**CONTROL FLOWS: If Statement**

* A program statement that causes a jump of control from one part of the program to another is called control structure or control statement.
* These control statements are compound statements used to alter the control flow of the process or program depending on the state of the process.
* There are three important control structures: **Sequential**, **Alternative or Branching** and **Iterative or Looping**.
* A sequential statement is composed of a sequence of statements which are executed one after another.
* A code to print the name, address and phone number is an example of sequential statement.
* **Example :** Program to print your name and address.

print ("Hello! This is Shyam")

print ("43, Second Lane, North Car Street, TN")

* **Output:**

Hello! This is Shyam

43, Second Lane, North Car Street, TN

**ALTERNATIVE OR BRANCHING STATEMENTS:**

* These statements refer to techniques where we decide whether to do something based on some condition or criteria.
* This type of decision making is what we are to learn through alternative or branching statement.
* Python provides the following types of alternative or branching statements:

• Simple if statement

• if..else statement

• if..elif statement

**SIMPLE IF STATEMENT:**

* It is the simplest of all decision making statements.
* Condition should be in the form of relational or logical expression.
* **Syntax:** if <condition>:

statements-block1

* In the above syntax if the condition is true statements - block 1 will be executed.
* **Example: Program to verify and print whether eligible for voting**

x=int (input("Enter your age :"))

if x>=18:

print ("You are eligible for voting")

**IF..ELSE STATEMENT:**

* The if .. else statement provides control to check the true block as well as the false block.
* It checks the if condition first. If the condition is true, it executes the if block. If the condition in false, it executes the else block.
* There will be no condition to verify in the else block.
* **Syntax:**

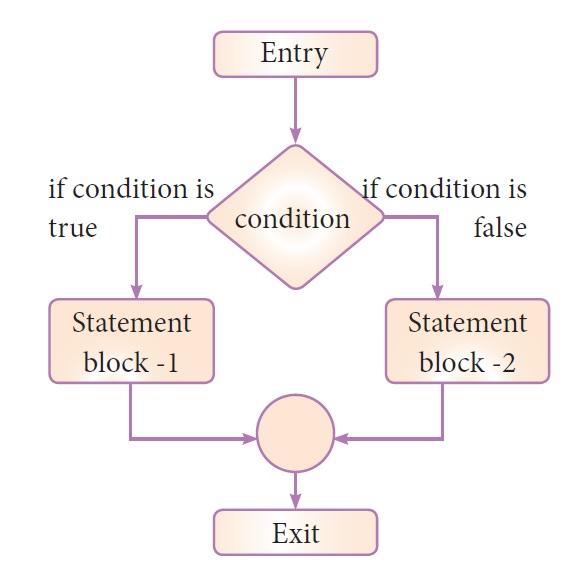
if <condition>:

statements-block 1

else:

statements-block 2

**FLOWCHART:**

****

if..else statement thus provides two possibilities and the condition determines which BLOCK is to be executed.

**IF..ELIF..ELSE STATEMENT:**

* When we need to construct a chain of if statement(s) then ‘elif’ clause can be used instead of ‘else’.
* **Syntax:** if <condition-1>:

statements-block1 #Executes if condition-1 is true

elif <condition-2>:

statements-block2 #Executes if condition-2 is true

else: #Executes if both the conditions are false

statements-block n

Example 1:

number=int(input("Enter number"))

if (number%2 == 0):

print("Even Number")

elif (number == 0):

print("Number is zero")

elif (number > 0):

print("positive number")

elif (number <0):

print("-ve")

else:

print("Odd numbers")

Example 2:

number=int(input("Enter number"))

if (number%2 == 0):

print("Even Number")

if (number == 0):

print("Number is zero")

if (number > 0):

print("positive number")

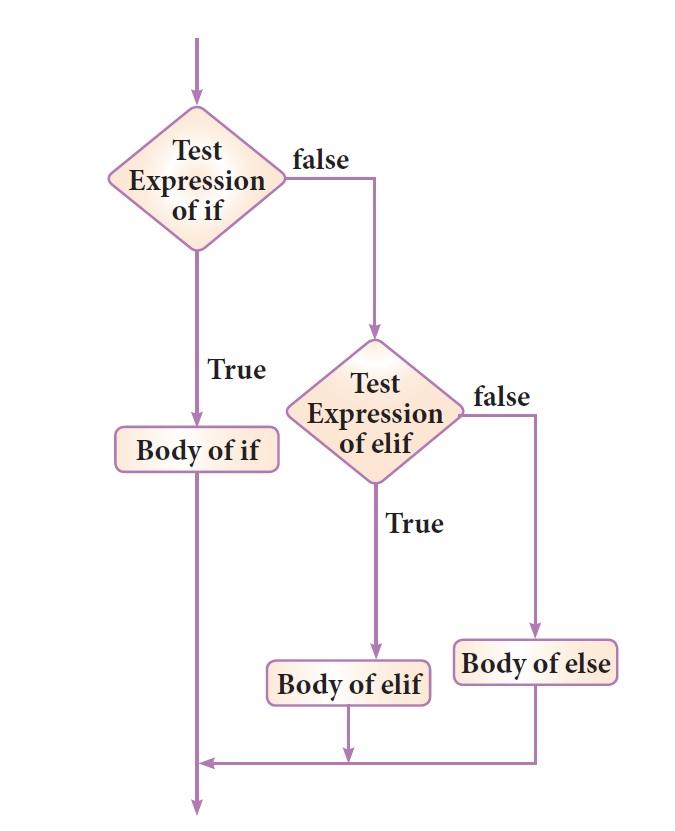
if (number <0):

print("-ve")

if (number%2 != 0):

print("odd")

**FLOWCHART:**

****

**INDENTATION:**

‘elif’ can be considered to be abbreviation of ‘else if’. In an ‘if’ statement there is no limit of ‘elif’ clause that can be used, but an ‘else’ clause should be placed at the end.

