I

Conditionals: Conditional statement in Python (if-else statement, its working and execution), Nested-if statement and Elif statement in Python, Expression Evaluation & Float Representation.

Loops: Purpose and working of loops, While loop including its working, For Loop, Nested Loops, Break and Continue.

when we write a program, the

statements in the program are executed one by one.

This type of execution is Called 's equential execution'.

Flow Control Statement

Decision/Selection Control Stalement

Iterative/Loop Statement for while

if else if else

if Stockment

if condition: statements

first the condition is tested. If the condition is True, then the statements given after: are executed

num=1
if num==1
print("one")

Olp: One

if-else statement.

if condition: statements1

else: stolements2

the ib. else statement enecutes a group of statements when a condition is True, otherwise it will enecute another group of statements.

oc=10

if n%2==0:

print ("EVEN")

else:

print ("ODD")

Nested-if statement and elif statement when we test multiple conditions, then we use it.

if condition 1:

Statement 1

) (= int (input ("Enteranumber))

elif condition 2:

is (22>0):

Statement 2

print ("Positive")

elif condition3:

elif (cco):

statement 3

print ("Negative")

else: Statements 4

else: print ("zero")

when condition 1 is True,

Statement 1 is encuted.

When Condition 2 is True,

Statement 2 is encuted.

When Condition 3 is True,

Statement 3 is Encuted,

condition 3 is False,

when Statement 4 will be executed.

Which means Statement 4 will be encuted only if none of the Conditions are true.

Expression Evaluation In Python 2+2 is called an expression. >>> 12+2 Expressions consist of values and operators. Expressions always exaluate is they always reduce down to a single value Order of operations. >>> 2+3 46 Single value itself is an empression: 1) Parenthesis 20 2) multiplication/ Division >>> 9 ie enpression evalutes to itself 222 (2+3) 46 3) Addition/Subtraction 30 >>)(5-1) * ((7+1)/(3-1)) is matter how complicated an expression is, All enpressions are no matter how complicated an expression is, All enpressions are no matter how complicated an expression is, All enpressions are no matter how complicated an expression is, All enpressions are no matter how complicated an expression is, All enpressions are no matter how complicated an expression is, All enpressions are no matter how complicated an expression is. Enpression evaluation is done by Operator precedence. 'aaa' 4 + ((7+1)/(3-1)) 4 * (8/(3-1) 4 * (8/2) 4 + 410. Float Representation: The numbers having . (point) are float By then only prints a decimal approximation to >>> 1/3 >>>4.0 the true desimal value of the binary approximation 0. 333 333333 33333333 4.0 stored by machine, which means that Python peeps the number of digits manageable by displaying a trounded value instead 1/10 - true decimal value 22211100 L>10.1000000000000005551115123125782 7021181583404541015625

abhi-

repetitive control statement or iterative control statement loops are also called as Loops while which means we use indening here. i terate statements till some condition is True. is some as n=x+1) /n+=1 while x <= 2: print ("abc") iterate statements for number of elements in a sequence. list, str, range, tuple etc. Number of iteration is known in advance The for loop does not require an indexing variable to set beforehand, as the Command itself allows for this. a=[1,2,3] for element in sequence: for i in a: perform_action print (i) 0/b: Nested Loops It is possible to write one loop iside onother loop. eg. we can write a for loop inside for j in range (4): a while loop or a for loop inside print (i, '(t', d) 1:0 another for loop. Such loops ore called nested loops, > for j in range (4): E print(i, 't', j) for i in Mange(3). i= 2 for jinglange (4) for print (1, 1t', j) for j in range (4): print (i, 't', j)

integer > [nange] - integer sequence

for i in range (3):

print (i)

0/p:

2

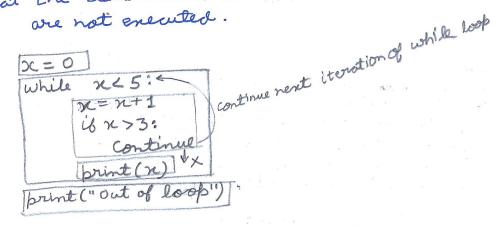
break statement: can be used inside a for loop or while loop to come out of the loop. When break is executed I the Python interpreter jumps out of the loop to process the next statement in the brogram.

Statement 1- DC = 5 Statement 2 > While IC >= 1: brint(x) statement3 -> [print("out of loop"

out of loop

The range function can generate a sequence of eg range (3) -> (0,3) Integers starting

Continue statement: is used in a loop to go back to the beginning of the loop. Which means when continue is enewted, the next repetition will start. which also means that the statements written below continue are not enewted.



1.
2
3
Out of loop