Assignment 03.

1.1 Write a Python Program to implement your own myreduce() function which works exactly like Python's built-in function reduce()

Code:

```
def myreduce(anyfunc, sequence):

 # Get first item in sequence and assign to result
  result = sequence[0]
 # iterate over remaining items in sequence and apply
reduction function
  for item in sequence[1:]:
   result = anyfunc(result, item)

  return result
def sum(x,y): return x + y

print ("Sum on list [1,2,3] using custom reduce
function "   + str(myreduce(sum, [1,2,3])) )
```

Output:

```
def myreduce(anyfunc, sequence):

 # Get first item in sequence and assign to result
  result = sequence[0]
 # iterate over remaining items in sequence and apply reduction function
  for item in sequence[1:]:
   result = anyfunc(result, item)

  return result
def sum(x,y): return x + y

print ("Sum on list [1,2,3] using custom reduce function "   + str(myreduce(sum, [1,2,3])) )
```

```
Sum on list [1,2,3] using custom reduce function 6
```

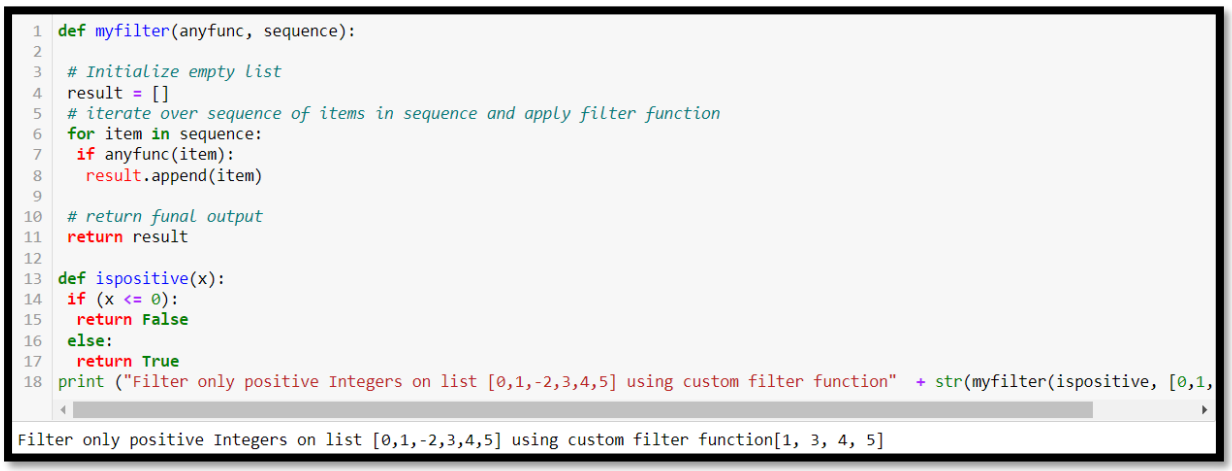
1.2 Write a Python program to implement your own myfilter() function which works exactly like Python's built-in function filter()

Code:

```
def myfilter(anyfunc, sequence):

 # Initialize empty list
 result = []
 # iterate over sequence of items in sequence and apply filter
function
 for item in sequence:
  if anyfunc(item):
```

```
   result.append(item)

 # return funal output
 return result

def ispositive(x):
 if (x <= 0):
  return False
 else:
  return True
print ("Filter only positive Integers on list [0,1,-2,3,4,5] using
custom filter function"  + str(myfilter(ispositive, [0,1,-
2,3,4,5])))
```

Output:



2. Implement List comprehensions to produce the following lists.
Write List comprehensions to produce the following Lists
['A', 'C', 'A', 'D', 'G', 'I', 'L', ' D']
['x', 'xx', 'xxx', 'xxxx', 'y', 'yy', 'yyy', 'yyyy', 'z', 'zz', 'zzz', 'zzzz']
['x', 'y', 'z', 'xx', 'yy', 'zz', 'xxx', 'yyy', 'zzz', 'xxxx', 'yyyy', 'zzzz']
[[2], [3], [4], [3], [4], [5], [4], [5], [6]] [[2, 3, 4, 5], [3, 4, 5, 6],
[4, 5, 6, 7], [5, 6, 7, 8]]
[(1, 1), (2, 1), (3, 1), (1, 2), (2, 2), (3, 2), (1, 3), (2, 3), (3, 3)]

Code:

- ```
  word = "ACADGILD"
  alphabet_list = [ i for i in word ]
  print ( str(alphabet_list))
  ```

- ```
  list_a = "xyz"
  result =[ i*num for i in list_a for num in range(1,5)]
  print ( str(result))
  ```

- ```
  list_b = "xyz"
  result = [ i*num for num in range(1,5) for i in list_b  ]
  print(result)
  ```

Output:

```
word = "ACADGILD"
alphabet_list = [ i for i in word ]
print ( str(alphabet_list))
```

```
['A', 'C', 'A', 'D', 'G', 'I', 'L', 'D']
```

```
list_a = "xyz"
result =[ i*num for i in list_a for num in range(1,5)]
print ( str(result))
```

```
['x', 'xx', 'xxx', 'xxxx', 'y', 'yy', 'yyy', 'yyyy', 'z', 'zz', 'zzz', 'zzzz']
```

```
list_b = "xyz"
result = [ i*num for num in range(1,5) for i in list_b  ]
print(result)
```

```
['x', 'y', 'z', 'xx', 'yy', 'zz', 'xxx', 'yyy', 'zzz', 'xxxx', 'yyyy', 'zzzz']
```

- ```
  input_lista = [2,3,4]
  input_listb = [2,3,4,5]
  result =  [[item+num] for item in input_lista for num in
  range(0,3)]
  value  =  [[item+num for item in input_listb] for num in
  range(0,2)]
  print(str(result) ,str(value))
  ```

- ```
  list_c = [2,3,4,5]
  result = [ [item+num for item in list_c] for num in range(2,4)
  ]
  print(str(result))
  ```

- ```
  list_d=[1,2,3]
  result = [ (b,a) for a in list_d for b in list_d]
  print(str(result))
  ```

Output:

```
input_lista = [2,3,4]
input_listb = [2,3,4,5]
result =  [[item+num] for item in input_lista for num in range(0,3)]
value  =  [[item+num for item in input_listb] for num in range(0,2)]
print(str(result) ,str(value))

```

```
[[2], [3], [4], [3], [4], [5], [4], [5], [6]] [[2, 3, 4, 5], [3, 4, 5, 6]]
```

```
list_c = [2,3,4,5]
result = [ [item+num for item in list_c] for num in range(2,4)  ]
print(str(result))
```

```
[[4, 5, 6, 7], [5, 6, 7, 8]]
```

```
list_d=[1,2,3]
result = [ (b,a) for a in list_d for b in list_d]
print(str(result))
```

```
[(1, 1), (2, 1), (3, 1), (1, 2), (2, 2), (3, 2), (1, 3), (2, 3), (3, 3)]
```