* LOOP :

Loop in python are used to execute a block of code repeatedly until a certain condition is met

* There may arise situations where a block of code needs to repeated a number of times or until a condition is false. These are called loops.

* Looping (or) iterating can be done in python using while and for statements.
* Looping statement are Highly used in writing logical Program

* Loops are mainly 2 types:

1) for loop
2) while loop
3) nested loop.
   (i) for loop inside for loop
   (ii) for loop inside while loop
   (iii) while loop inside for loop
   (iv) while loop inside a while loop

1) for loop have 2 type:

(i) for loop with sequence : str, list, tuple, set, dict

(ii) for loop with range

Syntax :

For variable in sequence : (for i in S)

Statement (S)

≡

normal statement

i = variable

S = sequence or input

#Ex:

```
S= Cake
for i in S :
    Print (i)
Print ("Done")
```

O/P:

C
a
k
e
Done

#Ex :

```
S = [1, 2, 'Hi', 10.5]
for i in S :
    Print(i)
```

O/P:

1
2
Hi
10.5

* Here variable name hold one value at a time from collection and it automatically updates to the nex value in the next loop

```
# name = "Sana"
for i in name;
    print(i)
```

O/P:
S
A
N
A

```
# name = "Sana"
for i in Enumerate(name):
    print(i)
```

O/P:
(0, S)
(1, A)
(2, N)
(3, A)

Enumerate shows a position.

* Range : (start, stop, step);

* It starts from 0 by default & step increments by 1 by default

We only need to mention stop value, other 2 values are optional

```
Ex:# for i in range(5):
        print(i)
```

O/P:
1
2
3
4

```
Ex# for i in range(3,7):
        print(i)
```

O/P:
3
4
5
6

```
Ex# for i in range(0,10,2);
        print(i)
```

O/P:
0
2
4
6
8

```
# for i in range(1,6,1):
    print("Sana")
```

O/P:

Sana
Sana
Sana
Sana
Sana

```
# for position & character:
# [] is indicate position
for name = "Sana"
    for i in range(0,5,1):
        print(i,name[i])
```

O/P:

0, S
1, A
2, N
3, A

. Also: (len)

```
for i in range(0,len(Sana),1)
    print(i, name[i])
```

O/P:

0 S
1 A
2 N
3 A

```
# write a program to print even and odd number
countinuisly # 1-10 (1-odd num, 2-even num --) like this
for i in range(1,11,1):
    if(i%2==0):
        print(i,"=Even number")
    else:
        print(i,"=odd number")
```

O/P:

1 = odd number
2 = even number
3 = odd number
4 = even number
:
10 = even number

```
# write a program to print tables of given number
# x=2  x*1=y
#      x*2=y
#      x*3=y
x = int(input("Enter a num:"))
for i in range(1,11,1)
    print(f"{x}*{i} = {x*i}")
```

O/P:

5*1=5
5*2=10
5*3=15
5*4=20
:
5*10=50

write a program to print factorial of given number
Sample output = 5 = 5 * 4 * 3 * 2 * 1 = 120

```
=> x = int(input())
   for i in range(x, 0, -1):
   fact = 0
   fact =(fact * i)
   print (fact)
```

o/p :
5
5
20
60
120
120

* while loop :-

Syntax :

```
intialization
while (condition):
    Statement
    incl/decrementation
```

* while loop can execute a set of statements as long as the condition is true
* make sure to initialize the required variables before using them in the condition and also increment them within the loop to avoid an infinite loop

```
Ex#  a = 0
     while (a<5):
         print (a)
         a = a+1
```

o/p
0
1
2
3
4

Infinite loop:

There may be situation where the while statement may loop infinite times due to the condition being always true.

Ex#

```
a=5
while(a<5):
    print(a)
```

O/P

0
0
0
0
≡

→ The above program when executed doesn't stop and keeps on printing zero as it is never incremented and the condition never becomes false

Ex#

```
S = "python"
i = 0
while (i<len(S)):
    print (S[i])
    i = i+1
```

this prints as per position

O/P:

p
y
t
h
o
n

Ex#

```
S = "python"
i = 0
while (i<len(S)):
    print(S)
    i+=1
```

O/P

python
python
python
python
Python
python

Ex# Write a program to print divisibles of 3 from 1 to 20

```
i=1
while (i<=20):
    if (i%3 ==0)
        print(i)
    Else:
        pass
    i=i+1
```

O/P:

3
6
9
12
15
18

Ex#
```
i=0
while (i<7):
    print("sara")
    i+=1
```
O/P:
Sara
Sara
Sara
Sara
Sara
Sara

Ex#
```
s = "python"
i = 0
while(i<len(s)):
    print(i, s[i])
    i = i+1
```
O/P:
0 P
1 y
2 t
3 h
4 o
5 n

[ ] it indicates position.

