Introduction to Decision Control Structure

Unit 3

Table of contents

01

- Introduction to algorithm and Flowchart

02

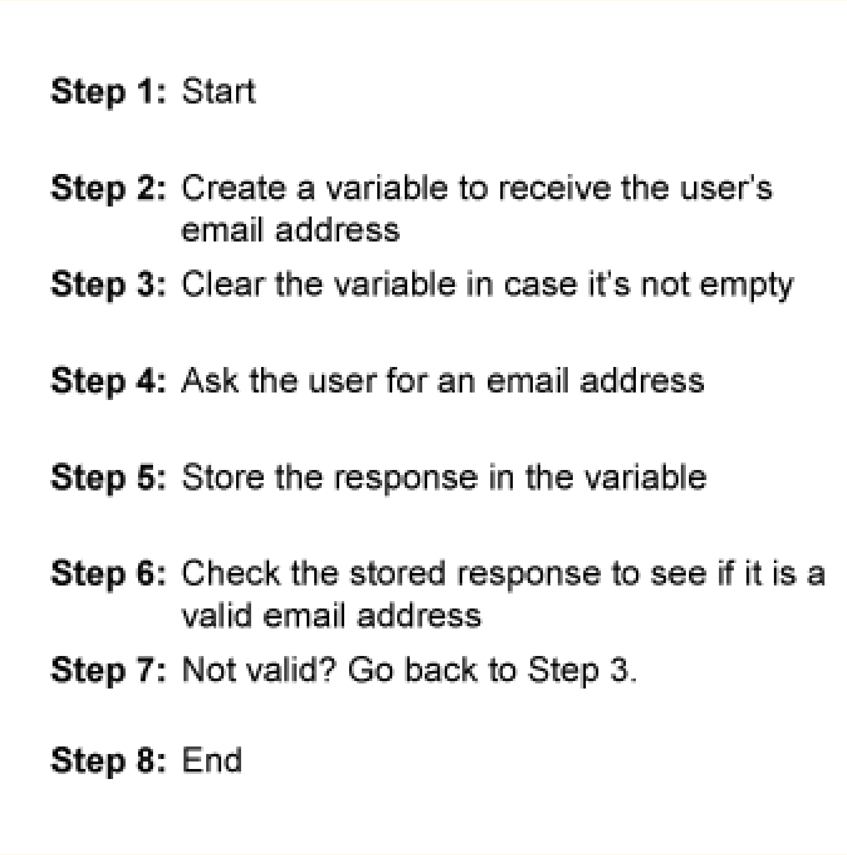
- Introduction to Decision Control Structure
  - ✓ If
  - ✓ If-else
  - ✓ If-else-if
  - ✓ Nested-if
  - ✓ Switch case

01

Introduction to algorithm and Flowchart

Algorithm

An algorithm is defined as sequence of steps to solve a problem (task)



Step 1: Start

Step 2: Create a variable to receive the user's email address

Step 3: Clear the variable in case it's not empty

Step 4: Ask the user for an email address

Step 5: Store the response in the variable

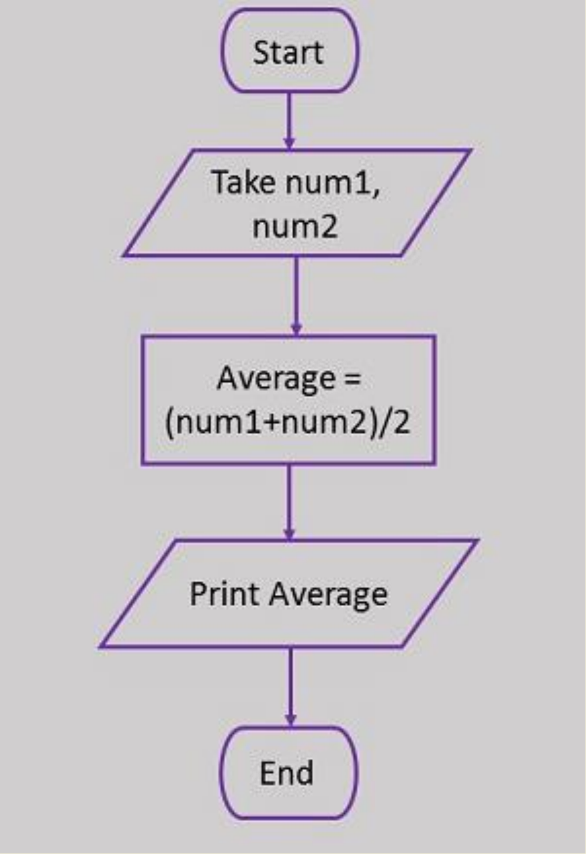
Step 6: Check the stored response to see if it is a valid email address

