

Working with Conditional Constructs

In your daily life, you take various decisions that are based on certain conditions.

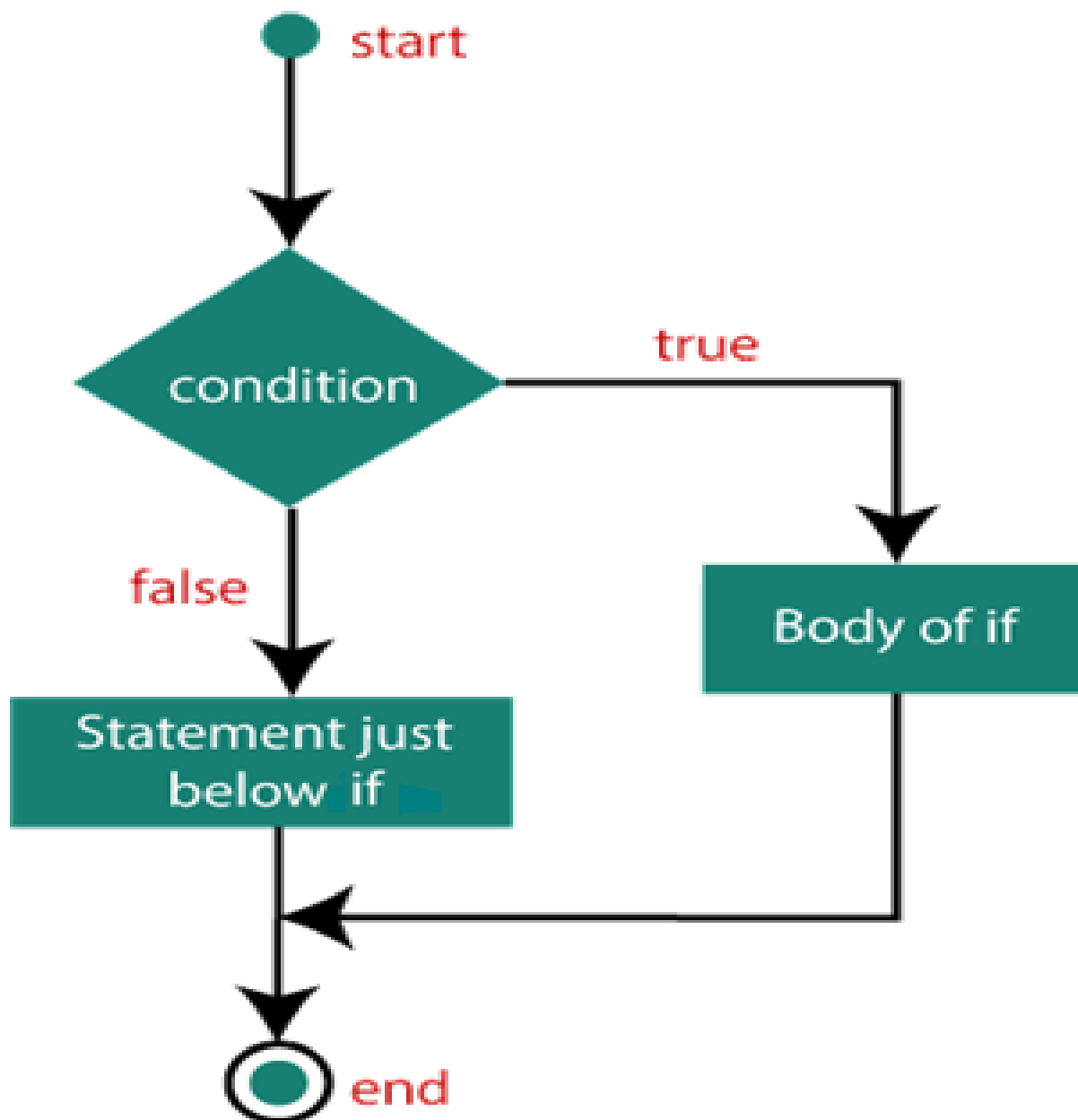
For example,

if it is raining, you will take an umbrella. In the same way, you can incorporate the decision making techniques in the Java programming language. The decision making technique can be implemented in the Java programs by using the following conditional constructs:

- The if construct
- The if...else construct
- Nesting of if ... else Statements
- The else if Ladder
- The switch construct
- The ?: Operator.