Ex# write a program to print Even number 1 to 20
```
i = 1
while(i<=20)
    if (i%2==0):
        print(i)
    Else:
        pass
    i = i+1
```
O/P
2
4
6
8
10
12
14
16
18
20

Ex# write a program to print even and odd number countinously (1-10) => (1-odd number 2=even number) like this
```
i = 1
while(i<=20):
    if(i%2==0):
        print(i,"=evennumber")
    Else:
        print(i,"=" odd number")
    i = i+1
```
O/P
1 = odd number
2 = even number
3 = odd number
.
.
.
10 = even number

write a program to print even tables of given number (x * i = y)

```
n = int(input())
i = 1
while (i<=10):
    print(f"{n} * {i} = {n*i}")
    i+=1
```

O/P:

5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
⋮
5 x 10 = 50

write a program to calculate factorial of given number

5 = 5 x 4 x 3 x 2 x 1

```
i = 1
fact = 1
while (i<=5)
    fact = fact * i
    print (fact)
    i+=1
```

O/P

1, 2, 6, 24, 120

write a program to calculate sum of 1st 5 num from 1-5

```
S=0
i = 1
while(i<6):
    S = S+i
    print (S)
    i+=1
```

O/P:

1
3
6
10
15

fibonacci series: default {0 1}

0
1
(0+1) = 1
(1+1) = 2
(2+1) = 3
(3+2) = 5
(5+3) = 8
(8+5) = 13
⋮
n

write a program for fibonacci series of 1 to 10 value

using loop

```
a = 0
b = 1
i = 1
while(i<11):
    print(a)
    S = a+b
    a = b   # a become b
            # b become S
    b = S
    i = i+1
```

O/P:

0
1
1
2
3
5
8
13
21
34

write a program to calculate sum of digits without function.

```
num = int(input())
Sum = 0
while (num>0):
    digit = num%10      # % = remainder = 537 => 7
    Sum = Sum + digit
    num = num//10       # // = roundup = 537/10 = 53
print(Sum)
```

O/P: 25 7 = 14

* NESTED LOOP :

A nested loop is a loop inside a loop. The inner loop will be executed completely for every iteration of the outer loop.

Ex :
```
a = [0, 1, 2]
b = [3, 4, 5]
for i in a:
    for j in b:
        print(i, j)
```

O/P
```
0 3
0 4
0 5
1 3
1 4
1 5
2 3
2 4
2 5
```

* Loop Control Statements: (Break, continue, pass)

Loop control statements are used to control flow of loop in python.

* These statements can be stop or change the flow of loop

i) **Break:** it is used to stop and exist the loop when the condition is true

Ex:
```
a = 1
while (a<=5)
    print(a)
    a+=1
    if a==3
      break
```

O/P
```
1
2
```

* In above Ex, though the loop is true until a=5, but it prints only till 2

ii) **Countinue:** The countinue statement is used to stop the current iteration & countinue with the next

Ex:
```
for i in range(6):
    if i==3
        countinue
    print(i)
```

O/P
```
0
1
2
4
5
```

* The value 3 is skipped and all other values are printed

* The Else Statement:

* The Else block would be executed once, only when the condition is no longer true

* If the loop stops due to a break statement, the Else block doesn't execute

Ex:

```
a = [2, 4, 6, 8]
for i in a:
    print(i)
Else:
    print("Done")
```

O/P

```
2
4
6
8
Done
```

You can see that it loops through the list completely and once it completes, The Else blocks executes

Example with Break

```
a = [2, 4, 6, 8]
for i in a:
    if i==6
        break
    print(i)
Else:
    print("Don")
```

O/P

```
2
4
```

* Note: The Else block is not executed as the for loop has encountered Break

* The Pass Statement:

it is used when block is needed but you do nothing with it

Ex:

```
for i in range(10)
    pass
```

Ex:

```
while a)5:
    pass
```

→ you can use these when you would write a block of code afterwards and just need to write the start of a block

Nested: for loop inside for loop

Syntax:

```
for var1 in range :  ——> outer
    for var2 in range; -> Inner
        Statement2
    Statement1
```

var1 take = no of rows
var2 take = no of Elem

Ex:

```
for i in range(1, 11, 1):
    for j in range(1, 11, 1):
        print(ixj, end="\t")
    print()
```

O/P

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	–	–	–	–	–	–	–	–	30
4									40
5	–	–	–	–	–	–	–	–	50
6									60
7	–	–	–		–		–	–	70
8									80
9	–	–	–	–	–	–	–	–	90
10	20	30	40	50	60	70	80	90	100

Ex#

```
for i in range(1, 6, 1):
    for j in range(1, 6, 1):
        if (ixj%2==0):
            print((ixj), end="\t"))
        Else:
            print(" ", end="\t")
```

O/P

	2		4	
2	4	6	8	10
	6		12	
4	8	12	16	20
	10		20	

odd num are spaced

* Nested while loop inside a while loop:

Syntax:

```
intialization of outer loop
while condition1:
    intialization of inner loop
    while condition2:
        Statement of innerloop
        incldec of inner loop
    Statement of outer loop
    incldec of outer loop
```

Ex

O/P

```
i = 1
while (i<=10):
    j = 1
    while(j<=10):
        print(i*j, End="\t")
        j+=1
    print()
    i+=1
```

```
1  2  3  4  5  6  7  8  9  10
2  4  6  8  10 12 14 16 18 20
3
4  -  -  -  -  -  -  40
5                     50
6  -  -  -  -  -  -  60
7  -  -  -  -  -  -  70
8  -  -  -  -  -  -  80
9                     100
10 20 30 40 50 60 70 80 90 100
```

* Pattern programming:

O/P

```
i = 1
while (i<=5):
    j = 1
    s = 0
    while(j<=i)
        s = s+j
        print('*', End=" ")
        j+=1
    print()
    i+=1
```

```
*
* *
* * *
* * * *
* * * * *
```