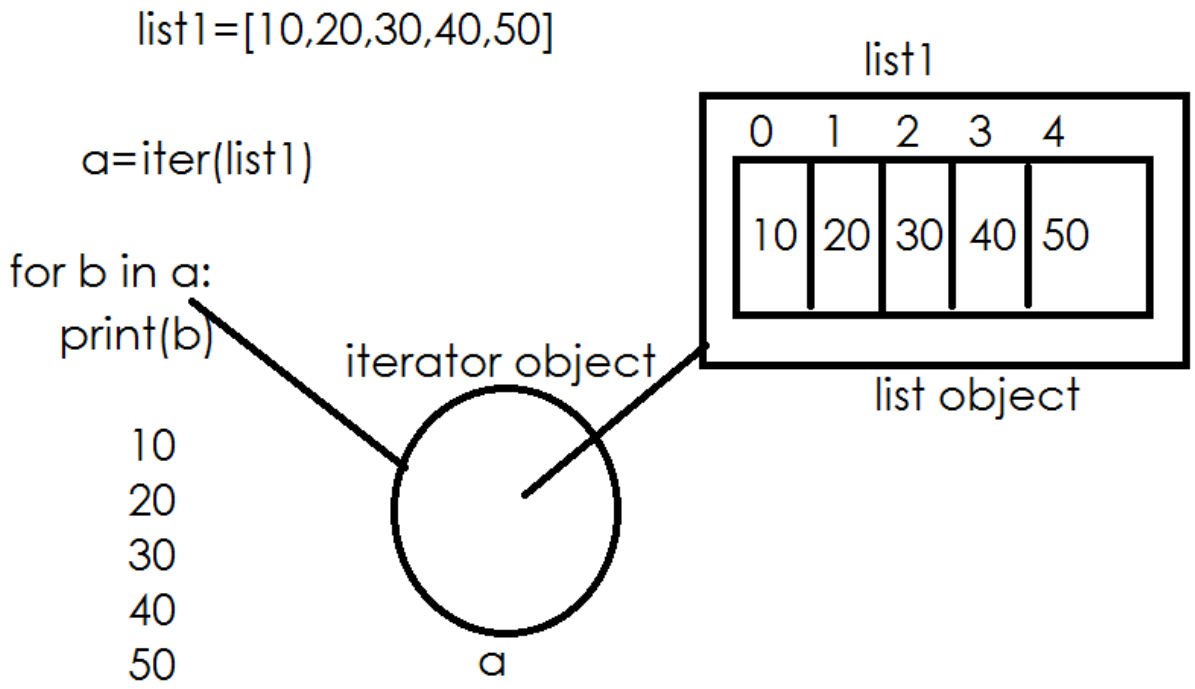
iter() function
It is a predefined function, this function returns iterator object. This object used with for loop to read each time one value from collection/iterable.

Iterator is read only. It is used to read values/elements from collection but it does not allow updating/modifying.



next()
next() is a predefined function, which returns one value generated by iterator.

Example:

```
>>> list1=[10,20,30,40,50]
>>> a=iter(list1)
>>> for value in a:
    print(value)


10
20
30
40
```

```
50
>>> for value in a:
    print(value)


>>> b=iter(list1)
>>> value1=next(b)
>>> print(value1)
10
>>> value2=next(b)
>>> print(value2)
20
>>> value3=next(b)
>>> print(value3)
30
>>> value4=next(b)
>>> print(value4)
40
>>> value5=next(b)
>>> print(value5)
50
>>> value6=next(b)
Traceback (most recent call last):
  File "<pyshell#19>", line 1, in <module>
    value6=next(b)
StopIteration
```

Example:

```
list1=list(range(1,11))
print(list1)
a=iter(list1)
for value in a:
    print(value**2,end=' ')
```