Step 7: Not valid? Go back to Step 3.

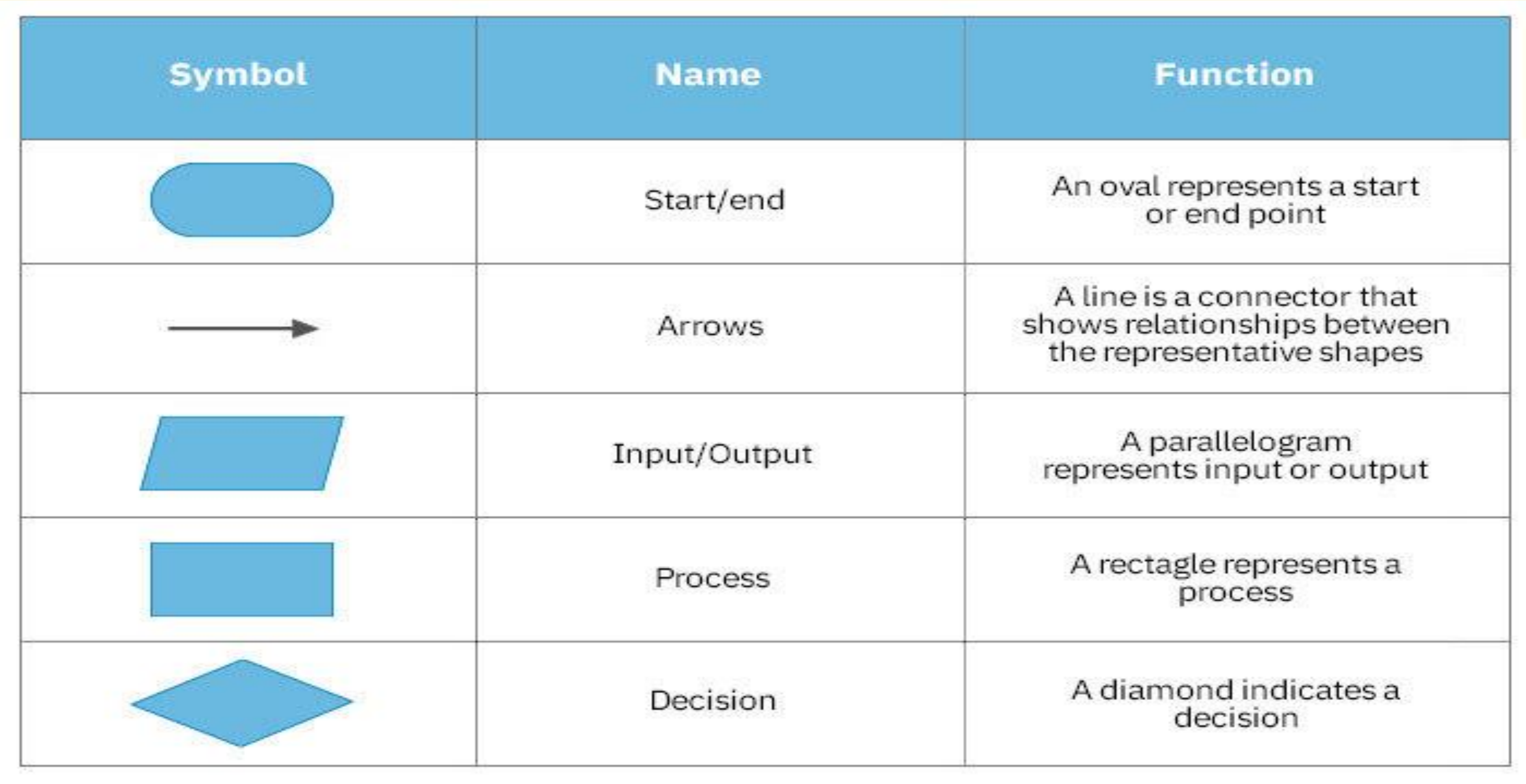
Step 8: End

Flowchart

A flow chart is a type of diagram that represents an algorithm, workflow or process. It shows the steps in the form of boxes of various kinds and their order by connecting them with arrows.



