Decision Making in Python

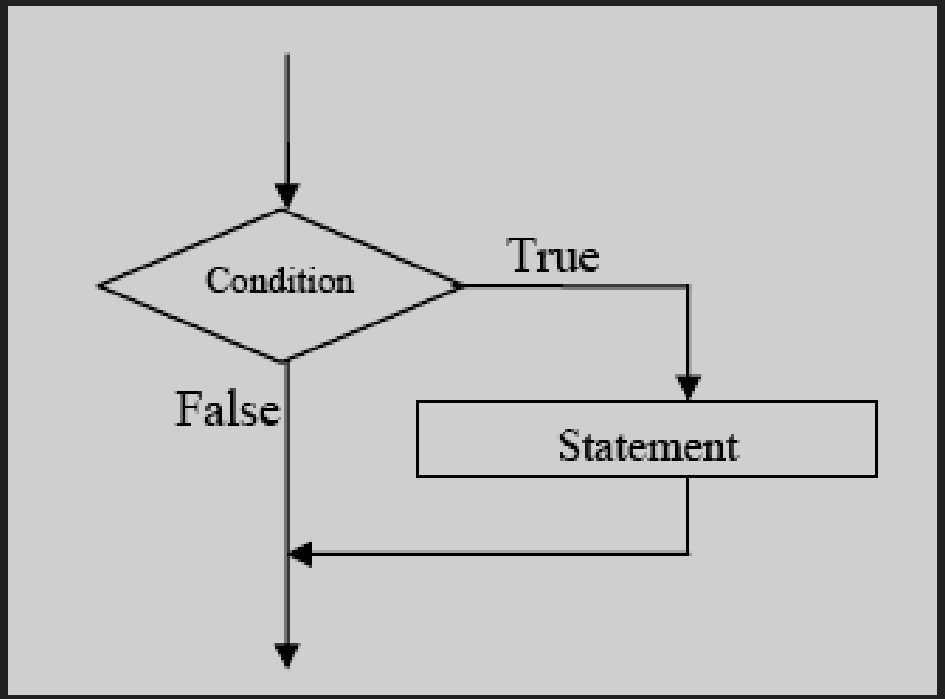
By Muhammad Umer

Decision Making in Programming

Decision making in programming is done in terms of testing an expression (logical or relational). The result of the test is either **TRUE** or **FALSE**.

A **TRUE** leads to the execution of a specified piece of code, whereas a **FALSE** leads to two possibilities; either a piece of code is executed that is different from the **TRUE** case or a branch takes place.

An important note regarding decision making structures is that they are not loops; they are executed only once.



Operators used in Decision Making

Arithmetic operators are incapable of generating **TRUE** or **FALSE**. For this we need operators that can result in **YES** and **NO**. In other words operators that can produce Boolean output.

YES	TRUE	1
NO	FALSE	0

There are three such operators in Python

1) Relational Operators OR Comparison Operators
2) Logical Operators
3) Membership Operators

Relational Operators

These operators check for equality or describe relative magnitude

All these operators ask question about the equality or inequality of the given data. The answer is YES(1) or NO(0)

Recommendation: Compare similar data-types (e.g. int with int). Dissimilar data-type comparison may lead to errors.

<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Double equal to
!=	Is not equal to

Logical Operators

- Basic logical operations are: **AND** **OR** **NOT**

Operators	***Keyword in Python***
AND	**and**
OR	**or**
NOT	**not**

AND Operator

It returns TRUE if both the operands (right side and left side) are true

Inp A	Inp B	Output
False(0)	False(0)	False(0)
False(0)	True(1)	False(0)
True(1)	False(0)	False(0)
True(1)	True(1)	True(1)

OR Operator

It returns TRUE if either of the operand (right side or left side) is true

Inp A	Inp B	Output
False(0)	False(0)	False(0)
False(0)	True(1)	True(1)
True(1)	False(0)	True(1)
True(1)	True(1)	True(1)

NOT Operator

It simply negates the input

Input	Output
True(1)	False(0)
False(0)	True(1)

Membership Operator

- These operators test for membership in a sequence such as lists, strings or tuples.
- There are two membership operators that are used in Python. **(in, not in).**
- It gives the result based on the variable present in specified sequence or string

