

Conditional Constructs

Learning Outcomes:

After successful completion of this lesson, you should be able to:

- Use single, two-way and multi-way selection in an algorithm.

Conditional Statement

In computer science, conditional statements, conditional expressions and conditional constructs are features of a programming language, which perform different computations or actions depending on whether a programmer-specified Boolean condition evaluates to true or false. It helps you to make a decision based on certain conditions. These conditions are specified by a set of conditional statements having Boolean expressions which are evaluated to a Boolean value true or false.

Conditional statements in a computer program support decisions based on a certain condition. If the condition is met, or "true," a certain piece of code is executed.

For example, you want to convert user-entered text to lowercase. Execute the code only if the user entered capitalized text. If not, you don't want to execute the code because it will lead to a runtime error.

There are two main conditional statements used in Java: the if-then and if-then-else statements, and the switch statement.

Conditional Operators

In the example above, we used a single operator. These are the standard operators you can use:

- equal to: =
- less than: <
- more than: >
- greater than or equal to: >=
- less than or equal to: <=

In addition to these, there are four more operators used with conditional statements:

- and: &&
- not: !
- or: ||
- is equal to: ==

For example, the driving age is considered to be from age 16 to age 85, in which case the AND operator can be used.

else if (age > 16 && age < 85)

This will return true only if both conditions are met. The operators NOT, OR, and IS EQUAL TO can be used in a similar way.

The If-Then and If-Then-Else Statements (Single Selection)

The most basic flow control statement in Java is if-then: if [something] is true, do [something]. This statement is a good choice for simple decisions. The basic structure of an if statement starts with the word "if," followed by the statement to test, followed by curly braces that wrap the action to take if the statement is true. It looks like this:

```
if ( statement ) { // do something here.... }
```

This statement can also be extended to do something else if the condition is false:

```
if ( statement ) { // do something here...}  
else { // do something else...}
```

For example, if you are determining whether someone is old enough to drive, you might have a statement that says "if your age is 16 or older, you can drive; else, you cannot drive."

```
Int age = 17;
```

```
if age >= 16 {System.out.println("You can drive.");}
```

```
else { System.out.println("You are not old enough to drive.")
```

There is no limit to the number of else statements you can add.

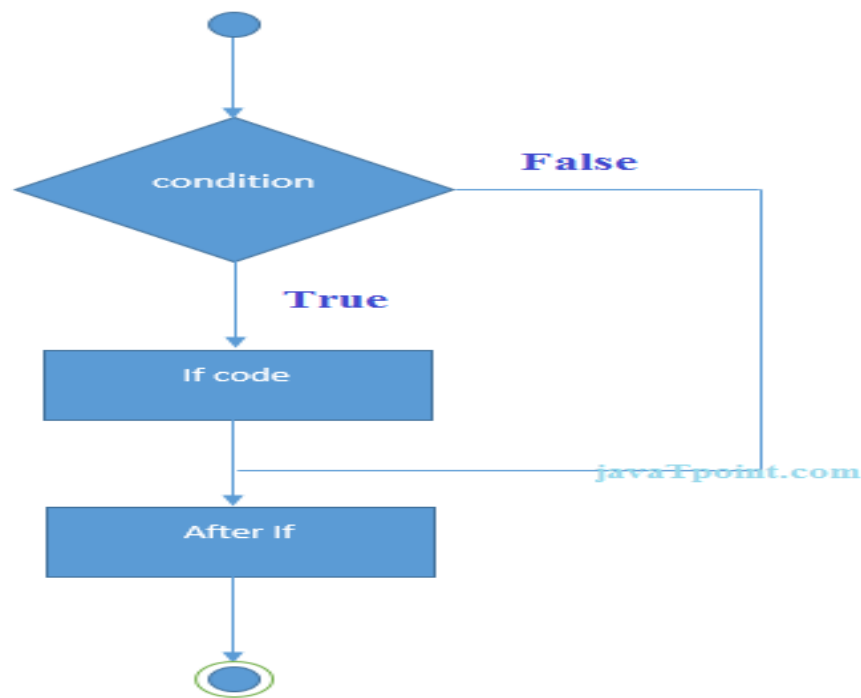
If Statement

Use the if statement to specify a block of Java code to be executed if a condition is true.

Syntax:

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

Note: that if is in lowercase letters. Uppercase letters (If or IF) will generate an error.



