# Assignment-1

Name:- U. Srilekha

AICTE id:- STU654618162c215169909.32 70

1) Enumerate:-

In simple words, the enumerate function takes as input, an iterable and adds a counter to each iterable element. and returns an enumerate object. The counter can also act as indices to each element which can be used to reference these elements at a later stage when required.

Enumerate() is a built in function in Python. This function allows us to loop over something and have an automatic counter. And this function works by adding a counter to an iterable and returning it in the form of an enumerate object. The enumerate object can be used directly for loops or converted into a tuple list of tuples using the list() Method.

Syntax:- enumerate (iterable, start_index).

Ex:- Let's Consider a simple where loops are used to Print the index along with the items of a list.

Program:-

```
Letters = ["a", "b", "c", "d", "e", "f", "g"]
index=0
for letter in letters:
    Print (index, (letter)
    index += 1.
```

(vi)-(b) convert a tuple into an enumerate object.

```
x = ('apple', 'banana', 'cherry')
y = enumerate(x)
```

2. Reduce ()

The Reduce() function is used to apply a Particular function passed in its argument to all of the list elements mentioned in the sequence passed along. This function is defined in "functools" Module.

The reduce() function is defined in the functools module. This function receives two arguments, a function and an iterable.

Syntax:-

```
functools.reduce(my-functions, iterable, initializer)
```

Example:-

```
import functools
def mult(x, y):
    Print("x=", x, "y=", y)
    return x*y
fact = functools.reduce(mult, range(1,4))
Print("Factorial of 3:", fact)
```

Output:-

```
x=1 y=2
x=2 y=3
factorial of 3: 6
```

3. Map():-

Python map() applies a function on all the items of an iterator given as input. An iterator, for example, can be a list, a tuple, a set, a dictionary, a string and it returns an iterable map object. Python map() is a built-in function

Syntax:- map (function, iterator1, iterator2 .... iteratorN)

Example:-

```
def square(n):
    return n*n
my_list = [2, 3, 4, 5, 6, 7, 8, 9]
updated_list = map(square, my_list)
Print (updated_list)
Print (list (updated_list))
```

Output: <map object at 0x0000002c59601748>
[4, 9, 16, 25, 36, 49, 64, 81].

4. filter()

filter() is a built in function in Python. The filter function can be applied to an iterable such as a list or a dictionary and create a new iterator. This new iterator can filter out certain specific elements based on the condition that you provide very efficiently.

Syntax:- filter (function, iterable)

Example:

```
def check(letter):
    list_of_vowels = ['a', 'e', 'i', 'o', 'u']
    if letter in list_of_vowels:
        return True
    else:
        return False
letters = ['u', 'a', 'q', 'c', 'i', 'd', 'z', 'p', 'e']
filtered_object = filter(check, letters)
Print("The type of returned object is:", type(filtered_object))
filtered_list = list(filtered_object)
Print("The list of vowels is:", filtered_list).
```

Output:-

The type of returned object is: <class 'filter'>

The list of vowel is: ['u', 'a', 'i', 'e']

5. zip() id()

In Python, id() function is a built-in function that returns the unique identifier of an object. The identifier is an integer, which represents the memory address of the object. The id() function is commonly used to check if two variables or objects refer to the same memory location.

Syntax:- id (object)

Example:-

```
x = 42
y = x
z = 42
Print (id(x))
Print (id(y))
Print (id(z))
```

Output:-

```
140642115230496
140642115230496
140642115230496.
```

6) zip ( ) :-

Python zip() function returns a zip object, which maps a similar index of multiple containers. It takes iterable, makes it an iterator that aggregates the elements based on iterables passed, and returns an iterator of tuple.

It is a built-in-function that is used to combine two or more iterables into a single iterable.

Syntax:- zip(* iterables)

Example:-

```
Co-ordinate = ['x', 'y', 'z']
Value = [6, 2, 8, 5, 1].
result = zip(coordinate, value)
result list = list(result)
Print (resultlist)
```

```
C,V = zip(* resultlist)
Print('c=', C)
Print ('v=', v)
```

output:-

```
[('x',6), ('y', 2), ('z', 8)]
c= ('x', 'y', 'z')
v= (6, 2, 8).
```