BCA – 502: Python Programming

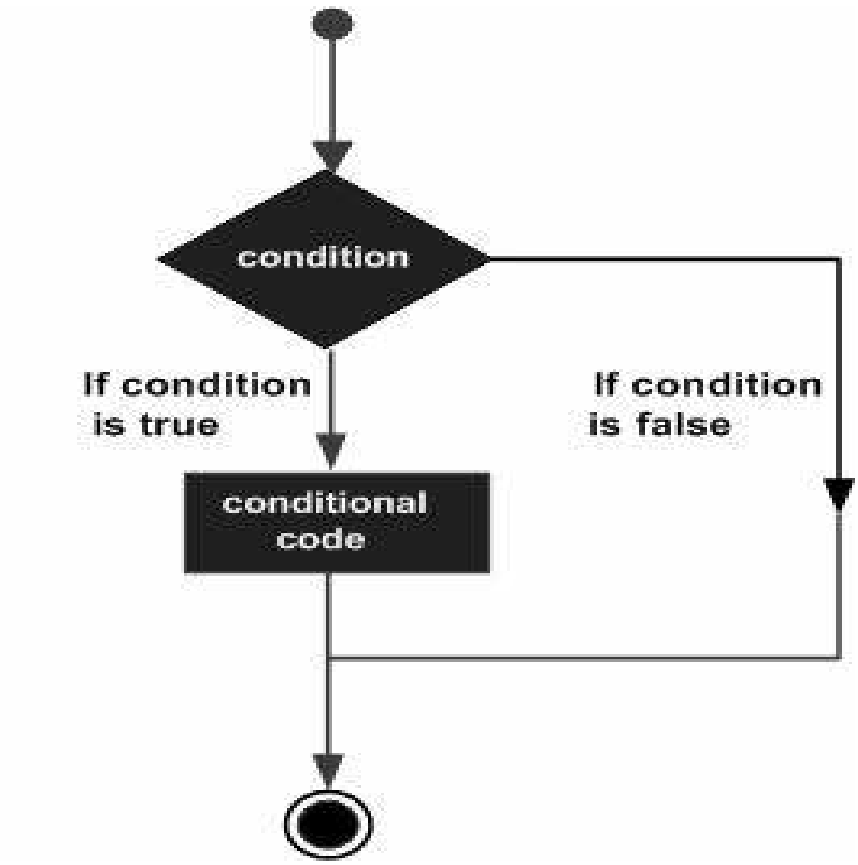
Rahul Kumar Singh

In today's Class we have discussed on Python Decision making Statements (Conditional Statements).

Decision making Statements (Conditional Statements):-

Decision making structures require that the programmer specify one or more conditions to be evaluated or tested by the program, along with a statement or statements to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false.

Following is the general form of a typical decision making structure found in most of the programming languages −



Decision structures evaluate multiple expressions which produce TRUE or FALSE as outcome. You need to determine which action to take and which statements to execute if outcome is TRUE or FALSE otherwise.

Python programming language assumes any non-zero and non-null values as TRUE, and if it is either zero or null, then it is assumed as FALSE value.

Types of Decision Making Statements:-

Python programming language provides following types of decision making statements.

1. if statements
2. if...else statements
3. nested if statements

1. if statements:-

- An if statement consists of a boolean expression followed by one or more statements.
- The if statement contains a logical expression using which data is compared and a decision is made based on the result of the comparison.
- If the boolean expression evaluates to TRUE, then the block of statement(s) inside the if statement is