Examples:

```
if (20 > 18) {  
    System.out.println("20 is greater than 18");  
}
```

We can also test variables:

```
int x = 20;  
int y = 18;  
if (x > y) {  
    System.out.println("x is greater than y");  
}
```

Example explained

In the example above we use two variables, x and y, to test whether x is greater than y (using the > operator). As x is 20, and y is 18, and we know that 20 is greater than 18, we print to the screen that "x is greater than y".

If Else Statement

Use the else statement to specify a block of code to be executed if the condition is false.

Syntax:

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

Example:

```
int time = 20;  
if (time < 18) {  
    System.out.println("Good day.");  
} else {  
    System.out.println("Good evening.");  
}  
// Outputs "Good evening."
```

Example explained

In the example above, time (20) is greater than 18, so the condition is false. Because of this, we move on to the else condition and print to the screen "Good evening". If the time was less than 18, the program would print "Good day".

A year is leap, if it is divisible by 4 and 400. But, not by 100.

```
1. public class LeapYearExample {
2.     public static void main(String[] args) {
3.         int year=2020;
4.         if(((year % 4 ==0) && (year % 100 !=0)) || (year % 400==0)){
5.             System.out.println("LEAP YEAR");
6.         }
7.         else{
8.             System.out.println("COMMON YEAR");
9.         }
10.    }
11. }
```

Output:

LEAP YEAR

Example of if-else statement

```
public class IfElseExample {

    public static void main(String args[]){
        int num=120;
        if( num < 50 ){
            System.out.println("num is less than 50");
        }
        else {
            System.out.println("num is greater than or equal 50");
        }
    }
}
```

Output:

num is greater than or equal 5

Multiple Selection Construct (If, Else If, Else Statement)

Use the else if statement to specify a new condition if the first condition is false.

Syntax:

```
if (condition1) {  
    // block of code to be executed if condition1 is true  
} else if (condition2) {  
    // block of code to be executed if the condition1 is false and condition2 is true  
} else {  
    // block of code to be executed if the condition1 is false and condition2 is false  
}
```

Example:

```
int time = 22;  
if (time < 10) {  
    System.out.println("Good morning.");  
} else if (time < 20) {  
    System.out.println("Good day.");  
} else {  
    System.out.println("Good evening.");  
}  
// Outputs "Good evening."
```

Example explained

In the example above, time (22) is greater than 10, so the first condition is false. The next condition, in the else if statement, is also false, so we move on to the else condition since condition1 and condition2 is both false - and print to the screen "Good evening".

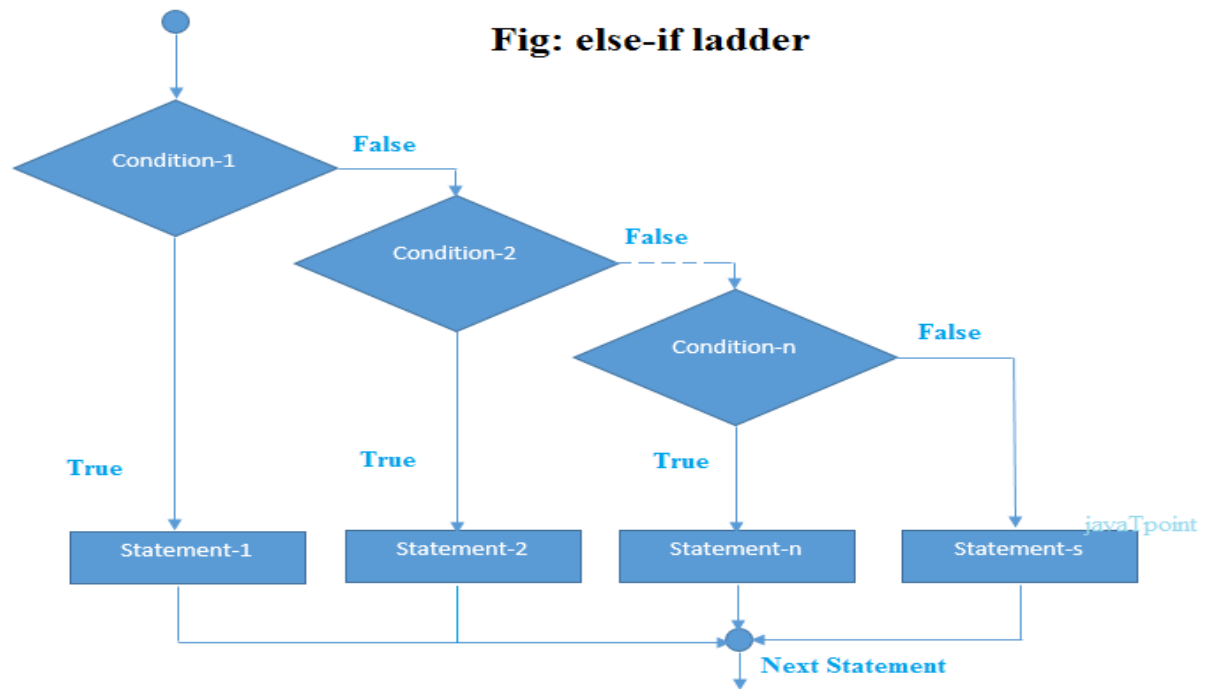
Multi-way Statement (if-else-if ladder Statement)