The if statement

It is one of the simplest decision-making statement which is used to decide whether a block of Java code will execute if a certain condition is true.



Example

//Java Program to demonstate the use of if statem ent.

```
public class IfExample
```

```
{
```

```
    public static void main(String[] args)
```

```
{
```

```
    //defining an 'age' variable
```

```
    int age=20;
```

```
    //checking the age
```

```
    if(age>18)
```

```
    {
```

```
        System.out.println("Age is greater than 18");
```

```
    }
```

```
        System.out.println("Part of Main()");
```

```
    }
```

```
}
```

Java if-else Statement

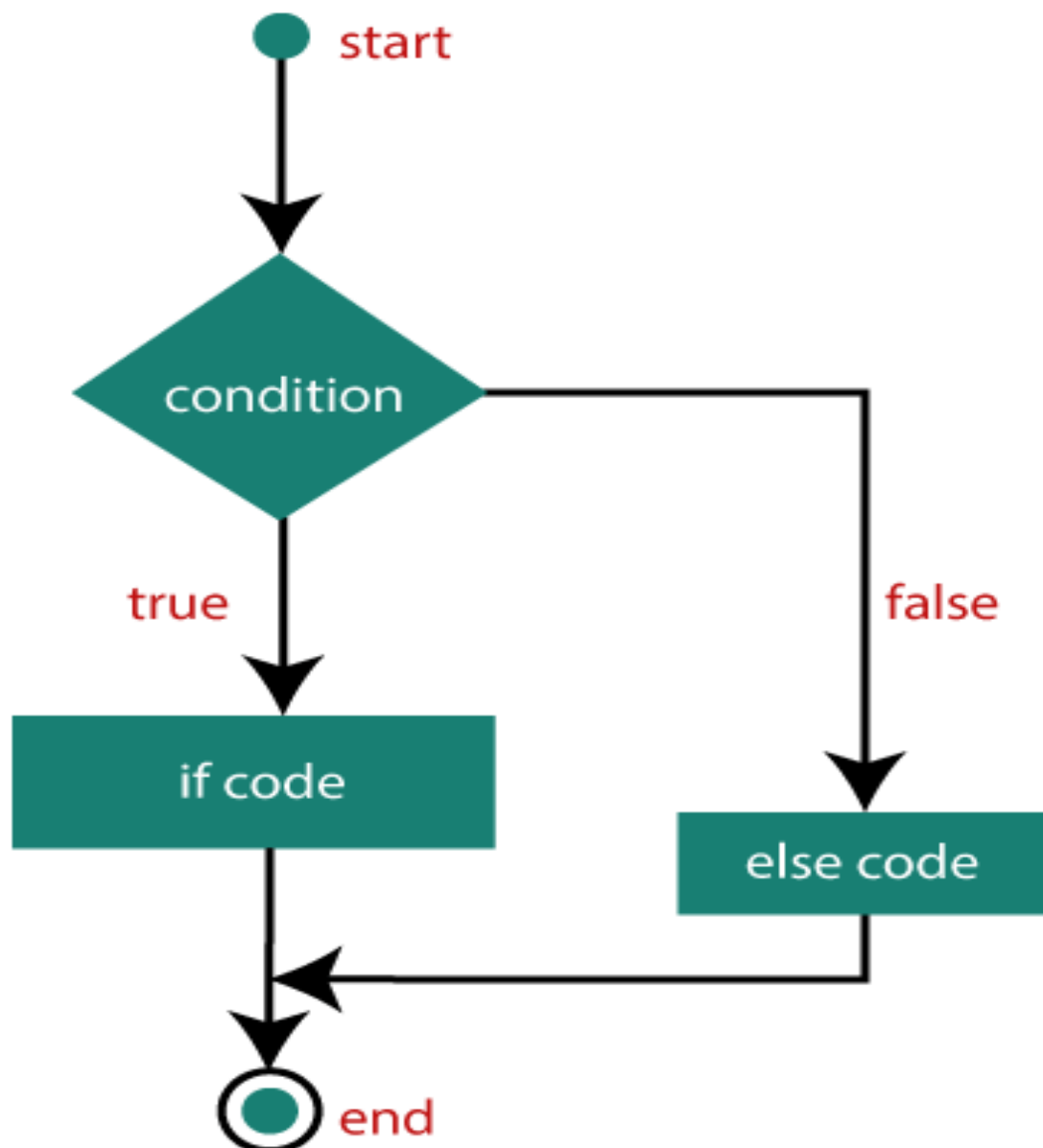
An **if...else** statement includes two blocks that are **if block** and **else block**. It is the next form of the control statement, which allows the execution of Java code in a more controlled way. It is used when you require to check two different conditions and execute a different set of codes. The **else statement** is used for specifying the execution of a block of code if the condition is false.