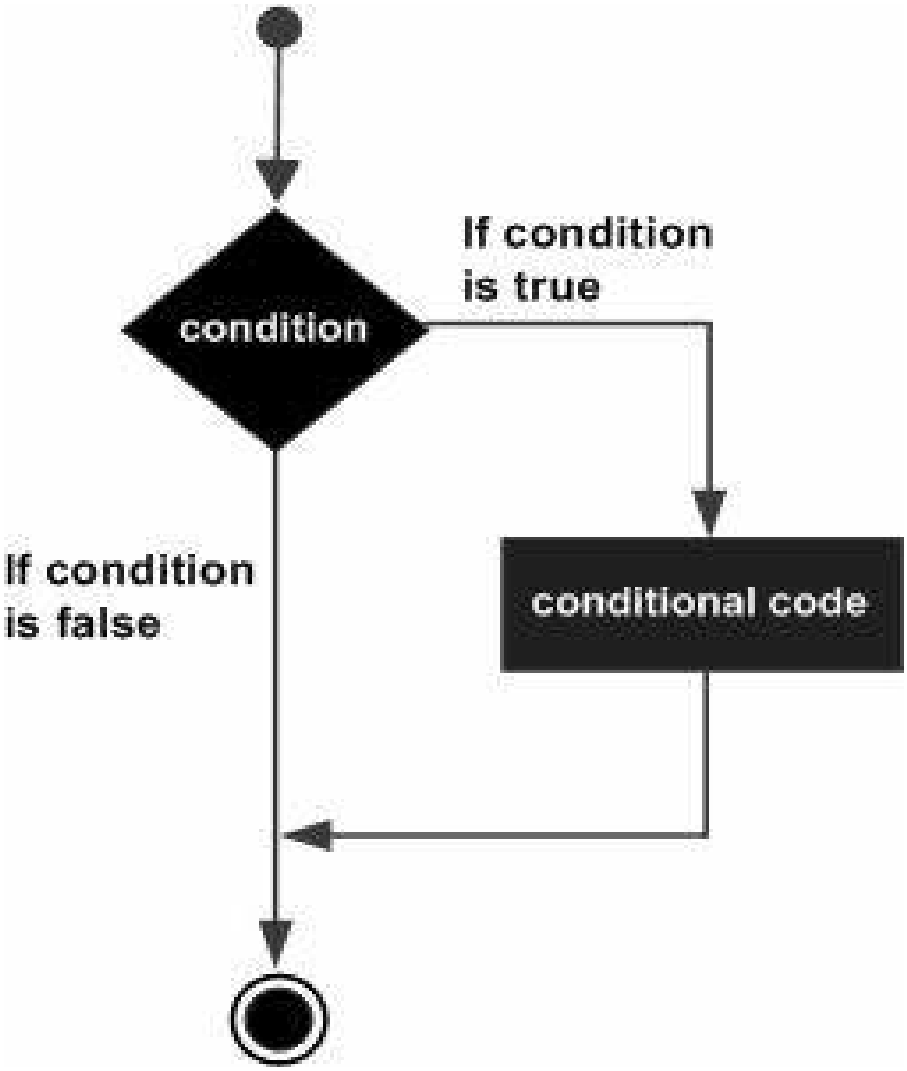
executed. If boolean expression evaluates to FALSE, then the first set of code after the end of the if statement(s) is executed.

Syntax of if statements:-

if expression:

  statement(s)

Flow Diagram of if statements:-



Example of if statements:-

```
#!/usr/bin/python

a = 33

b = 200

if b > a:

  print "b is greater than a"
```

2. if...else statements:-

- An if statement can be followed by an optional else statement, which executes when the boolean expression is FALSE.
- An else statement can be combined with an if statement. An else statement contains the block of code that executes if the conditional expression in the if statement resolves to 0 or a FALSE value.
- The else statement is an optional statement and there could be at most only one else statement following if.

Syntax:- The syntax of the if...else statement is –

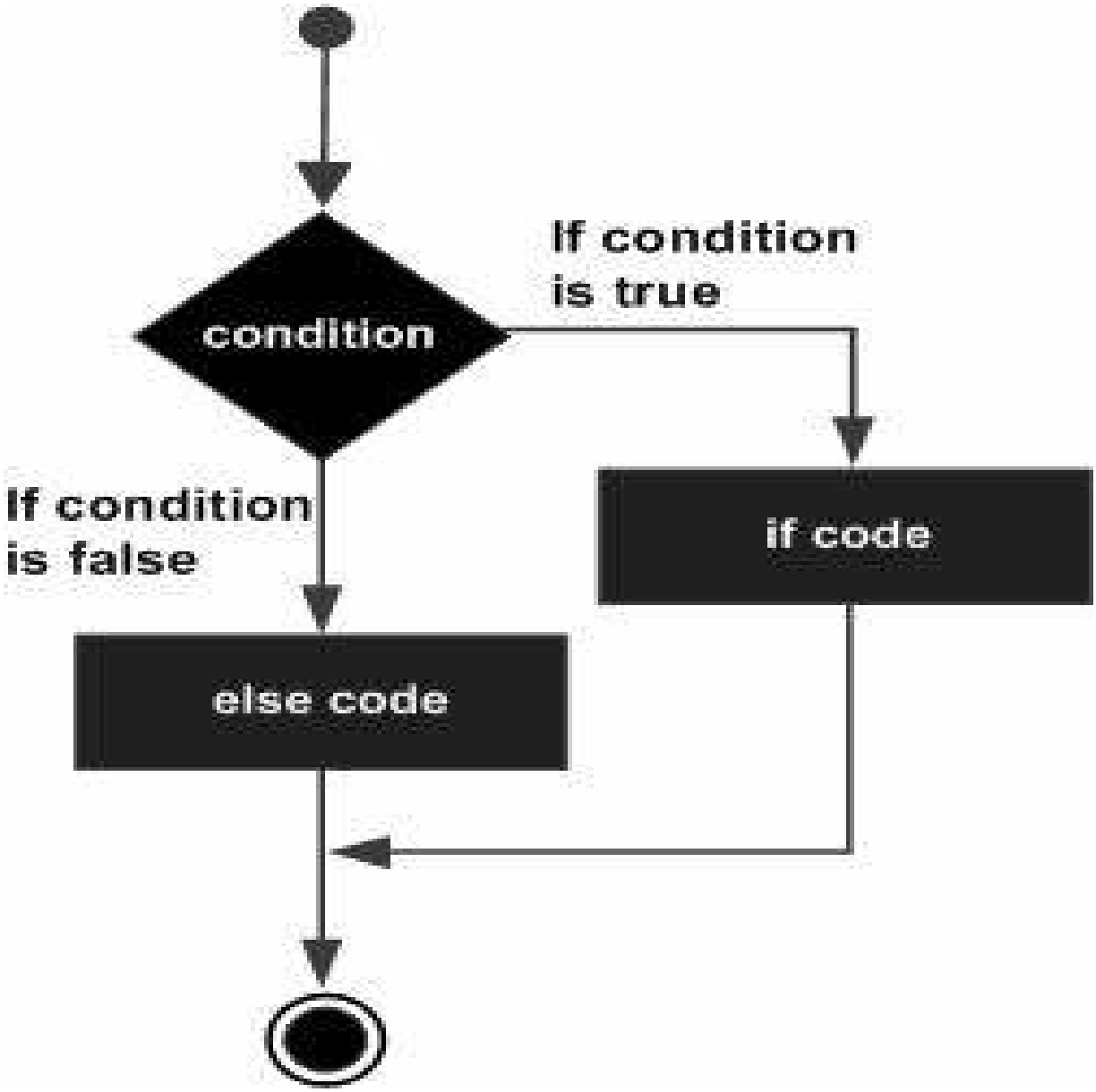
```
if expression:

   statement(s)

else:

   statement(s)
```

