loops

July 20, 2025

0.1 In Python, loops are used to repeat a block of code multiple times. There are two main types:

- For loop
- while loop

[4]:

```
# Initialization While------

#counter = initial_value
#while condition:
    #statements
    #update counter (to avoid infinite loop)

# Initialization For------

#for variable in iterable:
#    statements
for i in range(1, 6):
    print("Count:", i)
```

```
Count: 1
Count: 2
Count: 3
Count: 4
Count: 5
```

[1]:

```
# while loop
number = 0
while number < 10:
    print(number)
    number = number + 1
```

```
0
1
2
3
4
5
6
```

```
7
8
9
```

[3]:

```
help('while')
```

```
The "while" statement
*********************

The "while" statement is used for repeated execution as long as an
expression is true:

   while_stmt ::= "while" assignment_expression ":" suite
                  ["else" ":" suite]

This repeatedly tests the expression and, if it is true, executes the
first suite; if the expression is false (which may be the first time
it is tested) the suite of the "else" clause, if present, is executed
and the loop terminates.

A "break" statement executed in the first suite terminates the loop
without executing the "else" clause’s suite.  A "continue" statement
executed in the first suite skips the rest of the suite and goes back
to testing the expression.

Related help topics: break, continue, if, TRUTHVALUE
```

[2]:

```
# while loop
number = 0
while number < 10:
    print(number)
    number = number + 1
```

```
0
1
2
3
4
5
6
7
8
9
```

[3]:

```
# Example 2: Calculate 100th factorial
i = 1
result = 1
```

```python
if i == 0:
    result = 1
else:
    while (i <= 4):
        result = result * i
        i += 1
print(result)
```

```
24
```

[10]:

```python
# Example 3: Input number from user and compute the sum 1+2+3+4+....+n
num = int(input("Enter number"))
i = 1
sum = 0
while (i <= num):
    sum = sum + i
    i = i + 1
print(sum)
```

```
Enter number 4
```

```
10
```

[11]:

```python
# Example 3: Input number from user and compute the sum 1*2*3*4*....*n

n = int(input("Enter number for multiple"))
total = 1
i = 1
while (i <= n):
    total = total * i
    i = i + 1
print(total)
```

```
Enter number for multiple 4
```

```
24
```

[14]:

```python
#Example 4: while loop iterates over the elements until a certain condition
list1 = ['Learning', 'is', 'fun', 'with', 'me']
indx = 0
while(indx < len(list1)):
    print(list1[indx])
    indx = indx + 1
```

```
Learning
is
fun
with
me
```

[ ]:

```
# Example 5: Using iterables inside a while loop expression
mylist = ['Arif', 'Hadeed','Mujahid', 'Maaz']
print("mylist before the loop: ", mylist)
x = 1
while mylist:
    print("my list remaining: ", mylist)
    x = x + 1
    mylist.pop()
print("My list after pop", mylist)
```

[ ]:

```
# Example 6: Print Fibonacci series
#Every next number is the sum of the previous two numbers.
n = int(input("Enter number: "))
i = 1
if n < 1:
    fib = []
elif n == 1:
    fib = [0]
elif n == 2:
    fib = [0, 1]
elif n > 2:
    fib = [0, 1]
    while (i < n-1):
        fib.append(fib[i] + fib[i - 1])
        i = i + 1
print("Required fib", fib)

```

[36]:

```
n = int(input("Enter number: "))
i =1
fib = [0, 1] # important to initlaize
while (i < n-1):
        fib.append(fib[i] + fib[i - 1])
        i = i + 1
print("Required fib", fib)
```

```
Enter number:  10
```

```
Required fib [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

[ ]:

```
a = [1,2,3,4,5]
a.sort(reverse=True)  # sort descending
while (a):
    print("Outer: ", a.pop())
```

[45]:

```
# Example 1: A while loop nested inside another while loop
# Note the inner while loop works on a list that is declared again and again␣
 ↪inside the outer loop
```

```
a = [1,2,3,4,5]
while (a):
    print("Outer: ", a.pop())
    b= ["qaisar","imtiaz","Hussain"]
    while (b):
        print("\t Inner: ", b.pop())
```

```
Outer:  5
         Inner:  Hussain
         Inner:  imtiaz
         Inner:  qaisar
Outer:  4
         Inner:  Hussain
         Inner:  imtiaz
         Inner:  qaisar
Outer:  3
         Inner:  Hussain
         Inner:  imtiaz
         Inner:  qaisar
Outer:  2
         Inner:  Hussain
         Inner:  imtiaz
         Inner:  qaisar
Outer:  1
         Inner:  Hussain
         Inner:  imtiaz
         Inner:  qaisar
```

[1]:

```
#Example 1: Breaking an infinite while loop on a certain condition
n = 1
while (True):
    n = n + 1
    if (n == 5):
        break
    print(n)
```

```
2
3
4
```

[4]:

```
#table in while loop
i = 1
tbl = 2
result = 0
while (i <= 10):
    result = tbl * i
    print(tbl," X ",i," = ", result)
    i = i + 1
```

```
2  X  1  =  2
2  X  2  =  4
2  X  3  =  6
2  X  4  =  8
2  X  5  =  10
2  X  6  =  12
2  X  7  =  14
2  X  8  =  16
2  X  9  =  18
2  X  10  =  20
```

[7]:

```
#table in while loop
i = 1
tbl = int(input("Enter table: "))
result = 0
while (i <= 10):
    result = tbl * i
    print(tbl," X ",i," = ", result)
    i = i + 1
```

```
Enter table:  4
```

```
4  X  1  =  4
4  X  2  =  8
4  X  3  =  12
4  X  4  =  16
4  X  5  =  20
4  X  6  =  24
4  X  7  =  28
4  X  8  =  32
4  X  9  =  36
4  X  10  =  40
```

[6]:

```
#So the iter() and the next() functions makes the basis of a for loop inPython
#Basis For Loop
# Example 1
my_for_list = ['Arif', 'Hadeed', 'Muhahid']
for i in my_for_list:
    print(i)
```

```
Arif
Hadeed
Muhahid
```

[5]:

```
str_n = ("Qaisar Imtiaz")
for i in str_n:
    print(i)
```

```
Q
a
i
s
a
r

I
m
t
i
a
z
```

[12]:
```
table_no = 4
i = 1
for i in range(1, 11):
    results = table_no * i
    print(table_no," X ",i," = ", results)
```

```
4  X  1  =  4
4  X  2  =  8
4  X  3  =  12
4  X  4  =  16
4  X  5  =  20
4  X  6  =  24
4  X  7  =  28
4  X  8  =  32
4  X  9  =  36
4  X  10  =  40
```

[13]:
```
table_no = int(input("Enter table value: "))
i = 1
for i in range(1, 11):
    results = table_no * i
    print(table_no," X ",i," = ", results)
```

```
Enter table value:  6
```

```
6  X  1  =  6
6  X  2  =  12
6  X  3  =  18
6  X  4  =  24
6  X  5  =  30
6  X  6  =  36
```

```
6  X  7  =  42
6  X  8  =  48
6  X  9  =  54
6  X  10  =  60
```

[14]:

```python
# Example 3: Iterate a tuple using for loop
tple_fruits = ("mango","orange","bananna","apple")
for frts in tple_fruits:
    print(frts)
```

```
mango
orange
bananna
apple
```

[21]:

```python
# Example 4: Iterate a string using for loop and count the count of a specific␣
 ↪character
word = "Welcome to my channel"
count = 0
for chr in word:
    if chr == "l":
        count = count + 1
print(count)
```

```
2
```

[22]:

```python
# Example 4: Iterate a string using for loop and count the count of a specific␣
 ↪character
word = "Welcome to my channel"
count = 0
chr_input = input("Enter character: ")
for chr in word:
    if chr == chr_input:
        count = count + 1
print(count)
```

```
Enter character:  m
```

```
2
```

[20]:

```python
#Example 5: Iterating through a dictionary keys
dic = {
    'Name':'Qaisar',
    'Gender':'Male',
    'Age':45,
    'Height':5.8,
    'occupation':'Student'
}
```

```python
print("Iterating through dictionary keys:")
for lst in dic:
    print(lst)
print("\nAnother way of iterating through dictionary keys:")
for i in dic.keys():
    print(i)

#Iterating through the values
print("\nIterating through a dictionary values:")
for vlu in dic.values():
    print(vlu)

# Iterating through the key-value pairs
print("\nIterating through a key-value pairs:")
for prs in dic.items():
    print(prs)

print("\nIterating through a key-value pairs:")
for ky,vl in dic.items():
    print(ky,": ",vl)
```

```
Iterating through dictionary keys:
Name
Gender
Age
Height
occupation

Another way of iterating through dictionary keys:
Name
Gender
Age
Height
occupation

Iterating through a dictionary values:
Qaisar
Male
45
5.8
Student

Iterating through a key-value pairs:
('Name', 'Qaisar')
('Gender', 'Male')
('Age', 45)
('Height', 5.8)
```

```
('occupation', 'Student')

Iterating through a key-value pairs:
Name :  Qaisar
Gender :  Male
Age :  45
Height :  5.8
occupation :  Student
```

[23]:

```python
#Print all elements of list ignoring string "arif"
#The pass statement in a for loop
pass_lst = ["qaisar","arif","danish","ejaz","naeem"]

for ps in pass_lst:
    if ps == "arif":
        pass
    else:
        print(ps)
```

```
qaisar
danish
ejaz
naeem
```

[28]:

```python
# Print the string ignoring spaces
str1 = "This is great stuff"
for st in str1:
    if st == " ":
        pass
    else:
        print(st, end="")
```

```
Thisisgreatstuff
```

[43]:

```python
#Using range() Function in for Loops
print(range(5))
print(list(range(10)))
print(list(range(-5, 2, 1)))
print(list(range(20, 2, -3)))
print(list(range(20, 0, -2)))
print(list(range(2, 20, 2)))
```

```
range(0, 5)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[-5, -4, -3, -2, -1, 0, 1]
[20, 17, 14, 11, 8, 5]
[20, 18, 16, 14, 12, 10, 8, 6, 4, 2]
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18]
```

[44]:

```
r = range(5)
for rg in r:
    print(rg)
```

```
0
1
2
3
4
```

[46]:

```
for i in range(3, 10):
    print(i)
```

```
3
4
5
6
7
8
9
```

[8]:

```
# Example 4: Used to iterate over Lists, when you need to track the index of␣
 ↪elements while iterating.
friends = ['Rauf','Arif', 'Maaz', 'ahmed', 'jabar']
for nam in range(len(friends)):
    print("The values at position {} is {}".format(nam, friends[nam]))
```

```
The values at position 0 is Rauf
The values at position 1 is Arif
The values at position 2 is Maaz
The values at position 3 is ahmed
The values at position 4 is jabar
```

[7]:

```
#Using enumerate() Function in for Loops
fds = ['Rauf','Arif', 'Maaz', 'ahmed', 'jabar']
for index, nam in enumerate(fds):
    print("The values at position {} is {}".format(index, nam))
```

```
The values at position 0 is Rauf
The values at position 1 is Arif
The values at position 2 is Maaz
The values at position 3 is ahmed
The values at position 4 is jabar
```

[16]:

```
fds = ['Rauf','Arif', 'Maaz', 'ahmed', 'jabar']
rv = enumerate(fds)
```

```python
print(type(rv))
print(list(enumerate(fds)))
print(dict(enumerate(fds)))
```

```
<class 'enumerate'>
[(0, 'Rauf'), (1, 'Arif'), (2, 'Maaz'), (3, 'ahmed'), (4, 'jabar')]
{0: 'Rauf', 1: 'Arif', 2: 'Maaz', 3: 'ahmed', 4: 'jabar'}
```

[18]:

```python
#Use of break and continue statement inside a for loop
# Example 1: Break the loop when it reaches the element "cherry"
fruits = ["apple", "banana", "cherry", "guava"]
for fr in fruits:
    if fr == "cherry":
        break
    print(fr)
```

```
apple
banana
```

[19]:

```python
# Do not print banana from the list
fruits = ["apple", "banana", "cherry", "guava"]
for fr in fruits:
    if fr == "banana":
        continue
    print(fr)
```

```
apple
cherry
guava
```

[31]:

```python
#Example 3: Print odd numbers from 1 to 11
for i in range(1,12):
    if i%2 == 0:
        continue
    print(i)
```

```
1
3
5
7
9
11
```

[22]:

```python
for i in range(1,11,2):
    print(i)
```

```
1
3
```

```
5
7
9
```

[33]:

```
#A for loop with else and break
mydict = {
'anas':90,
'mehtab':95,
'farooq':81,
'asad':77,
'taha':86,
'saliq':100
}
student_name = input('enter name: ')
for name in mydict.keys():
    if name == student_name:
        print(mydict[name])
        break
    else:
        print('No entry with that name found.')
```

```
enter name:  anas
```

```
90
```

[35]:

```
# Example: A for loop nested inside another for loop
# Note the inner for loop works on a list that is declared again and again␣
 ↪inside the outer loop
list1 = [1,2,3,4]
for numb in list1:
    print("Outer: ", numb)
    list2 = ['Arif', 'Rauf']
    for name in list2:
        print("\t Inner: ", name)
print("Outside loops")
```

```
Outer:  1
         Inner:  Arif
         Inner:  Rauf
Outer:  2
         Inner:  Arif
         Inner:  Rauf
Outer:  3
         Inner:  Arif
         Inner:  Rauf
Outer:  4
         Inner:  Arif
         Inner:  Rauf
```

```
Outside loops
```

[39]:

```
#List Comprehension
old_list = [5 , 7, 3, 6, 8]
new_list = []
for i in old_list:
    new_list.append(i*i)
new_list
```

[39]:

```
[25, 49, 9, 36, 64]
```

[40]:

```
oldlist = [5, 3, 6, 2]
newlist = [i*i for i in oldlist]
newlist
```

[40]:

```
[25, 9, 36, 4]
```

[43]:

```
list1 = [1, 9, 12, 88, 65, 7, 20, 55, 47, 32]
newlist = []
for i in list1:
    if i%2 == 0:
        newlist.append(i)
newlist
```

[43]:

```
[12, 88, 20, 32]
```

[44]:

```
#Dictionary Comprehension
list1 = [1, 2, 3, 4, 5]
dict1 = {key: key**3 for key in list1}
print (dict1)
```

```
{1: 1, 2: 8, 3: 27, 4: 64, 5: 125}
```

[ ]: