

Date : 17/12/2025

---

To-Do:

- Loops
  - Data Structures
- 

## **LOOPS in Python :**

We use loop to repeat a block of code for specific times.

### **1. FOR loop**

For loop is used when the number of iteration is fixed.

#### **Syntax: iterating over a list**

```
for variable in sequence:  
    Statement
```

#### **Example:**

```
Colors = ["red", "green", "blue", "purple"]
```

```
For color in Colors:
```

```
    print(color) # here color is variable that will iterate the  
    Colors list
```

#### **Syntax: iterating over a string**

```
for variable in str:  
    Statement
```

#### **Example:**

```
For x in "MAHEK":
```

```
    print(x) #each letter will be printed separately
```

#### **→ Range() Function:**

The range() function is used to generate a sequence of numbers.

##### **Syntax**

- `range(n)`

##### **Description**

**0 to n-1**

- `range(start,stop)` start to stop-1
  - `range(start,stop,iteration)` start and iterate then stop-1

## → Nested for loop:

A nested for loop is a loop inside another for loop.

The inner loop executes completely for every single iteration of the outer loop.

Mostly used in working with rows & columns, patterns, to handle tables

## Syntax:

For outer\_var in sequence:

For inner\_var in sequence:

## statement

## Example:

```
For i in range(1,4):
```

For j in range(1,4):

print(i,j)

Python > Practical > 17.12.2025.py > ...

```
1 #For loop
2
3 colors = ["red","blue","green"]
4
5 for color in colors:
6     print(color)
7 print("-----")
8 #range() function
9
10 for x in range(1,13,2):
11     print(x)
12 print("-----")
13 #iteration over str
14
15 for i in "Python":
16     print(i)
17 print("-----")
18
19 #nested for loop
20
21 for i in range(1,4):
22     for j in range(1,4):
23         print(i,j)
24 print("-----")
25
```

```
y  
red  
blue  
green  
-----  
1  
3  
5  
7  
9  
11  
-----  
P  
y  
t  
h  
o  
n  
-----  
1 1  
1 2  
1 3  
2 1  
2 2  
2 3  
3 1  
3 2  
3 3
```

## 2. WHILE loop

A while loop executes a block of code until the given condition is true  
We use while loop when the iterations are not known.

### Syntax:

```
While condition:  
    statement
```

### Example:

```
i=1  
While i<=5:  
    print(i)  
    i=i+1
```

## 3. LOOP CONTROLS

- Break: this statement is used to terminate the loop immediately(usually put inside a if )
- Continue: this statement skip the current iteration and moves to the next iteration of the loop(again usually put inside a if)

→ Pass: this statement is used as a placeholder where a statement is syntactically required but no action is needed

```
25
26     #While loop
27
28     i=0
29     while i<=5:
30         print(i)
31         i=i+1
32     print("-----")
33
34     #break
35     i=0
36     while i<=5:
37         print(i)
38         if i==3:
39             break
40         i=i+1
41     print("-----")
42
43     #continue
44     i=0
45     while i<=5:
46         print(i)
47         if i==3:
48             i=i+1
49             continue
50         i=i+1
51     print("-----")
```

```
0
1
2
3
4
5
-----
0
1
3
-----
0
1
2
3
4
5
-----
```

## DATA STRUCTURES of Python:

## → LIST:

ordered(elements have index position)  
Mutable (can make changes)  
Allows duplicate values  
Isko print karne for loop lagega coz indexed hai

### Syntax:

```
list_name= [element1, element2, element3]
```

### Example:

```
fruits=[“orange”, “apple”, “mangooo”]
```

### Common Functions:

```
fruits.append(“pomogranet”) #add element at end of list  
fruits.insert(2, “banana”) #adds at specific position  
fruits.remove(4) #removes element  
fruit[3] = “grape” #modify  
len(fruits) # length of the list
```

```
#DATA STRUCTURES

# list

list = [1,2,3,4,5]
print(list)
"""for x in list:
    print(x)"""

list.append(29)
list.insert(3,567)
list.remove(5)
list[1]=96
len(list)

for x in list:
    print(x)
print("-----")
```

```
[1, 2, 3, 4, 5]
1
96
3
567
4
29
-----
-----
```

## → TUPLE:

Ordered  
Immutable(no changes can be made, therefore NO function here)

Allows duplicate values  
Faster and more efficient than list  
Isko print karne for loop lagega coz indexed hai

**Syntax:**

tuple\_name=(element1, element2)

**Example:**

Vehicle = ("car", "bike", "truck")

**→ SET:**

Unordered  
Mutable  
Does not allow duplicate values  
No indexing  
Isko direct print kar sakte coz indexing ni hai

**Syntax:**

set\_name= {ele1, ele2, ele3}

**Example:**

S ={1 ,2 ,3 ,3 ,5}  
print(s)

**Common Operations:**

s.add(4) # this will add the elements, aur indexing ni hoti  
tho basically random kahi bhi add hogta  
s.remove(2 ) # 2 element ko nikalega

**→ DICTIONARY:**

Stores data in key: value format  
Key must be unique  
Values can be duplicate  
Fast lookup

**Syntax:**

Dict\_name = {key1: value1, key2: value2}

**Example:**

Student = { "name": "mahek", "surname": "Junnedi"}

**Common Operations:**

Student["name"] = "Zia" #update

```
Student["age"] = 18 #add  
Del Student["name"]
```

```
#Tuple  
  
tup=("mahek","sayma","zia")  
  
for x in tup:  
|   print(x)  
print("-----")  
  
#set  
  
set1={1,2,3,4,5,5,6}  
  
print(set1)  
print("-----")  
  
#Dictionary  
  
dic = {"name" : "Mahek","surname": "Junneddi","age" : 19}  
print(dic)  
  
dic["age"]=18  
dic["course"] = "python"  
  
print(dic)  
print("-----")
```

```
mahek  
sayma  
zia  
-----  
{1, 2, 3, 4, 5, 6}  
-----  
{'name': 'Mahek', 'surname': 'Junneddi', 'age': 19}  
{'name': 'Mahek', 'surname': 'Junneddi', 'age': 18, 'course': 'python'}  
-----
```