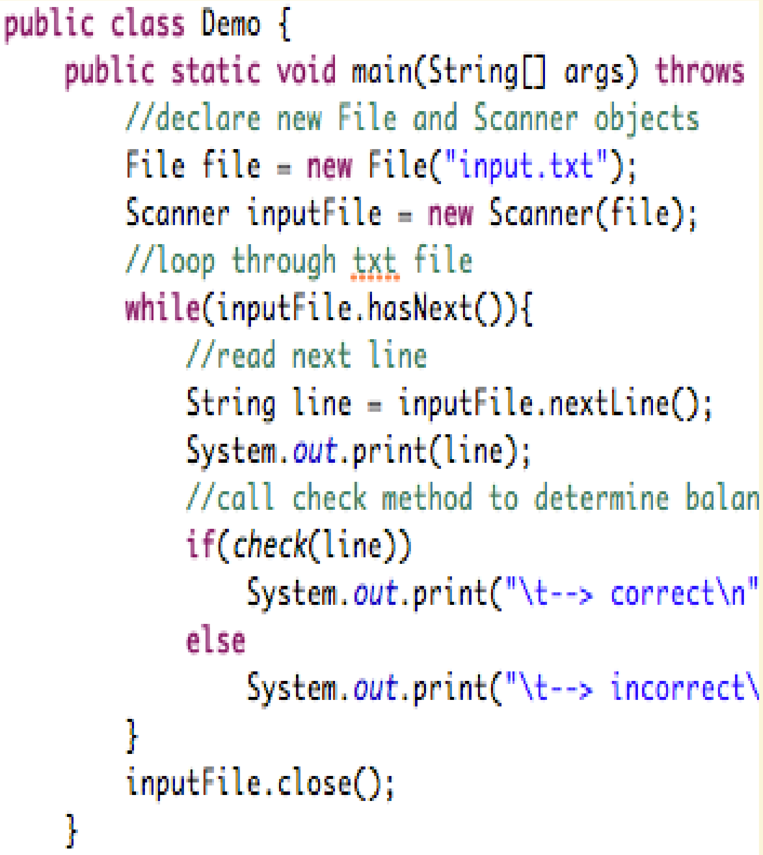
Flowchart building blocks



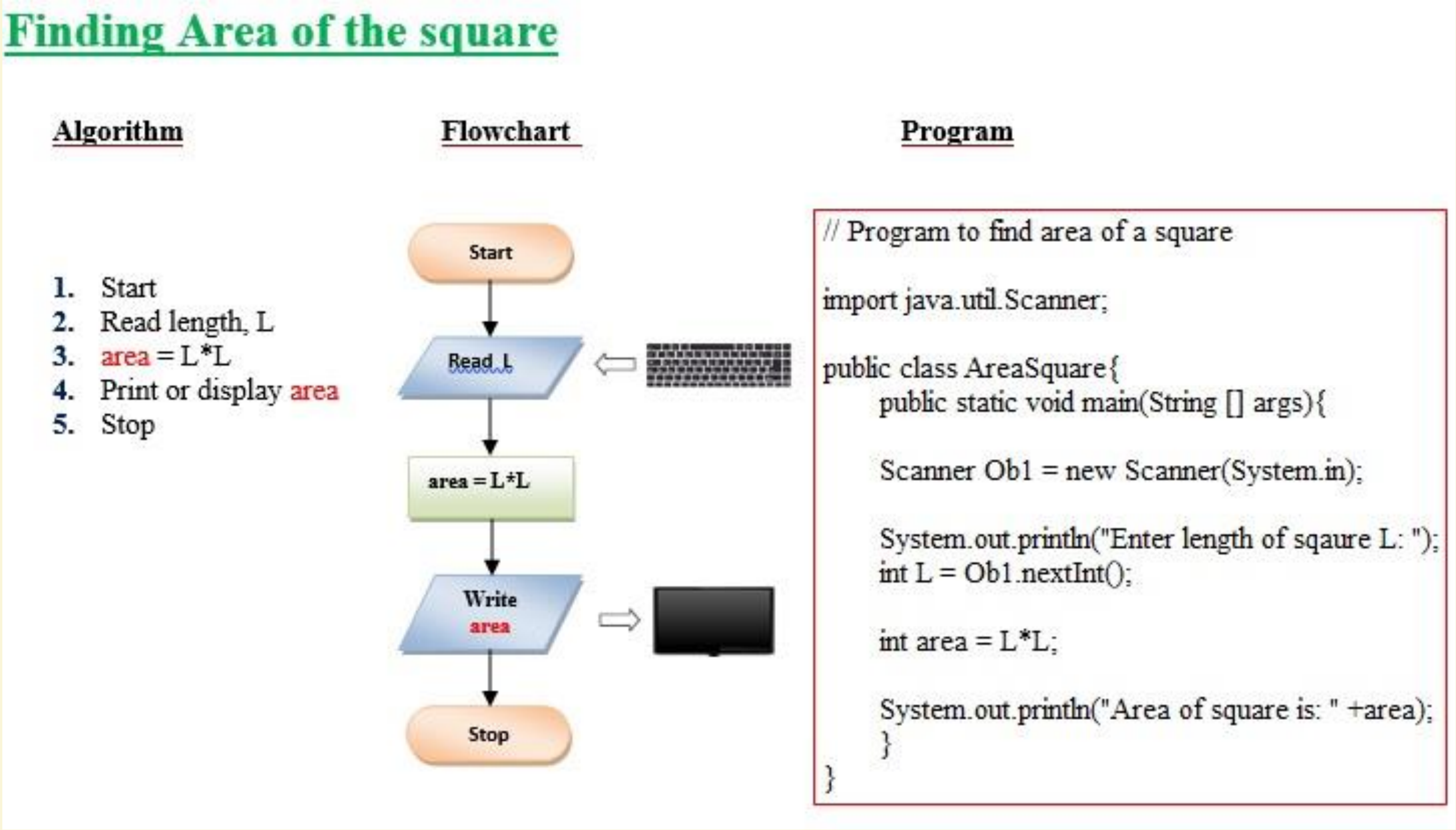
Symbol	Name	Function
⬭	Start/end	An oval represents a start or end point
→	Arrows	A line is a connector that shows relationships between the representative shapes
▱	Input/Output	A parallelogram represents input or output
▭	Process	A rectagle represents a process
◆	Decision	A diamond indicates a decision

Program

Set of instructions instructed to command to the computer to do some task.



```
public class Demo {
    public static void main(String[] args) throws
        //declare new File and Scanner objects
        File file = new File("input.txt");
        Scanner inputFile = new Scanner(file);
        //loop through txt file
        while(inputFile.hasNext()){
            //read next line
            String line = inputFile.nextLine();
            System.out.print(line);
            //call check method to determine balan
            if(check(line))
                System.out.print("\t--> correct\n"
            else
                System.out.print("\t--> incorrect\
        }
        inputFile.close();
    }
```



Finding Area of the square

Algorithm

1. Start
2. Read length, L
3. area = L*L
4. Print or display area
5. Stop

Flowchart

Program

```
// Program to find area of a square

import java.util.Scanner;

public class AreaSquare{
        public static void main(String [] args){

        Scanner Ob1 = new Scanner(System.in);

        System.out.println("Enter length of sqaure L: ");
        int L = Ob1.nextInt();

        int area = L*L;

        System.out.println("Area of square is: " +area);
        }
}
```

02

Introduction to Decision Control Structure

Decision Control Structure

A statement or set of statements that is executed when a particular condition is True and ignored when the condition is False

There are the 6 ways of exercising decision making in Java:

1. if
2. if-else
3. nested-if
4. if-else-if
5. switch-case
6. jump-break, continue, return

03

If
If-else
If-else-if
Nested-if

If Statement

It encompasses a Boolean condition followed by a scope of code that is executed only when the condition evaluates to true.