Flow Diagram of the if...else statement:-



Example of the if...else statement:-

```
#!/usr/bin/python

a = 200

b = 33
```

```
if b > a:
  print("b is greater than a")
else:
  print("a is greater than b")
```

The elif Statement:-

- The elif statement allows you to check multiple expressions for TRUE and execute a block of code as soon as one of the conditions evaluates to TRUE.
- Similar to the else, the elif statement is optional. However, unlike else, for which there can be at most one statement, there can be an arbitrary number of elif statements following an if.
- Core Python does not provide switch or case statements as in other languages, but we can use if..elif...statements to simulate switch case as follows−

Syntax of the elif Statement:-

```
if expression1:
   statement(s)
elif expression2:
   statement(s)
elif expression3:
```

```
  statement(s)
else:
  statement(s)
```

Example of elif Statement:-

```
#!/usr/bin/python
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

3. nested if statements:-

- ➤ You can have if statements inside if statements, this is called nested if statements.
- ➤ You can use one if or else if statement inside another if or else if statement(s).
- ➤ There may be a situation when you want to check for

another condition after a condition resolves to true. In such a situation, you can use the nested if construct.

- In a nested if construct, you can have an if...elif...else construct inside another if...elif...else construct.

Syntax

The syntax of the nested if construct may be –

```
if expression1:
   statement(s)
   if expression2:
      statement(s)
   else:
      statement(s)
```

The syntax of the nested if...elif...else construct may be –

```
if expression1:
   statement(s)
   if expression2:
      statement(s)
   elif expression3:
      statement(s)
   elif expression4:
```

```
    statement(s)
  else:
    statement(s)
else:
  statement(s)
```

Example of nested if statements

```
x = 41
if x > 10:
  print("Above ten,")
  if x > 20:
    print("and also above 20!")
  else:
    print("but not above 20.")
```

Single Statement Suites / Short Hand If:-

If the suite of an if clause consists only of a single line, it may go on the same line as the header statement.

If you have only one statement to execute, you can put it on the same line as the if statement.

Here is an example of a one-line if clause –

```
#!/usr/bin/python
if a > b: print("a is greater than b")
```

Short Hand If ... Else

If you have only one statement to execute, one for if, and one for else, you can put it all on the same line:

Example

One line if else statement:

```
#!/usr/bin/python

a = 2

b = 330

print("A") if a > b else print("B")
```

Example

One line if else statement, with 3 conditions:

```
#!/usr/bin/python

a = 330

b = 330

print("A") if a > b else print("=") if a == b else print("B")
```

And

The **and** keyword is a logical operator, and is used to combine conditional statements:

Example

Test if a is greater than b, AND if c is greater than a:

```
#!/usr/bin/python

a = 200

b = 33

c = 500

if a > b and c > a:

  print("Both conditions are True")
```

Or

The **or** keyword is a logical operator, and is used to combine conditional statements:

Example

Test if a is greater than b, OR if a is greater than c:

```
#!/usr/bin/python

a = 200

b = 33

c = 500

if a > b or a > c:

  print("At least one of the conditions is True")
```

The pass Statement

if statements cannot be empty, but if you for some reason have an if statement with no content, put in the pass statement to avoid getting an error.

Example:-

```
a = 33

b = 200

if b > a:

  pass
```

Note:- In next Class we will discuss on Loop in Python.