Syntax

```
if (condition)
{
    // block of code will execute if the condition is true
}
else
{
    // block of code will execute if the condition is false
}
```



If the condition is true, then the statements inside **if** block will be executed, but if the condition is false, then the statements of the **else** block will be executed.



//A Java Program to demonstrate the use of else statement.

//It is a program of odd and even number.

```
public class IfElseExample
```

```
{
```

```
    public static void main(String[] args)
```

```
{
```

```
    //defining a variable
```

```
    int number=13;
```

```
    //Check if the number is divisible by 2 or not
```

```
        if(number%2==0)
```

```
        {
```

```
            System.out.println("even number");
```

```
        }
```

```
        else
```

```
        {
```

```
            System.out.println("odd number");
```

```
        }
```

```
    }
```

```
}
```

Leap Year Example

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

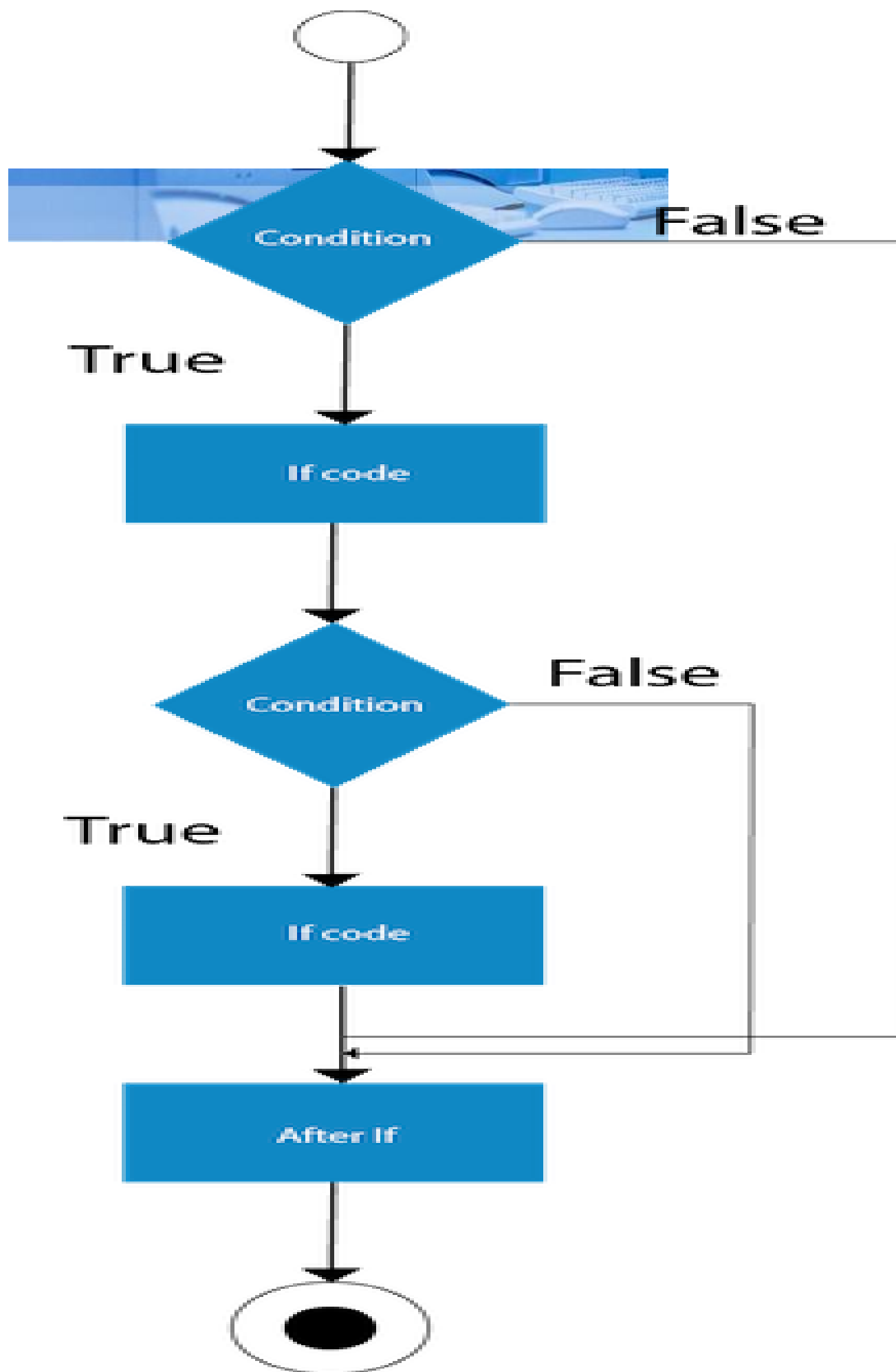
```
public class LeapYearExample
{
    public static void main(String[] args)
    {
        int year=2020;
        if(((year % 4 ==0) && (year % 100 !=0)) || (year % 400==0))
        {
            System.out.println("LEAP YEAR");
        }
        else
        {
            System.out.println("COMMON YEAR");
        }
    }
}
```