The if-else-if ladder statement executes one condition from multiple statements.

```
if(condition_1) {  
    /*if condition_1 is true execute this*/  
    statement(s);  
}  
else if(condition_2) {  
    /* execute this if condition_1 is not met and  
    * condition_2 is met  
    */  
    statement(s);  
}  
else if(condition_3) {  
    /* execute this if condition_1 & condition_2 are  
    * not met and condition_3 is met  
    */  
    statement(s);  
}  
.  
.  
.  
else {  
    /* if none of the condition is true  
    * then these statements gets executed  
    */  
    statement(s);  
}
```

Note:

The most important point to note here is that in if-else-if statement, as soon as the condition is met, the corresponding set of statements get executed, rest gets ignored. If none of the condition is met then the statements inside “else” gets executed.



Example of if-else-if

```

public class IfElseIfExample {

    public static void main(String args[]){
        int num=1234;
        if(num <100 && num>=1) {
            System.out.println("Its a two digit number");
        }
        else if(num <1000 && num>=100) {
            System.out.println("Its a three digit number");
        }
        else if(num <10000 && num>=1000) {
            System.out.println("Its a four digit number");
        }
        else if(num <100000 && num>=10000) {
            System.out.println("Its a five digit number");
        }
        else {
            System.out.println("number is not between 1 & 99999");
        }
    }
}
  
```


Output:

It's a four digit number

Example:

```
1. //Java Program to demonstrate the use of If else-if ladder.
2. //It is a program of grading system for fail, D grade, C grade, B grade, A grade and A+.

3. public class IfElseIfExample {
4.     public static void main(String[] args) {
5.         int marks=65;
6.
7.         if(marks<50){
8.             System.out.println("fail");
9.         }
10.        else if(marks>=50 && marks<60){
11.            System.out.println("D grade");
12.        }
13.        else if(marks>=60 && marks<70){
14.            System.out.println("C grade");
15.        }
16.        else if(marks>=70 && marks<80){
17.            System.out.println("B grade");
18.        }
19.        else if(marks>=80 && marks<90){
20.            System.out.println("A grade");
21.        }else if(marks>=90 && marks<100){
22.            System.out.println("A+ grade");
23.        }else{
24.            System.out.println("Invalid!");
25.        }
26.    }
27. }
```

Output: C grade

Activities/Assessment

- I. Application.** Read the questions carefully and confine your responses to an analysis of the questions as written.

Use ***If-Then and If-Then-Else Statements only***

1. Write a program that will accept an integer and execute one of the following based on the input statement.
 - If 0, Display only "Hello World".
 - If 2, Display only "I am Groot".
 - If 3, Display only "To The Top".
 - If 4, Display only "Where is the Horizon?"
 - If 5, Display only "I don't know".
 - If 6, Display only "Yeah, I will".
2. Write a java program that keeps a number from the user and generates an integer between 1 and 7 and display the weekdays.

Sample output:

Input number: 3

Wednesday

3. Write a Java Program that takes the user to provide a single character from the alphabet. Print Vowel or Consonant depending on the user input. If the user input is not a letter print an error message.
4. Write a program that reads a whole number and prints "zero" if the number is zero. Otherwise, print "positive" or "negative".
5. Write a program that takes of the age from the user and display "child" if the age below 13, display "teen" if the age around 13-19, and "adult" from 20 and above.

Switched 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.

