Day – 6 Morning Assessment

Magic Methods

1. \_\_init\_\_() magic method is used to initialize the object’s parameter in the python class.
2. \_\_str\_\_() returns normal string where as \_\_repr()\_\_ returns official string which is used in advance purposes/processes like debugging etc..
3. Eg:

class Example:

def \_\_init\_\_(self, a,b):

self.a = a

self.b = b

def \_\_add\_\_(self):

return {self.a+self.b+c}

ex1 = Example(2,7)



Itertools

1. itertools.product() is like importing the product function from the itertools library.

Eg:

From itertools import product

A,B = 4,8

Print(prod(A,B))

1. itertools.permutations() is importing permutations function or method from itertools library and it is used to get the permutations for the given set of data. Whereas itertools.combinations() is importing combinations method from itertools library and it is used to get the combinations of the given set of data.
2. Itertools.chain() is used to take multiple iterators and gives a single return

of all the values one by one.

1. from itertools import cycle

count = 0

for i in cycle([‘A’,’B’,’C’,’D’]):

print(i)

count = count+1

if count >= 6:

break

1. map() is used when we have to take multiple input values into a single variable. It has two arguments, one is function and the other is iterable. And it returns an object.
2. Lis1 = [1,2,3,4]

Lis2 = [5,6,7,8]

Res = list(map(lambda x,y : x+y, Lis1,Lis2))

Print(Res)

1. Map() is used to modify the elements using function where as filter() filters the elements according to the given condition.
2. Yes, map can work with lambda functions

Eg:

Lis1 = [1,2,3,4]

Lis2 = [5,6,7,8]

Res = list(map(lambda x,y : x\*y, Lis1,Lis2))

Print(Res)

Generators

1. Generator function is used to yield one value at a time. It is used when we need to yield values one by one instead of printing all the values at a time.
2. Yield generates the value and yields only one at a time whereas returns returns the total result from the function once at a time. And also yields pauses the execution whereas return ends the execution of that particular function.
3. def even\_numbers():

for i in range(1,11):

if i %2 ==0:

yield i

for num in even\_numbers():

print(num)

1. So, if the next() gets exhausted, then it raises a StopIteration exception.

Iterators

1. Iterator is the object which iterates the iterable, where iterable is the data type which has multiple data stored in a single variable. Eg: for loop, while loop are iterators where as list, tuple, etc are iterables.
2. \_\_iter\_\_() and \_\_next\_\_() are two magic methods for a class to be an iterator.
3. def \_\_init\_\_(self):

self.num = 1

def\_\_iter\_\_(self):

return self

def\_\_next\_\_(self):

if self.num >5:

raise StopIteration

else:

print(self.num)

self.num +=1

1. iter() allows the traversal of the list with the help of next().