Java 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:

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

//Java Program to demonstrate the use of Nested If Statement

```
public class JavaNestedIfExample
{
    
    public static void main(String[] args)
    {
        //Creating two variables for age and weight
        int age=20;
        int weight=80;
        //applying condition on age and weight
        if(age>=18)
        {
            if(weight>50)
            {
                System.out.println("You are eligible to donate blood");
            }
        }
    }
}
```

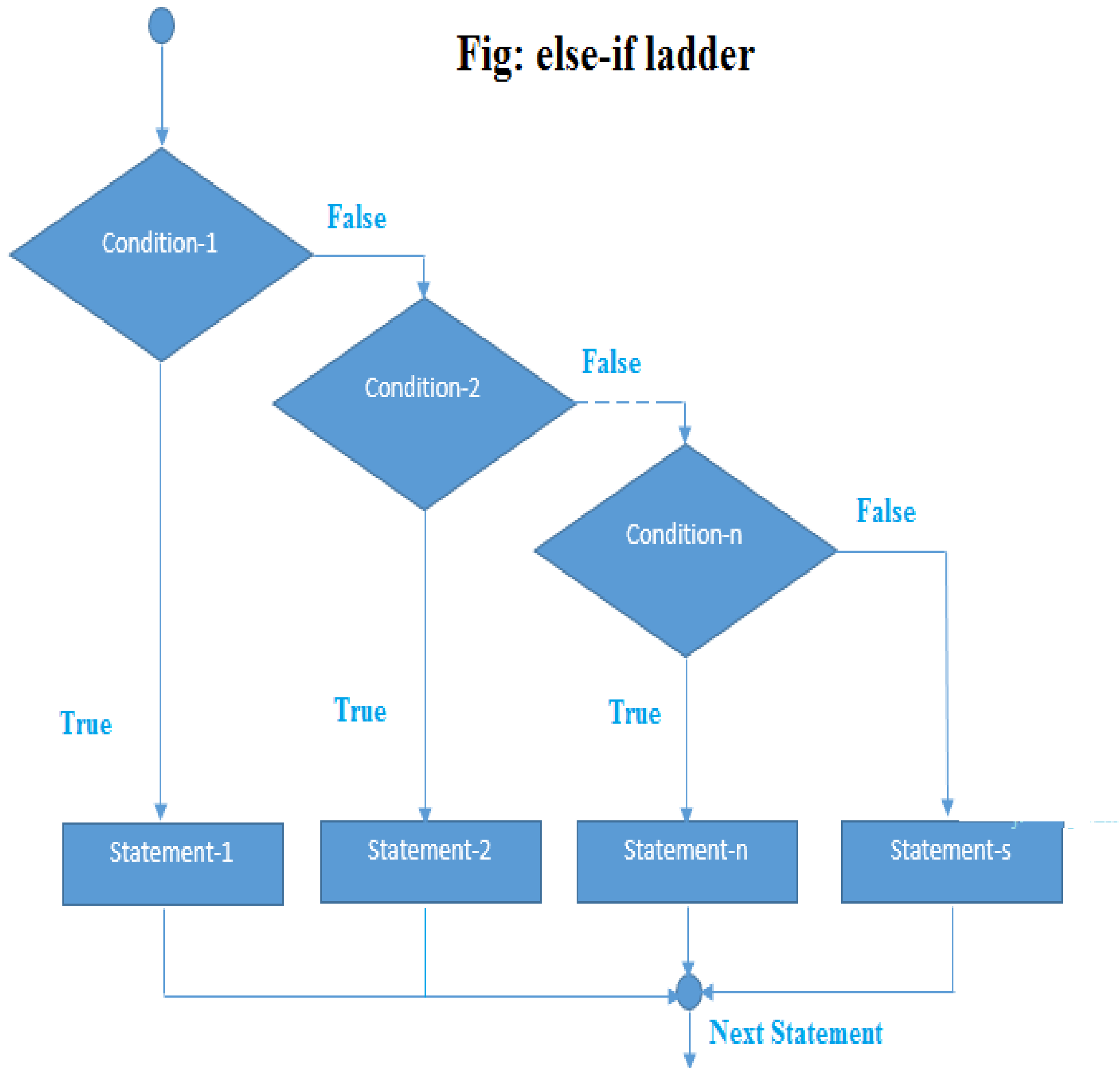
Java if-else-if ladder Statement

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

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 the conditions are false
}
```

Fig: else-if ladder



//Java Program to demonstrate the use of If else-if ladder.

//It is a program of grading system for fail, D grade, C grade, B grade, A grade and A+.

```
public class IfElseIfExample
{
    public static void main(String[] args)
    {
        int marks=65;

        if(marks<50){
