LOOPS

ITERATIVE CONTROL:

For statement

While statement

Repetedly executed the insruction so these are also refered as loops.

For loop:

```
For variable in sequence:
        Action.
```

For loop in string:

```
for letter in "python":
   print("character is ",letter)
```

output:

```
character is  p
character is  y
character is  t
character is  h
character is  o
character is  n
```

for loop in list:

```
for number in [10,20,30,40]:
   print("number is",number)
```

output:

```
===============================
number is 10
number is 20
number is 30
number is 40
```

for loop list in if condition:

```
for number in [10,20,30,40]:
    if number>=25:
        print(number,"greater then 25")
    else:
        print(number,"less than 25")
```

output:

```
10 less than 25
20 less than 25
30 greater then 25
40 greater then 25
```

While loop:

```
While condition:
          Statements:
```

Program:

```
count=1

while count<=5:

   print(count)

   count=count+1

print("good bye")
```

output:

```
1

2

3

4

5

good bye
```

two types of while loop

1)defined loop

2)indefined loop:

Indefined loop means that

Program:

```
count=1

while count<=5:

   print(count)


print("good bye")
```

output:

```
1
1
1
1
Going on
```

Now we are discussing about loop control.some situation you want to come out of the loop before completing the loop or you want to skip the loop and you want execution.

It is possible only break and continue.

Firsr we will see about break.

BREAK:

Break statement will terminates the current loop and it will continue execution of next statement.

We can take break statement in while loop

Program:

```
count=0
while count<=5:
   if count==3:
      break
   else:
      print(count)
   count=count+1
```

```
print("thank you")
```

output:

```
0
1
2
thank you
```

we can take break statement in for loop also

program:

```
for letter in "abcdef":
    if letter=='d':
        break
    else:
        print(letter)
print("thank you")
```

output:

```
a
b
c
thank you
```

<u>continue:</u>

continue statement in for loop

```
for letter in "abcdef":
    if letter=='d':
        continue
    else:
        print(letter)
print("thank you")
output:
a
b
c
e
f
thank you
```

continue statement in while loop:

```
var=10
while var>0:
    var=var-1
    if var==3:
        continue
    print(var)
print("thank you")
output:
9
```

```
8

7

6

5

4

2

1

0

thank you
```

nested loops:

```
for i in range(0,5):
    print(i)
    for j in 'python':
        print(j)
```

output:

```
=========
0
p
y
t
h
o
```

n

1

p

y

t

h

o

n

2

p

y

t

h

o

n

3

p

y

t

h

o

n

```
4
p
y
t
h
o
n
```

nested loop:

```
for i in range(1,11):
    print('Table of'+str(i))
    for j in range(1,11):
        x=i*j
        print(str(i)+'x'+str(j)+'='+str(x))
```

output:

```
=========
Table of1
1x1=1
1x2=2
1x3=3
1x4=4
1x5=5
1x6=6
```

1x7=7

1x8=8

1x9=9

1x10=10

Table of2

2x1=2

2x2=4

2x3=6

2x4=8

2x5=10

2x6=12

2x7=14

2x8=16

2x9=18

2x10=20

Table of3

3x1=3

3x2=6

3x3=9

3x4=12

3x5=15

3x6=18

3x7=21

3x8=24

3x9=27

3x10=30

Table of4

4x1=4

4x2=8

4x3=12

4x4=16

4x5=20

4x6=24

4x7=28

4x8=32

4x9=36

4x10=40

Table of5

5x1=5

5x2=10

5x3=15

5x4=20

5x5=25

5x6=30

5x7=35

5x8=40

5x9=45

5x10=50

Table of6

6x1=6

6x2=12

6x3=18

6x4=24

6x5=30

6x6=36

6x7=42

6x8=48

6x9=54

6x10=60

Table of7

7x1=7

7x2=14

7x3=21

7x4=28

7x5=35

7x6=42

7x7=49

7x8=56

7x9=63

7x10=70

Table of8

8x1=8

8x2=16

8x3=24

8x4=32

8x5=40

8x6=48

8x7=56

8x8=64

8x9=72

8x10=80

Table of9

9x1=9

9x2=18

9x3=27

9x4=36

9x5=45

9x6=54

```
9x7=63
9x8=72
9x9=81
9x10=90
Table of10
10x1=10
10x2=20
10x3=30
10x4=40
10x5=50
10x6=60
10x7=70
10x8=80
10x9=90
10x10=100
```

Using else Statement with Loops:

<u>Single statement suite:</u>

Program:

```
marks=int((input("enter marks:")))
if marks>=50 :print("pass")
else: print("fail")
```

output:

```
enter marks34
fail
enter marks:55
pass
```

For Loop Iterating by Sequence Index:

```
color=['red','blue','yellow']
for i in color:
    print(i)
```

output:

```
red
blue
yellow
```

but the index value is not there.so

```
>>> help(enumerate)
Help on class enumerate in module builtins:

class enumerate(object)
 |  enumerate(iterable, start=0)
 |
 |  Return an enumerate object.
 |
 |    iterable
 |      an object supporting iteration
 |
 |  The enumerate object yields pairs containing a count (from start,
which
 |  defaults to zero) and a value yielded by the iterable argument.
 |
 |  enumerate is useful for obtaining an indexed list:
 |      (0, seq[0]), (1, seq[1]), (2, seq[2]), ...
 |
 |  Methods defined here:
 |
 |  __getattribute__(self, name, /)
 |      Return getattr(self, name).
 |
 |  __iter__(self, /)
 |      Implement iter(self).
 |
 |  __next__(self, /)
 |      Implement next(self).
 |
 |  __reduce__(...)
 |      Return state information for pickling.
 |
 |  ----------------------------------------------------------------------
 |  Static methods defined here:
 |
 |  __new__(*args, **kwargs) from builtins.type
 |      Create and return a new object.  See help(type) for accurate
signature.
```

```
>>> color=["red","blue","yellow"]
>>> for i,j in enumerate(color):
        print(i,j)



0 red
1 blue
2 yellow
```

Otherwise we can start 1 also

```
>>> color=["red","blue","yellow"]
>>> for i,j in enumerate(color,1):
        print(i,j)
```

```
1 red
2 blue
3 yellow
```

Pass:

```
for i in range(1,101):
    if(i%2!=0):
        pass
    else:
        print(i)
print("bye")
```

output:

```
2
4
6
8
10
12
14
16
18
```

66

68

70

72

74

76

78

80

82

84

86

88

90

92

94

96

98

100

bye