Syntax:

```
switch(expression) {  
    case value :  
        // Statements  
        break; // optional  
  
    case value :  
        // Statements  
        break; // optional  
  
    // You can have any number of case statements.  
    default : // Optional  
        // Statements  
}
```

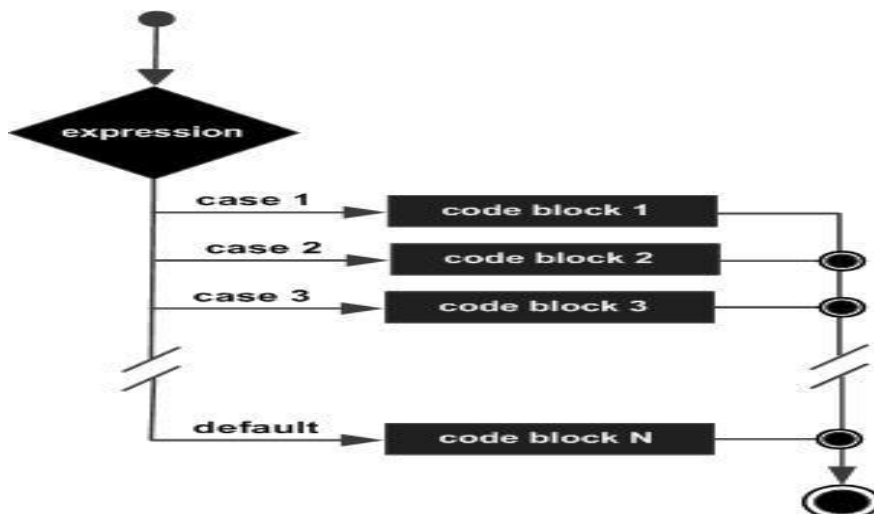
Use the switch statement to select one of many code blocks to be executed.

The break Keyword

- When Java reaches a break keyword, it breaks out of the switch block.
- This will stop the execution of more code and case testing inside the block.
- When a match is found, and the job is done, it's time for a break. There is no need for more testing.
- A break can save a lot of execution time because it "ignores" the execution of all the rest of the code in the switch block.

The default Keyword

The default keyword specifies some code to run if there is no case match:



Note that if the default statement is used as the last statement in a switch block, it does not need a break.

Points to Remember

- There can be *one or N number of case values* for a switch expression.
- The case value must be of switch expression type only. The case value must be *literal or constant*. It doesn't allow variables.
- The case values must be *unique*. In case of duplicate value, it renders compile-time error.
- The Java switch expression must be of *byte, short, int, long (with its Wrapper type), enums and string*.
- Each case statement can have a *break statement* which is optional. When control reaches to the break statement, it jumps the control after the switch expression. If a break statement is not found, it executes the next case.
- The case value can have a *default label* which is optional.

Example:

```
public class Test {  
  
    public static void main(String args[]) {  
        // char grade = args[0].charAt(0);  
        char grade = 'C';  
  
        switch(grade) {  
            case 'A' :  
                System.out.println("Excellent!");  
                break;  
            case 'B' :  
            case 'C' :  
                System.out.println("Well done");  
                break;  
            case 'D' :  
                System.out.println("You passed");  
            case 'F' :  
                System.out.println("Better try again");  
                break;  
            default :  
                System.out.println("Invalid grade");  
        }  
        System.out.println("Your grade is " + grade);  
    }  
}
```

Compile and run the above program using various command line arguments. This will produce the following result:

Output:

Well done
Your grade is C

Example:

```
int day = 4;
switch (day) {
    case 6:
        System.out.println("Today is Saturday");
        break;
    case 7:
        System.out.println("Today is Sunday");
        break;
    default:
        System.out.println("Looking forward to the Weekend");
}
// Outputs "Looking forward to the Weekend"
```

Example:

```
1. public class SwitchExample {
2.     public static void main(String[] args) {
3.         //Declaring a variable for switch expression
4.         int number=20;
5.         //Switch expression
6.         switch(number){
7.             //Case statements
8.             case 10: System.out.println("10");
9.             break;
10.            case 20: System.out.println("20");
11.            break;
12.            case 30: System.out.println("30");
13.            break;
14.            //Default case statement
15.            default: System.out.println("Not in 10, 20 or 30");
16.        } } }
```

Output:

20

Finding Month Example:

```
1. //Java Program to demonstrate the example of Switch statement
2. //where we are printing month name for the given number
3. public class SwitchMonthExample {
4.     public static void main(String[] args) {
5.         //Specifying month number
6.         int month=7;
7.         String monthString="";
8.         //Switch statement
9.         switch(month){
10.            //case statements within the switch block
11.            case 1: monthString="1 - January";
12.            break;
13.            case 2: monthString="2 - February";
14.            break;
15.            case 3: monthString="3 - March";
16.            break;
17.            case 4: monthString="4 - April";
18.            break;
19.            case 5: monthString="5 - May";
20.            break;
21.            case 6: monthString="6 - June";
22.            break;
23.
24.            case 7: monthString="7 - July";
25.            break;
26.            case 8: monthString="8 - August";
27.            break;
28.
```

```
29. case 9: monthString="9 - September";
30. break;
31. case 10: monthString="10 - October";
32.
33. break;
34. case 11: monthString="11 - November";
35. break;
36. case 12: monthString="12 - December";
37. break;
38. default: System.out.println("Invalid Month!");
39. }
40. //Printing month of the given number
41. System.out.println(monthString);
42. }
43. }
```