            System.out.println("fail");
        }
        else if(marks>=50 && marks<60){
            System.out.println("D grade");
        }
        else if(marks>=60 && marks<70){
            System.out.println("C grade");
        }
        else if(marks>=70 && marks<80){
            System.out.println("B grade");
        }
        else if(marks>=80 && marks<90){
            System.out.println("A grade");
        }else if(marks>=90 && marks<100){
            System.out.println("A+ grade");
        }
        else{
            System.out.println("Invalid!");
        }
    }
}
```

The switch construct

The Java *switch* statement executes one statement from multiple conditions. It is like if-else-if ladder statement. The switch statement works with byte, short, int, long, enum types, String and some wrapper types like Byte, Short, Int, and Long. Since Java 7, we can use strings in the switch statement.

In other words, the switch statement tests the equality of a variable against multiple values.

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.

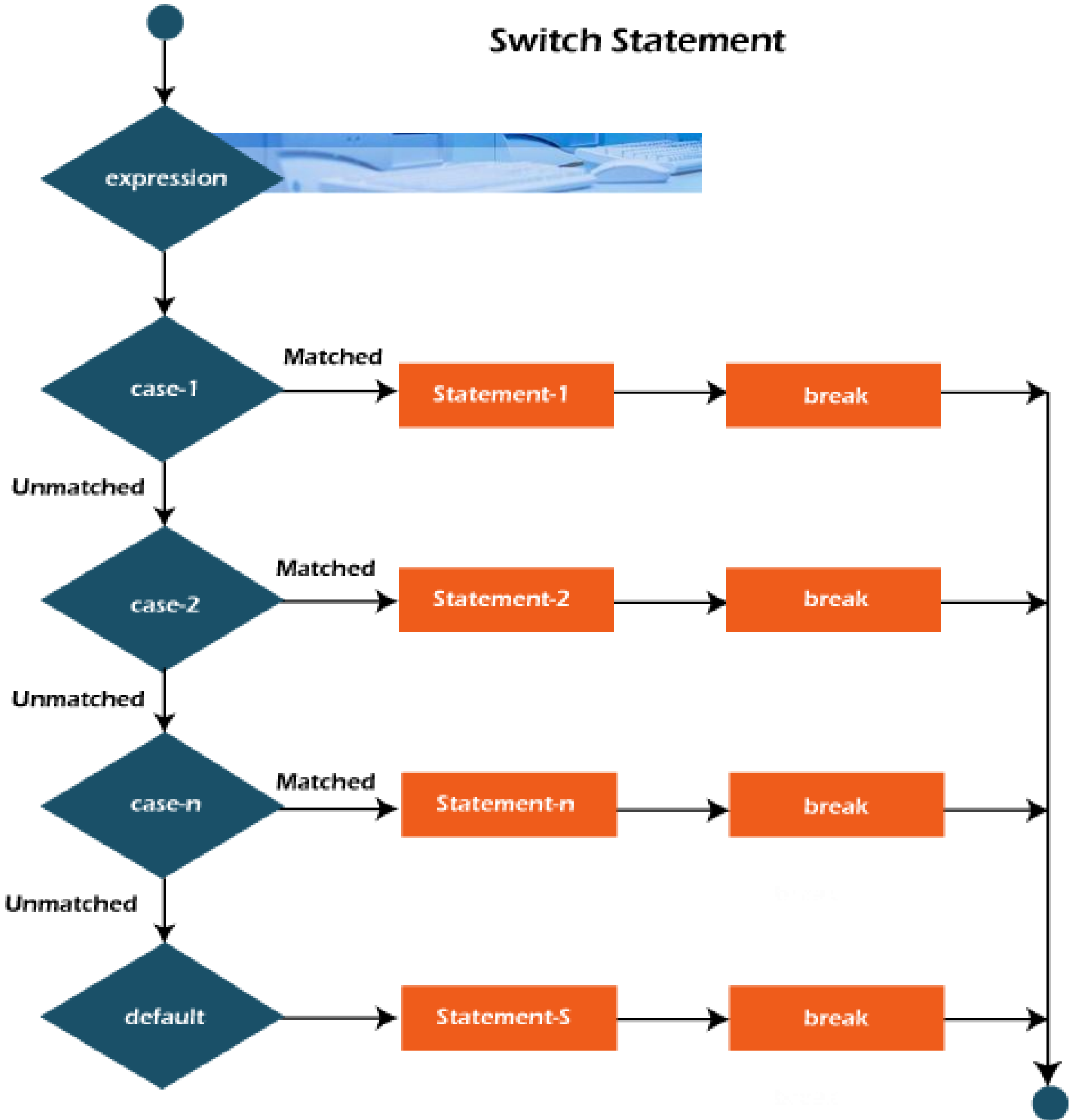
Syntax