However, if there are no curly braces to limit the scope of sentences to be executed if the condition evaluates to true, then only the first line is executed.

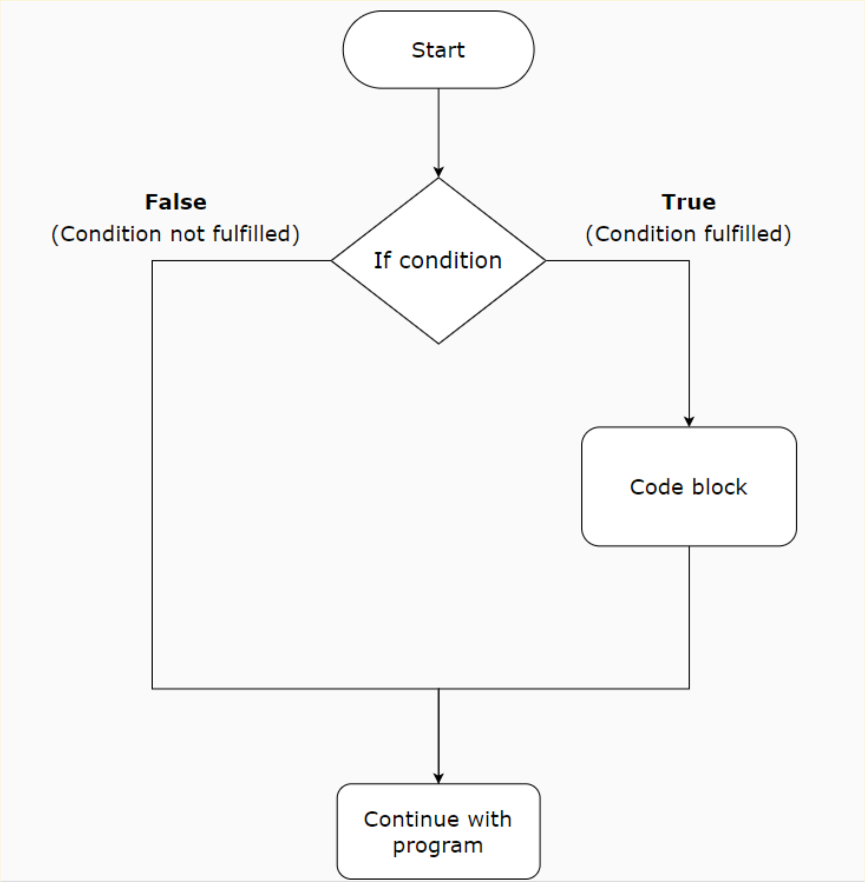
Syntax:

```
if(condition)
{

//code to be executed

}
```