Finding Month Example:

Output: 7 – July

Program to check Vowel or Consonant:

```
1. public class SwitchVowelExample {
2.     public static void main(String[] args) {
3.         char ch='O';
4.         switch(ch)
5.         {
6.             case 'a':
7.                 System.out.println("Vowel");
8.                 break;
9.             case 'e':
10.                System.out.println("Vowel");
11.                break;
12.             case 'i':
13.                System.out.println("Vowel");
14.                break;
15.             case 'o':
16.                System.out.println("Vowel");
17.                break;
18.             case 'u':
19.                System.out.println("Vowel");
20.                break;
21.             case 'A':
22.                System.out.println("Vowel");
23.                break;
```

```
24.     case 'E':
25.         System.out.println("Vowel");
26.         break;
27.     case 'I':
28.         System.out.println("Vowel");
29.         break;
30.     case 'O':
31.         System.out.println("Vowel");
32.         break;
33.     case 'U':
34.         System.out.println("Vowel");
35.         break;
36.     default:
37.         System.out.println("Consonant");
38.     }
39. }
40. }
```

Output:

Vowel

Activities/Assessment

- I. **Application.** Read the questions carefully and confine your responses to an analysis of the questions as written.

Use **Switched Statement only**

1. Write a program that will accept an integer and execute one of the following based on the input statement.
 - If 0, Display only "Hello World".
 - If 2, Display only "I am Groot".
 - If 3, Display only "To The Top".
 - If 4, Display only "Where is the Horizon?"
 - If 5, Display only "I don't know".
 - If 6, Display only "Yeah, I will".
2. Write a java program that keeps a number from the user and generates an integer between 1 and 7 and display the weekdays.

Sample output:

Input number: 3

Wednesday

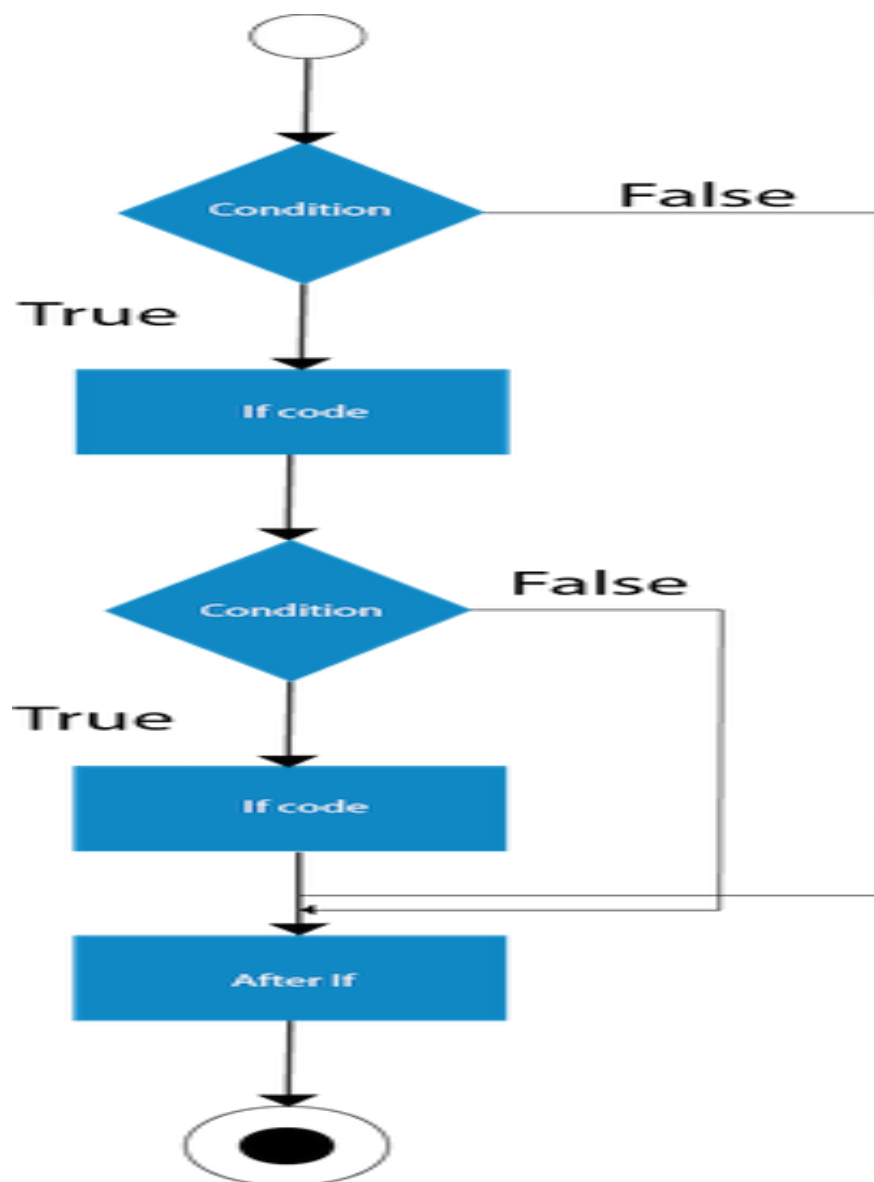
3. Write a Java Program that takes the user to provide a single character from the alphabet. Print Vowel or Consonant depending on the user input. If the user input is not a letter print an error message.
4. Write a program that reads a whole number and prints "zero" if the number is zero. Otherwise, print "positive" or "negative".
5. Write a program that takes of the age from the user and display "child" if the age below 13, display "teen" if the age around 13-19, and "adult" from 20 and above.