```
switch(expression)
{
    case value1:
        //code to be executed;
        break; //optional

    case value2:
        //code to be executed;
        break; //optional
    .....

    default:
    code to be executed if all cases are not matched;
}
```

Switch Statement



//Java Program to demonstrate the example of Switch statement

//where we are printing month name for the given number

public class SwitchMonthExample

{

public static void main(String[] args)

{

//Specifying month number

int month=7;

String monthString="";

//Switch statement

switch(month)

{

//case statements within the switch block

case 1: monthString= "1 - January"; **break;**

case 2: monthString= "2 - February"; **break;**

case 3: monthString= "3 - March"; **break;**

case 4: monthString= "4 - April"; **break;**

case 5: monthString= "5 - May"; **break;**

case 6: monthString= "6 - June"; **break;**

case 7: monthString= "7 - July"; **break;**

case 8: monthString= "8 - August"; **break;**

case 9: monthString= "9 - September"; **break;**

case 10: monthString= "10 - October"; **break;**

case 11: monthString= "11 - November"; **break;**

case 12: monthString= "12 - December"; **break;**

default: System.out.println("Invalid Month!");

}

//Printing month of the given number

System.out.println(monthString);

}

}

Using Ternary Operator

We can also use ternary operator (`? :`) to perform the task of `if...else` statement. It is a shorthand way to check the condition. If the condition is true, the result of `?` is returned. But, if the condition is false, the result of `:` is returned.

Example:

```
public class IfElseTernaryExample
{
public static void main(String[] args)
{
    int number=13;
    //Using ternary operator
    String output=(number%2==0)?"even number":
    "odd number";
    System.out.println(output);
}
}
```

```
public class MathLibraryExample {  
    public static void main(String[] args) {  
        int i = 7;  
        int j = -9;  
        double x = 72.3;  
        double y = 0.34;
```

```
        System.out.println("i is " + i);  
        System.out.println("j is " + j);  
        System.out.println("x is " + x);  
        System.out.println("y is " + y);
```

/* The absolute value of a number is equal to the number if the number is positive or zero and equal to the negative of the number if the number is negative.*/

```
        System.out.println("|" + i + "|" is " + Math.abs(i));  
        System.out.println("|" + j + "|" is " + Math.abs(j));  
        System.out.println("|" + x + "|" is " + Math.abs(x));  
        System.out.println("|" + y + "|" is " + Math.abs(y));
```

// Truncating and Rounding functions You can round off a floating point
//number to the nearest integer with round()

```
        System.out.println(x + " is approximately " + Math.round(x));  
        System.out.println(y + " is approximately " + Math.round(y));
```

// The "ceiling" of a number is the smallest integer greater
//than or equal to the number. Every integer is its own
//ceiling.

