What is an Iteration

Iteration is a general term for taking each item of something, one after another. Any time you use a loop, explicit or implicit, to go over a group of items, that is iteration.

```
# example
L=[1,2,3]
for i in L:
    print(i)
```

```
1
2
3
```

What is Iterator

An Iterator is an object that allows the programmer to traverse through a sequence of data without having to store the entire data in the memory

```
# Example
L=[x for x in range(1,10000)]
# for i in L:
    #print(i*2)

import sys
print(sys.getsizeof(L))

x=range(1,100000000)
# for i in x:
    #print(i*2)

print(sys.getsizeof(x))# ye bytes me batayega ki kitna memory use kar raha hai
# you can see same kaam karne ke lye yadi list ka help le rahe hain to 85176 bytes and jab range ke help
# se same kaam kar rahe hain to 48 bytes ka memeory use ho raha hai
```

```
85176
48
```

What is Iterable

Iterable is an object, which one can iterate over

It generates an Iterator when passed to iter() method

- (yadi aap kisi object ke upar loop laga ke uske har item ko fetch kar sakte ho to wah object iterable hota hai jaise ki `range object`, `list` etc)

```
# simply yadi aap kisi ke upar loop laga kar uske har item ko nikal
sakte ho to wo iterable hai

# Example

L=[1,2,3]
print(type(L))

# L is an iterable

print(iter(L))
print(type(iter(L)))


# iter(L)---> iterator

# it prooves ki iterable ke iterator generate karta hai
```

```
<class 'list'>
<list_iterator object at 0x0000017F8A772DA0>
<class 'list_iterator'>
```

Point to remember

- Every Iterator is also and Iterable
- Not all Iterables are Iterators

Trick

- Every Iterable has an iter function
- Every Iterator has both iter function as well as a next function

```
# simply yadi check karna hai ki koi iterable hai ki nahi to loop
chala ke dekh lo

a=2
for i in a:
    print(i)



# clearly you can see ki a (int) object iterable nahi hai
```

```
----------------------------------------------------------------------
-----
TypeError                                 Traceback (most recent call
last)
Cell In[13], line 4
      1 # simply yadi check karna hai ki koi iterable hai ki nahi to
loop chala ke dekh lo
      3 a=2
```

```
----> 4 for i in a:
      5     print(i)

TypeError: 'int' object is not iterable
```

```
L=[1,2,3]
for i in L:
    print(i)


# yaha error nahi aaya means ki List iterable hai
```

```
1
2
3
```

```
# Another method to check ki iterable hai ki nahi

a=2
dir(a)

# dir object ko call karo usse sare magic meethod print ho jayega abb
check karo yadi usme 'iter' magic method hai to wo object
#iterable hai otherwise nahi hai
```

```
['__abs__',
 '__add__',
 '__and__',
 '__bool__',
 '__ceil__',
 '__class__',
 '__delattr__',
 '__dir__',
 '__divmod__',
 '__doc__',
 '__eq__',
 '__float__',
 '__floor__',
 '__floordiv__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getnewargs__',
 '__getstate__',
 '__gt__',
 '__hash__',
 '__index__',
 '__init__',
 '__init_subclass__',
 '__int__',
 '__invert__',
 '__le__',
 '__lshift__',
 '__lt__',
 '__mod__',
 '__mul__',
 '__ne__',
 '__neg__',
 '__new__',
 '__or__',
 '__pos__',
 '__pow__',
 '__radd__',
 '__rand__',
 '__rdivmod__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__rfloordiv__',
 '__rlshift__',
 '__rmod__',
 '__rmul__',
 '__ror__',
 '__round__',
 '__rpow__',
 '__rrshift__',
 '__rshift__',
 '__rsub__',
 '__rtruediv__',
 '__rxor__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__sub__',
 '__subclasshook__',
 '__truediv__',
 '__trunc__',
 '__xor__',
 'as_integer_ratio',
 'bit_count',
 'bit_length',
 'conjugate',
 'denominator',
 'from_bytes',
 'imag',
 'is_integer',
 'numerator',
 'real',
 'to_bytes']
```

```
L=[1,2,3]
dir(L)
```

```
['__add__',
 '__class__',
 '__class_getitem__',
 '__contains__',
 '__delattr__',
 '__delitem__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getitem__',
 '__getstate__',
 '__gt__',
 '__hash__',
 '__iadd__',
 '__imul__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__len__',
 '__lt__',
 '__mul__',
 '__ne__',
 '__new__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__reversed__',
 '__rmul__',
 '__setattr__',
 '__setitem__',
 '__sizeof__',
 '__str__',
 '__subclasshook__',
 'append',
 'clear',
 'copy',
 'count',
 'extend',
 'index',
 'insert',
 'pop',
 'remove',
 'reverse',
 'sort']
```

```python
T=(1,2,3)
dir(a)
```

```
['__abs__',
 '__add__',
 '__and__',
 '__bool__',
 '__ceil__',
 '__class__',
 '__delattr__',
 '__dir__',
 '__divmod__',
 '__doc__',
 '__eq__',
 '__float__',
 '__floor__',
 '__floordiv__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getnewargs__',
 '__getstate__',
 '__gt__',
 '__hash__',
 '__index__',
 '__init__',
 '__init_subclass__',
 '__int__',
 '__invert__',
 '__le__',
 '__lshift__',
 '__lt__',
 '__mod__',
 '__mul__',
 '__ne__',
 '__neg__',
 '__new__',
 '__or__',
 '__pos__',
 '__pow__',
 '__radd__',
 '__rand__',
 '__rdivmod__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__rfloordiv__',
 '__rlshift__',
 '__rmod__',
 '__rmul__',
 '__ror__',
 '__round__',
 '__rpow__',
 '__rrshift__',
 '__rshift__',
 '__rsub__',
 '__rtruediv__',
 '__rxor__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__sub__',
 '__subclasshook__',
 '__truediv__',
 '__trunc__',
 '__xor__',
 'as_integer_ratio',
 'bit_count',
 'bit_length',
 'conjugate',
 'denominator',
 'from_bytes',
 'imag',
 'is_integer',
 'numerator',
 'real',
 'to_bytes']

# now trick to check ki koi object iterator hai ki nahi

#--> dir method ko call karo and check ki yadi uske anddar 'iter and
next' dono magic method available hai to wo iterator hai

# Example

L=[1,2,3]

#L is not an iterator

dir(a)


# you can see iske andar iter to hai hai but next nahi hai means ye ek
iterator nahi hai

['__abs__',
 '__add__',
 '__and__',
 '__bool__',
 '__ceil__',
 '__class__',
 '__delattr__',
 '__dir__',
 '__divmod__',
 '__doc__',
 '__eq__',
 '__float__',
 '__floor__',
 '__floordiv__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getnewargs__',
 '__getstate__',
 '__gt__',
 '__hash__',
 '__index__',
 '__init__',
 '__init_subclass__',
 '__int__',
 '__invert__',
 '__le__',
 '__lshift__',
 '__lt__',
 '__mod__',
 '__mul__',
 '__ne__',
 '__neg__',
 '__new__',
 '__or__',
 '__pos__',
 '__pow__',
 '__radd__',
 '__rand__',
 '__rdivmod__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__rfloordiv__',
 '__rlshift__',
 '__rmod__',
 '__rmul__',
 '__ror__',
 '__round__',
 '__rpow__',
 '__rrshift__',
 '__rshift__',
 '__rsub__',
 '__rtruediv__',
 '__rxor__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__sub__',
 '__subclasshook__',
 '__truediv__',
 '__trunc__',
 '__xor__',
 'as_integer_ratio',
 'bit_count',
 'bit_length',
 'conjugate',
 'denominator',
 'from_bytes',
 'imag',
 'is_integer',
 'numerator',
 'real',
 'to_bytes']
```

```python
L=[1,2,3]

# L is not an iterator

iter_L=iter(L)
dir(iter_L)

# iter_L is an iterator---> qki iske andar "iter and next " dono hai
```

```
['__class__',
 '__delattr__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__getattribute__',
 '__getstate__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__iter__',
 '__le__',
 '__length_hint__',
 '__lt__',
 '__ne__',
 '__new__',
 '__next__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__setattr__',
 '__setstate__',
 '__sizeof__',
 '__str__',
 '__subclasshook__']
```