* Indentation plays a vital role in Python programming.
* Indentation is important in loops and control statements.
* Statements in a block are written with indentation.
* Usually, a block begins when a line is indented (by four spaces) and all the statements of the block should be at same indent level.
* Indentation only creates blocks and sub-blocks.
* Python Interpreter throws error for mismatched indentations.
* **EXAMPLE:**

if a%2==0:

print (a, " is an even number")

**NESTED IF STATEMENT:**

* if statement can be checked inside other if statement.
* This conditional statement is called a nested if statement.
* Inner if condition is checked **only if** outer if condition is true.
* **Syntax**:

if (condition1):

# Executes when condition1 is true

if (condition2):

# Executes when condition2 is true

# if Block ends here (condition 2)

# if Block ends here (condition 1)

number=int(input("Enter number"))

if (number > 0):

print("positive number")

if (number%2 == 0):

print("Even Number")

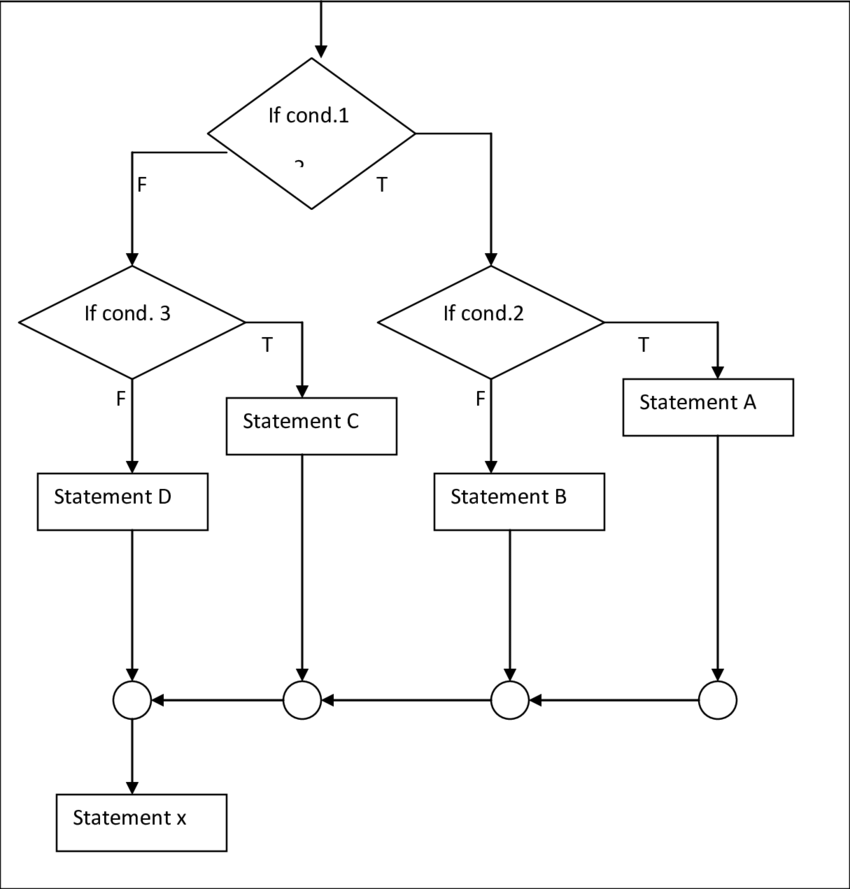
else:

print("odd")

else:

print("-ve")

**FLOWCHART**



**PASS STATEMENT:**

* The pass statement is used as a placeholder for future code.
* Empty code is not allowed in loops, function definitions, class definitions, or in if statements.
* Pass statement is used to avoid the error caused by empty line.
* **EXAMPLE:**

for x in [0, 1, 2]:

Pass

Ex 2: for i in range(10):#0,1,2,3,4,5,6,7,8,9

if i == 5:

pass

print(i)

In the above example, an empty loop would raise an error without the pass statement. This error is resolved by adding the pass statement.

1. Rahul is measuring the ph level of different solutions. He wants to write a program in python to show the nature of the solution on based on pH level following the given rule:

**pH Level - Solution**

1 - 3 strong acid

4 - 6 weak acid

7 neutral

8 -11 weak base

12-14 strong base

Else not valid ph level

Help Rahul to create a python program that takes pH level as input and print the nature of the solution on the screen.

1. What plays a vital role in Python programming?

A) Statements

B) Control

C) Structure

D) Indentation

2. Which statement is generally used as a placeholder?

A) continue

B) break

C) pass

D) goto

3. Which punctuation should be used in the blank?

if <condition>\_

statements-block 1

else:

statements-block 2

A) ; B) :

C) :: D) !

4. What keyword would you use to add an alternative condition to an if statement?

A) else if

B) elseif

C) elif

D) None of the above

5. Find the output of the given Python program?

a, b, c = 1, 3, 5

if a > 0:

if b < 2:

print("Hi")

elif c > 3:

print("Hello")

else:

print("Know Program")

A) Hi

B) Hello

C) Know Program

D) Compiled Successfully, No Output.