```
        System.out.println("The ceiling of " + i + " is " + Math.ceil(i));  
        System.out.println("The ceiling of " + j + " is " + Math.ceil(j));  
        System.out.println("The ceiling of " + x + " is " + Math.ceil(x));  
        System.out.println("The ceiling of " + y + " is " + Math.ceil(y));
```

// The "floor" of a number is the largest integer less than or equal
//to the number. Every integer is its own floor.

```
System.out.println("The floor of " + i + " is " + Math.floor(i));  
System.out.println("The floor of " + j + " is " + Math.floor(j));  
System.out.println("The floor of " + x + " is " + Math.floor(x));  
System.out.println("The floor of " + y + " is " + Math.floor(y));
```

// Comparison operators

// min() returns the smaller of the two arguments you pass it

```
System.out.println("min(" + i + ", " + j + ") is " + Math.min(i,j));  
System.out.println("min(" + x + ", " + y + ") is " + Math.min(x,y));  
System.out.println("min(" + i + ", " + x + ") is " + Math.min(i,x));  
System.out.println("min(" + y + ", " + j + ") is " + Math.min(y,j));
```

// There's a corresponding max() method

// that returns the larger of two numbers

```
System.out.println("max(" + i + ", " + j + ") is " + Math.max(i,j));  
System.out.println("max(" + x + ", " + y + ") is " + Math.max(x,y));  
System.out.println("max(" + i + ", " + x + ") is " + Math.max(i,x));  
System.out.println("max(" + y + ", " + j + ") is " + Math.max(y,j));
```

// The Math library defines a couple of useful constants:

```
System.out.println("Pi is " + Math.PI);  
System.out.println("e is " + Math.E);
```

// Trigonometric methods. All arguments are given in radians

// Convert a 45 degree angle to radians

```
double angle = 45.0 * 2.0 * Math.PI/360.0;  
System.out.println("cos(" + angle + ") is " + Math.cos(angle));  
System.out.println("sin(" + angle + ") is " + Math.sin(angle));
```

// Inverse Trigonometric methods. All values are returned as radians

```
double value = 0.707;
```

```
System.out.println("acos(" + value + ") is " + Math.acos(value));
```

```
System.out.println("asin(" + value + ") is " + Math.asin(value));
```

```
System.out.println("atan(" + value + ") is " + Math.atan(value));
```

// Exponential and Logarithmic Methods

// exp(a) returns e (2.71828...) raised

// to the power of a.

```
System.out.println("exp(1.0) is " + Math.exp(1.0));
```

```
System.out.println("exp(10.0) is " + Math.exp(10.0));
```

```
System.out.println("exp(0.0) is " + Math.exp(0.0));
```

// log(a) returns the natural

// logarithm (base e) of a.

```
System.out.println("log(1.0) is " + Math.log(1.0));
```

```
System.out.println("log(10.0) is " + Math.log(10.0));
```

```
System.out.println("log(Math.E) is " + Math.log(Math.E));
```

// pow(x, y) returns the x raised

// to the yth power.

```
System.out.println("pow(2.0, 2.0) is " + Math.pow(2.0,2.0));
```

```
System.out.println("pow(10.0, 3.5) is " + Math.pow(10.0,3.5));
```

```
System.out.println("pow(8, -1) is " + Math.pow(8,-1));
```




```
// sqrt(x) returns the square root of x.  
for (i=0; i < 10; i++) {  
    System.out.println(  
        "The square root of " + i + " is " +  
Math.sqrt(i));  
}  
  
// Finally there's one Random method  
// that returns a pseudo-random number  
// between 0.0 and 1.0;  
  
    System.out.println("Here's one random  
number: " + Math.random());  
    System.out.println("Here's another random  
number: " + Math.random());  
  
}  
}
```