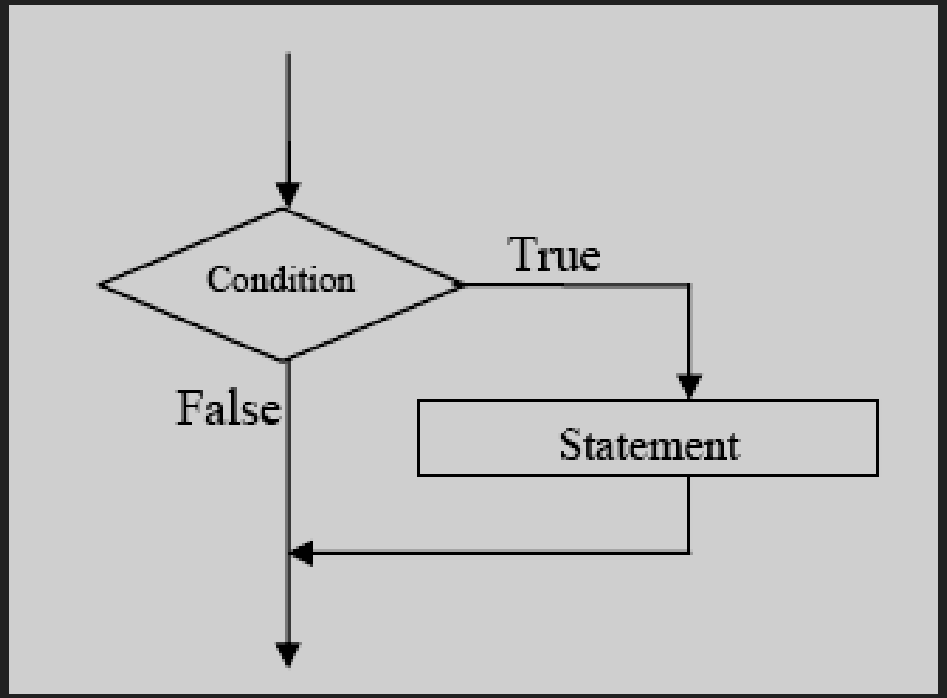
The if Statement

Syntax :

If test condition :

Block of code
that has to be
executed if
condition is true

This Indented block is called the body of if statement



The if-else statement

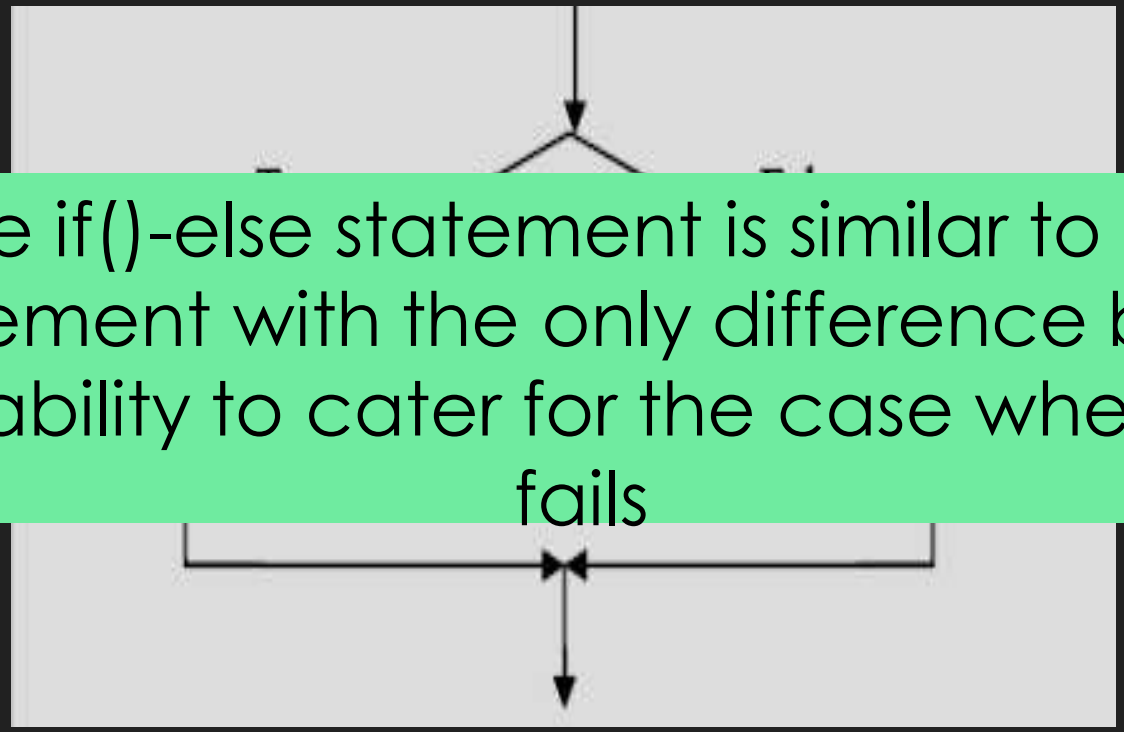
Syntax :

If test condition :

Block of code that has to be executed if condition is true

else :

Block of code that has to be executed if condition is false



➔ The if()-else statement is similar to the if() statement with the only difference being the ability to cater for the case when test fails