Nested If- Statement

The nested if statement represents the if block within another if block. Here, the inner if block condition executes only when outer if block condition is true.

Syntax:

```
1. if(condition){  
2. //code to be executed  
3. if(condition){  
4. //code to be executed  
5. }  
6. }
```



Example 1:

```
1. //Java Program to demonstrate the use of Nested If Statement.
2. public class JavaNestedIfExample {
3.     public static void main(String[] args) {
4.         //Creating two variables for age and weight
5.         int age=20;
6.         int weight=80;
7.         //applying condition on age and weight
8.         if(age>=18){
9.             if(weight>50){
10.                System.out.println("You are eligible to donate blood");
11.            }
12.        }
13.    }}
```

Output:

You are eligible to donate blood

Example 2:

```
1. //Java Program to demonstrate the use of Nested If Statement.
2. public class JavaNestedIfExample2 {
3.     public static void main(String[] args) {
4.         //Creating two variables for age and weight
5.         int age=25;
6.         int weight=48;
7.         //applying condition on age and weight
8.         if(age>=18){
9.             if(weight>50){
10.                System.out.println("You are eligible to donate blood");
11.            } else{
12.                System.out.println("You are not eligible to donate blood");
13.            }
14.        } else{
15.            System.out.println("Age must be greater than 18");
16.        }
17.    } }
```

Output:

You are not eligible to donate blood

```
// Example for Nested If in Java Programming

package ConditionalStatements;

import java.util.Scanner;

public class NestedIf {
    private static Scanner sc;

    public static void main(String[] args) {
        int age;
        sc = new Scanner(System.in);
        System.out.println(" Please Enter you Age: ");
        age = sc.nextInt();

        if (age < 18) {
            System.out.println("You are Minor.");
            System.out.println("You are Not Eligible to Work");
        }
        else {
            if (age >= 18 && age <= 60 ) {
                System.out.println("You are Eligible to Work");
                System.out.println("Please fill in your details and
apply");
            }
            else {
                System.out.println("You are too old to work as per
the Government rules");
                System.out.println("Please Collect your pension!");
            }
        }
        System.out.println("\nThis Message is coming from Outside the IF
ELSE STATEMENT");
    }
}
```