Understanding how for loop works

```
num = [1,2,3]

for i in num:
    print(i)
```

```
1
2
3
```

```
num=[19,32,63]
## step-1--> `` fetch the iterator

# sabse pahle iterable se iterator ko fetch karta hai

iter_num=iter(num)

## step-2 --> next
# uske baad next ko call karta hai

print(next(iter_num))
print(next(iter_num))
print(next(iter_num))
print(next(iter_num))   # iss line se error aayega qki abb koi element hai hi nahi
```

```
19
32
63
```

```
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Cell In[36], line 14
     12 print(next(iter_num))
     13 print(next(iter_num))
```

```
---> 14 print(next(iter_num))

StopIteration:
```

Making ouw own for loop

```
def mera_khudka_for_loop(iterable):

    iterator=iter(iterable)
    while True:

        try:
            print(next(iterator))
        except StopIteration:
            break

a=[1,2,3]
b=range(1,11)
c=(1,2,3)
d={1,2,3}
e={0:1,1:1}

mera_khudka_for_loop(e)

0
1

mera_khudka_for_loop(c)

1
2
3

mera_khudka_for_loop(a)

1
2
3

mera_khudka_for_loop(b)

1
2
3
4
5
6
7
8
9
10
```

```
mera_khudka_for_loop(d)

1
2
3
```

A confusing point

```
num=[1,2,3]
iter_obj=iter(num)

print(id(iter_obj),'Address of iterator 1')

iter_obj2 = iter(iter_obj)
print(id(iter_obj2),'Address of iterator 2')


1765174461728 Address of iterator 1
1765174461728 Address of iterator 2

# dono ka address same hai matlab ki iterator ke upat iter chalane se
jo iterator milta hai wo wahi khud hota hai
```

Let's create our own range() function

```
class mera_range:
    def __init__ (self,start,end):
        self.start=start
        self.end=end

    def __iter__(self):
        return mera_range_iterator(self)


class mera_range_iterator:
    def __init__(self,iterable_obj):
        self.iterable=iterable_obj

    def __iter__(self):
        return self

    def __next__(self):

        if self.iterable.start >= self.iterable.end:
            raise StopIteration

        current=self.iterable.start
        self.iterable.start+=1
        return current
```

```
for i in mera_range(1,11):
    print(i)
```

```
1
2
3
4
5
6
7
8
9
10
```

suppose aapke pass bahut bada image ka data hai hai suppose 20 gb ka but aapke passs memory hai 8 gb ka hi to aap kaise data ko memory me load karoge to yaha pe iterator ka help le sakte ho wo one by one image ko pakdega and memory me storre karte jayega