is assign in *income_tax* to salary by the **if** statement when *salary* is 55000?
If (salary >70000)

```
Income_tax= 1000;
```

```
else if (salary > 50000 )
```

```
Income_tax = 500;
```

```
else
```

```
Income_tax = 0;
```

- a) 1000
 - b) 500
 - c) 0
 - d) Nothing above all
6. Which is type of ternary operator?
- a) Logical operator
 - b) Assignments operator
 - c) Relational
 - d) Conditional operator