Python For Loop

- The for loop in Python is used to iterate over a sequence (list, tuple, string) or other iterable objects. Iterating over a sequence is called traversal.

Syntax

```
for val in sequence:
 Body of for
```

- Here, val is the variable that takes the value of the item inside the sequence on each iteration.
- Loop continues until we reach the last item in the sequence. The body of for loop is separated from the rest of the code using indentation.

Flowchart:

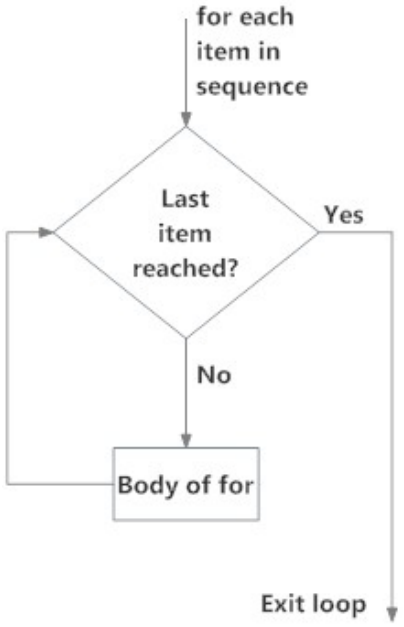

Fig: operation of for loop

Example:

In [1]:
```
#Find product of all numbers present in a list

lst = [10, 20, 30, 40, 50]

product = 1
#iterating over the list
for ele in lst:
    product *= ele

print("Product is: {}".format(product))
```

```
Product is: 12000000
```

The range() function

- We can generate a sequence of numbers using range() function. range(10) will generate numbers from 0 to 9 (10 numbers).
- We can also define the start, stop and step size as range(start,stop,step size). step size defaults to 1 if not provided.
- This function does not store all the values in memory, it would be inefficient. So it remembers the start, stop, step size and generates the next number on the go.
- To force this function to output all the items, we can use the function list().

Example:

In [2]:
```
#print range of 10
for i in range(10):
    print(i)
```

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

In [3]:
```
#print range of numbers from 1 to 20 with step size of 2
for i in range(0, 20, 5):
    print(i)
```

```
0
5
10
15
```

In [4]:
```
lst = ["Dinkar", "Mahadevi", "Nirala", "Pant", "Jai Sankar"]

#iterate over the list using index
#for index in range(len(lst)):
#    print(lst[index])
for ele in lst:
    print(ele)
```

```
Dinkar
Mahadevi
Nirala
Pant
Jai Sankar
```

for loop with else

- A for loop can have an optional else block as well. The else part is executed if the items in the sequence used in for loop exhausts.
- break statement can be used to stop a for loop. In such case, the else part is ignored.
- Hence, a for loop's else part runs if no break occurs.

In [5]:
```
numbers = [1, 2, 3]

#iterating over the list
for item in numbers:
    print(item)
else:
    print("no item left in the list")
```

```
1
2
3
no item left in the list
```

In [6]:
```
for item in numbers:
    print(item)
    if item % 2 == 0:
        break
else:
    print("no item left in the list")
```

```
1
2
```

Python Program to display all prime numbers within an interval

In [7]:
```
index1 = 20
index2 = 50

print("Prime numbers between {0} and {1} are :".format(index1, index2))

for num in range(index1, index2+1):      #default step size is 1
    if num > 1:
        isDivisible = False;
        for index in range(2, num):
            if num % index == 0:
                isDivisible = True;
        if not isDivisible:
            print(num)
```

```
Prime numbers between 20 and 50 are :
23
29
31
37
41
43
47
```