Analysis

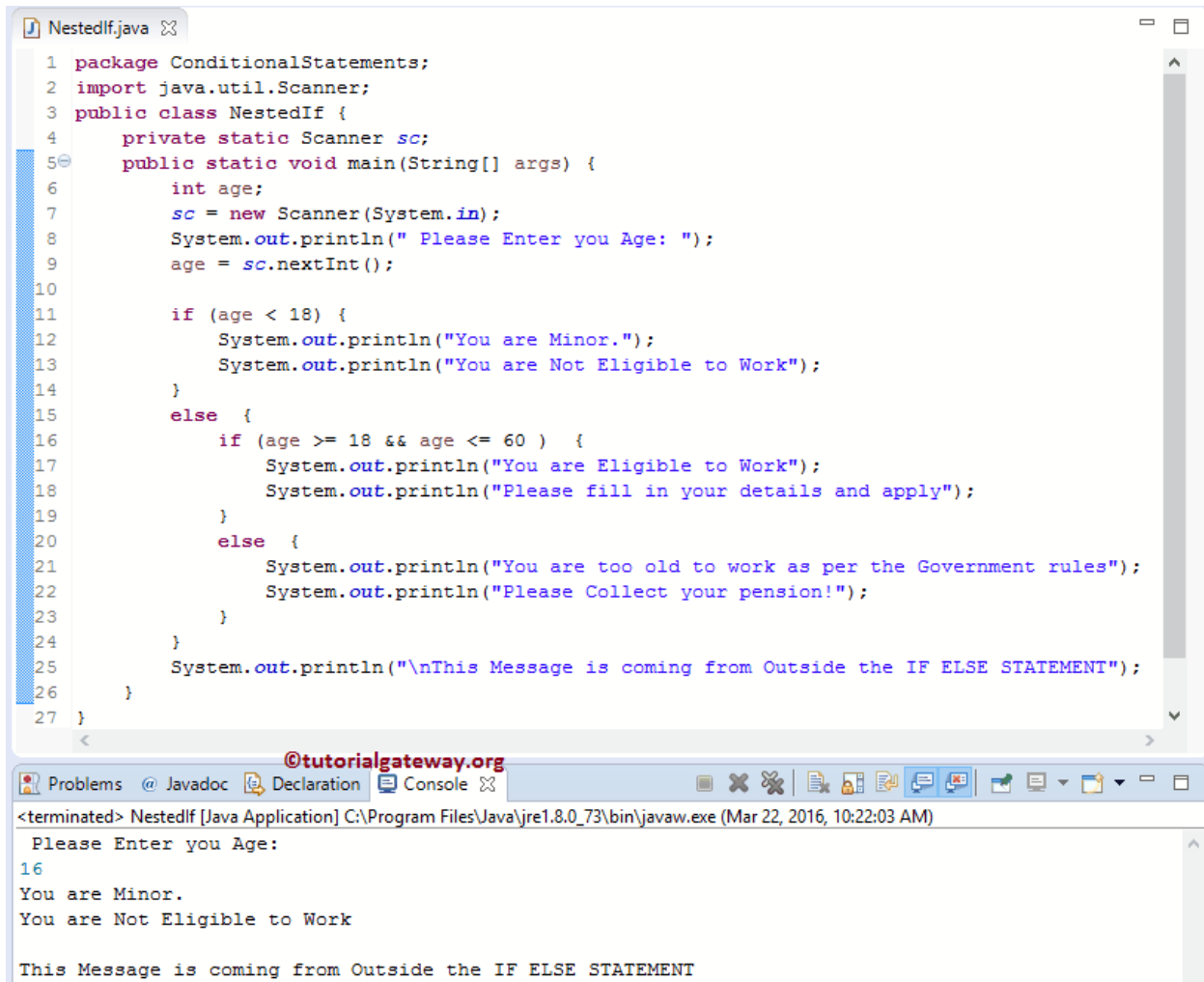
Within this Java nested if else program, If the person's age is less than 18, then he is not eligible to work. If the person's age is greater than or equal to 18, then the first condition fails. It will check the else statement.

Within the Else statement, there is another if condition (called as Nested If).

- Java Nested IF Statement will check whether the person's age is greater than or equal to 18 and less than or equal to 60. If the condition is TRUE, then he can apply for the job.

- If the Nested If condition is FALSE, then he is too old to work as per the government.
- We also place one System.out.println statement outside the If Else block, and it will execute irrespective of condition result.

Output 1: From the screenshot below, you can observe that We entered the age of 16. Here, the first If condition is TRUE. So, the statements inside the first if block will execute.



```

1 package ConditionalStatements;
2 import java.util.Scanner;
3 public class NestedIf {
4     private static Scanner sc;
5     public static void main(String[] args) {
6         int age;
7         sc = new Scanner(System.in);
8         System.out.println(" Please Enter you Age: ");
9         age = sc.nextInt();
10
11         if (age < 18) {
12             System.out.println("You are Minor.");
13             System.out.println("You are Not Eligible to Work");
14         }
15         else {
16             if (age >= 18 && age <= 60 ) {
17                 System.out.println("You are Eligible to Work");
18                 System.out.println("Please fill in your details and apply");
19             }
20             else {
21                 System.out.println("You are too old to work as per the Government rules");
22                 System.out.println("Please Collect your pension!");
23             }
24         }
25         System.out.println("\nThis Message is coming from Outside the IF ELSE STATEMENT");
26     }
27 }

```

Problems @ Javadoc Declaration Console

<terminated> NestedIf [Java Application] C:\Program Files\Java\jre1.8.0_73\bin\javaw.exe (Mar 22, 2016, 10:22:03 AM)

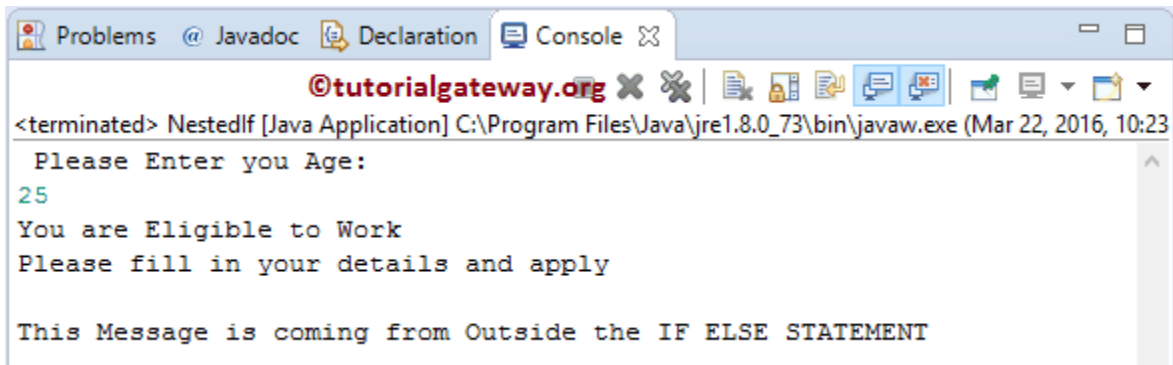
```

Please Enter you Age:
16
You are Minor.
You are Not Eligible to Work

This Message is coming from Outside the IF ELSE STATEMENT

```

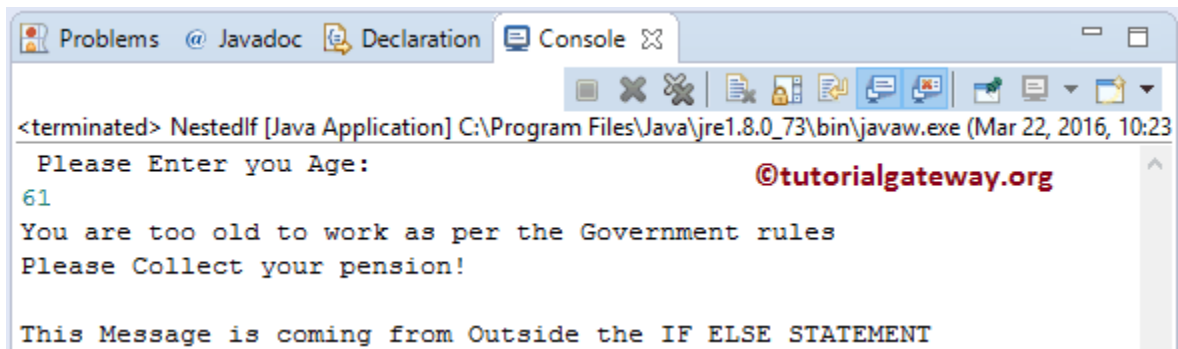
Output 2: We are going to enter age as 25 means the first IF condition is FALSE. It will go to else block. Within the else block, Javac will check the if (age >= 18 && age <=60), which is TRUE. So it will print the statements inside this block.



```
<terminated> NestedIf [Java Application] C:\Program Files\Java\jre1.8.0_73\bin\javaw.exe (Mar 22, 2016, 10:23)
Please Enter you Age:
25
You are Eligible to Work
Please fill in your details and apply

This Message is coming from Outside the IF ELSE STATEMENT
```

Output 3: This time, we are going to enter age as 61 to test the Nested If. It means the first IF condition is FALSE. It will go to else block. Within the else block, the Javac compiler will check the **if (age >= 18 && age <=60)**, which is FALSE. That is why this program will print the statements inside Nested else block.



```
<terminated> NestedIf [Java Application] C:\Program Files\Java\jre1.8.0_73\bin\javaw.exe (Mar 22, 2016, 10:23)
Please Enter you Age:
61
You are too old to work as per the Government rules
Please Collect your pension!

This Message is coming from Outside the IF ELSE STATEMENT
```