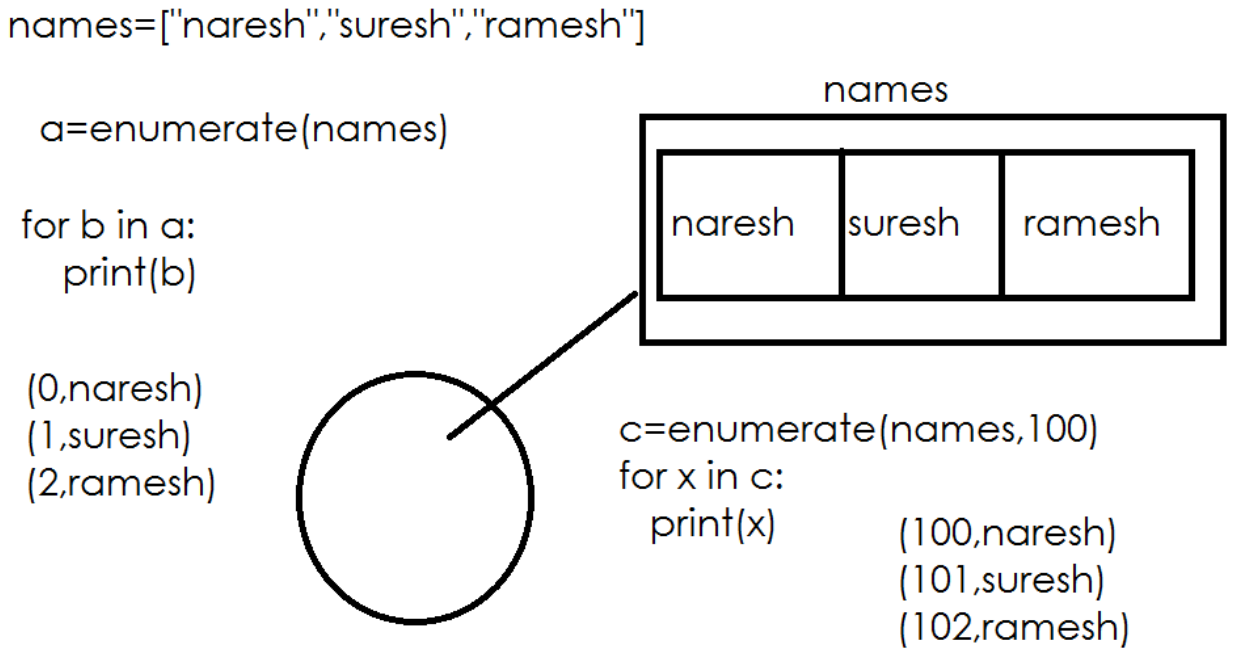
Output:

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
1 4 9 16 25 36 49 64 81 100
```

enumerate(iterable, start=0)

Return an enumerate object. iterable must be a sequence, an iterator, or some other object which supports iteration.

enumerate() returns a tuple containing a count (from start which defaults to 0) and the values obtained from iterating over iterable.



Example:

```
sales=[10000,20000,30000,40000,50000]

e=enumerate(sales,2001)
for a in e:
    print(a)
```

Output:

```
(2001, 10000)
(2002, 20000)
(2003, 30000)
(2004, 40000)
(2005, 50000)
```

Enumerate object is used to represent data in other collections like set and mapping.

Q: What is difference between iterator and enumerate?
Iteator read values from collection or iterable
Enumerate read value and count from collection or iterable

Iter() function returns iterator object
Enumerate() function returns enumerate object

Q: What is difference between iterator and iterable?

Iterator object is used to read content from iterables or collection.
Iterable is an object which represents more than one value. All collections are called iterables. All collections implements iterator protocol.

Mutable Operations of list
List is a mutable collection, after creating list changes can be done. This changes are done using mutable functions/methods provided by list data types/class.

1. **append()**
   This method appends or add element at end of list.

   Example:

```
>>> list1=[]
>>> list1.append(10)
>>> print(list1)
[10]
>>> list1.append(20)
>>> print(list1)
[10, 20]
>>> list1.append(30)
>>> print(list1)
[10, 20, 30]
>>> list1.append(40)
>>> print(list1)
[10, 20, 30, 40]
>>> list1.append(10,20,30)
Traceback (most recent call last):
  File "<pyshell#29>", line 1, in <module>
    list1.append(10,20,30)
TypeError: list.append() takes exactly one argument (3 given)
```

Example:

```
# Write a program to add n players score into list

score=[]
n=int(input("Enter How Many Players?")) # 2
for i in range(n): # start=0,stop=2,step=1 0 1
    s=int(input("Enter Score :")) # 100 50
    score.append(s)


print(f'Players Score {score}')
```

Output:

```
Enter How Many Players?5
Enter Score :0
Enter Score :10
Enter Score :50
Enter Score :20
Enter Score :35
Players Score [0, 10, 50, 20, 35]
```

Example:

```
# Write a program to read/input n integers into list
# and separate even numbers into one list and odd numbers
# into another list

n=int(input("Enter value of n"))
list1=[]
for i in range(n):
    num=int(input("Enter any integer "))
    list1.append(num)

print(f'Numbers List {list1}')
even_list=[]
odd_list=[]
for a in list1:
    if a%2==0:
        even_list.append(a)
    else:
```

```
        odd_list.append(a)

print(f'Even Numbers List {even_list}')
print(f'Odd Numbers List {odd_list}')
```

Output:

```
Enter value of n5
Enter any integer 1
Enter any integer 2
Enter any integer 3
Enter any integer 4
Enter any integer 5
Numbers List [1, 2, 3, 4, 5]
Even Numbers List [2, 4]
Odd Numbers List [1, 3, 5]
```

Example:

```
# Write a program to read name and n subject marks

name=input("Enter Name ")
n=int(input("Enter How Many Subjects ?"))
marks_list=[]
for i in range(n):
    sub=int(input("Enter Marks :"))
    marks_list.append(sub)

print(f'Name {name}')
print(f'Marks {marks_list}')
```

Output:

```
Enter Name naresh
Enter How Many Subjects ?3
Enter Marks :60
Enter Marks :80
Enter Marks :89
Name naresh
Marks [60, 80, 89]
```