If Statement



If Statement

```
if( grade >= 60 )
System.out.println( “Passed” );
```

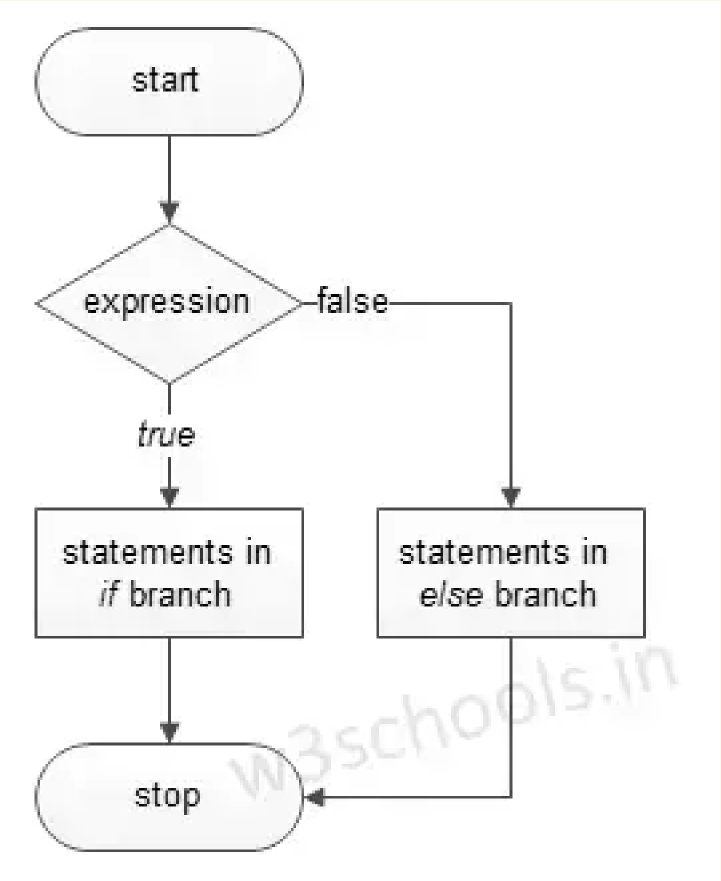
If-else Statement

This pair of keywords is used to divide a program to be executed into two parts, one being the code to be executed if the condition evaluates to true and the other one to be executed if the value is false.

Syntax:

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

If-else Statement



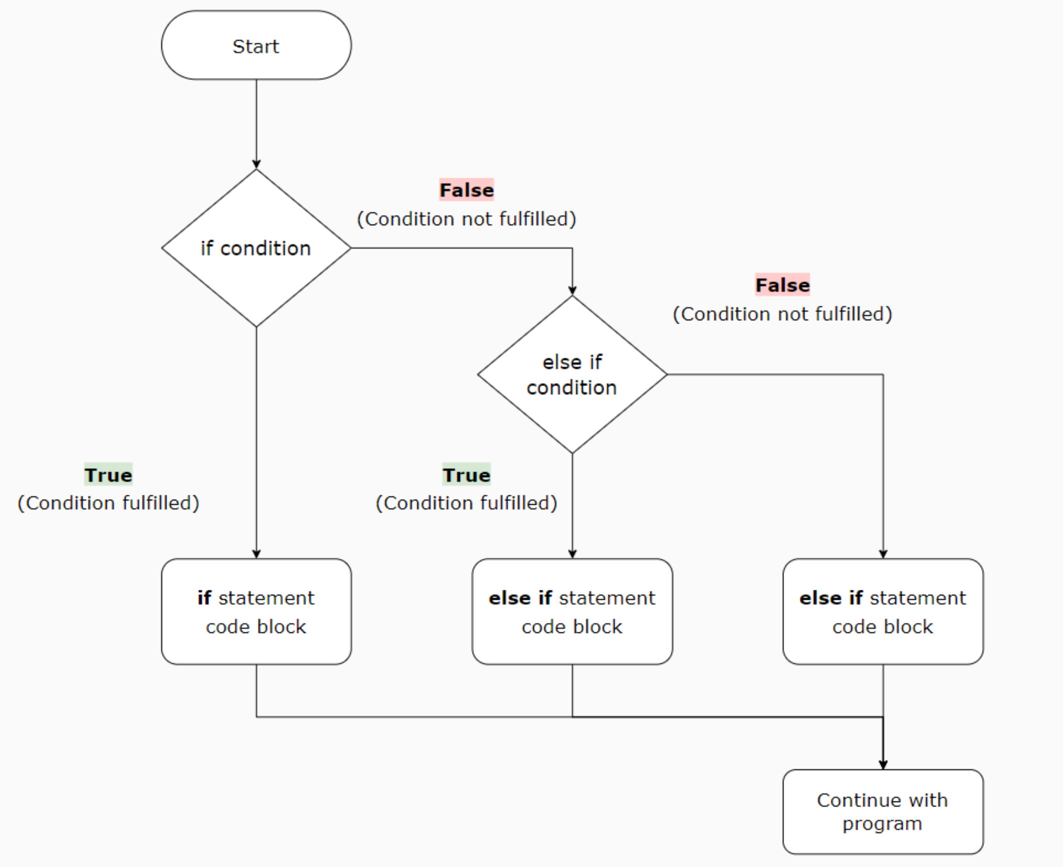
If-else-if ladder Statement

- The if statements are executed from the top down.
- As soon as one of the conditions controlling the if is true, the statement associated with that ‘if’ is executed, and the rest of the ladder is bypassed.
- If none of the conditions is true, then the final else statement will be executed.
- There can be as many as ‘else if’ blocks associated with one ‘if’ block but only one ‘else’ block is allowed with one ‘if’ block.

Syntax:

```
if (logical expression) {
    // if statements code
   block
}
else if (logical expression) {
// else if statements code
   block
}
else {
// else statements code
   block
}
```

If-else-if Ladder Statement



Nested if Statements

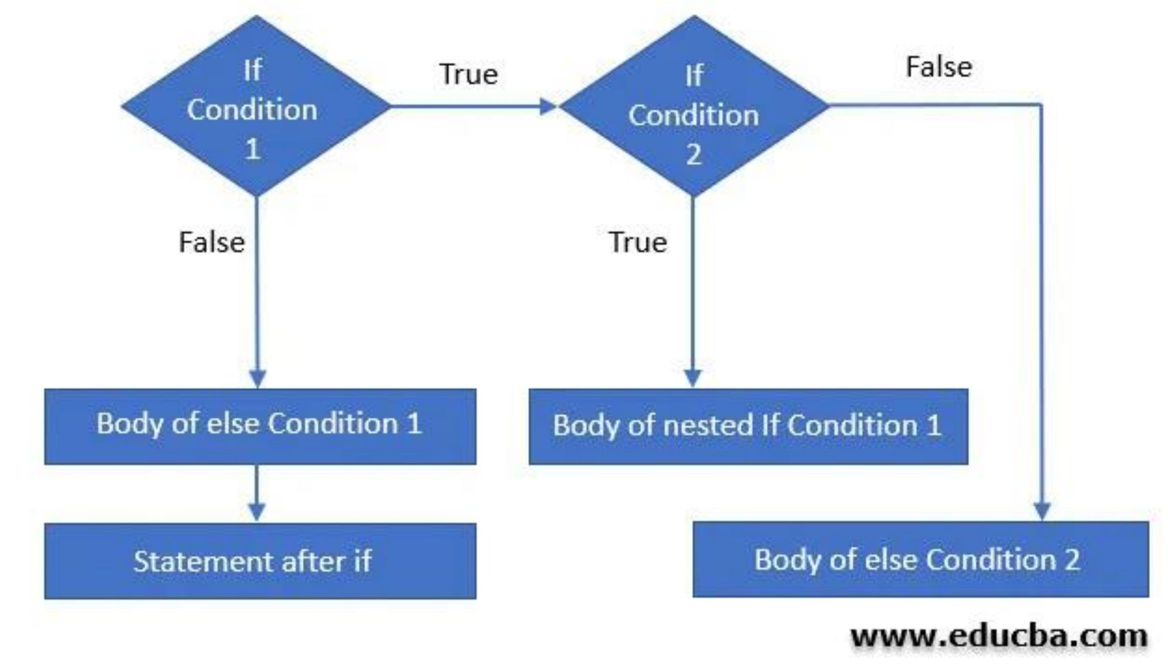
If the condition of the outer if statement evaluates to true then the inner if statement is evaluated.

Nested if’s are important if we have to declare extended conditions to a previous condition

Syntax:

```
if (condition1)
{
  // Executes when
    condition1 is satisfied
  if (condition2)
  {
    // Executes when
    condition2 is satisfied
  }
}
```

Nested if Statements



Switch Statement

The switch statement is a multiway branch statement.
It provides an easy way to dispatch execution to different parts of code based on the value of the expression.

Used to execute different cases based on equality.

Syntax:

```
switch(expression)
{
case <value1>:
//code to be executed
break;
case <value2>:
//code to be executed
break;
default:
//code to be defaultly executed
}
```

References

https://data-flair.training/blogs/decision-making-in-java/

https://www.youtube.com/watch?v=O4KGYGQvHmw

https://javatutoring.com/java-switch-case-tutorial/