##TASK 1##

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

In [12]:
```
def sum(x,y):
    return x+y

def my_reduce(a,b):
    count=b[0]
    for x in b[1:]:
        count=a(count,x)
    return count

print(my_reduce(sum, [11,22,33,44,55]))
```

```
165
```

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

In [13]:

```python
ages = [5,8,12,15,17,21,18,45,24,53,32]

def my_age(x):
    if x < 18:
        return False
    else:
        return True

adults = filter(my_age, ages)

for x in adults:
    print(x)
```

```
21
18
45
24
53
32
```

(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', 'xx', 'yy', 'zz', 'xxxx', 'yyyy', 'zzzz']

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

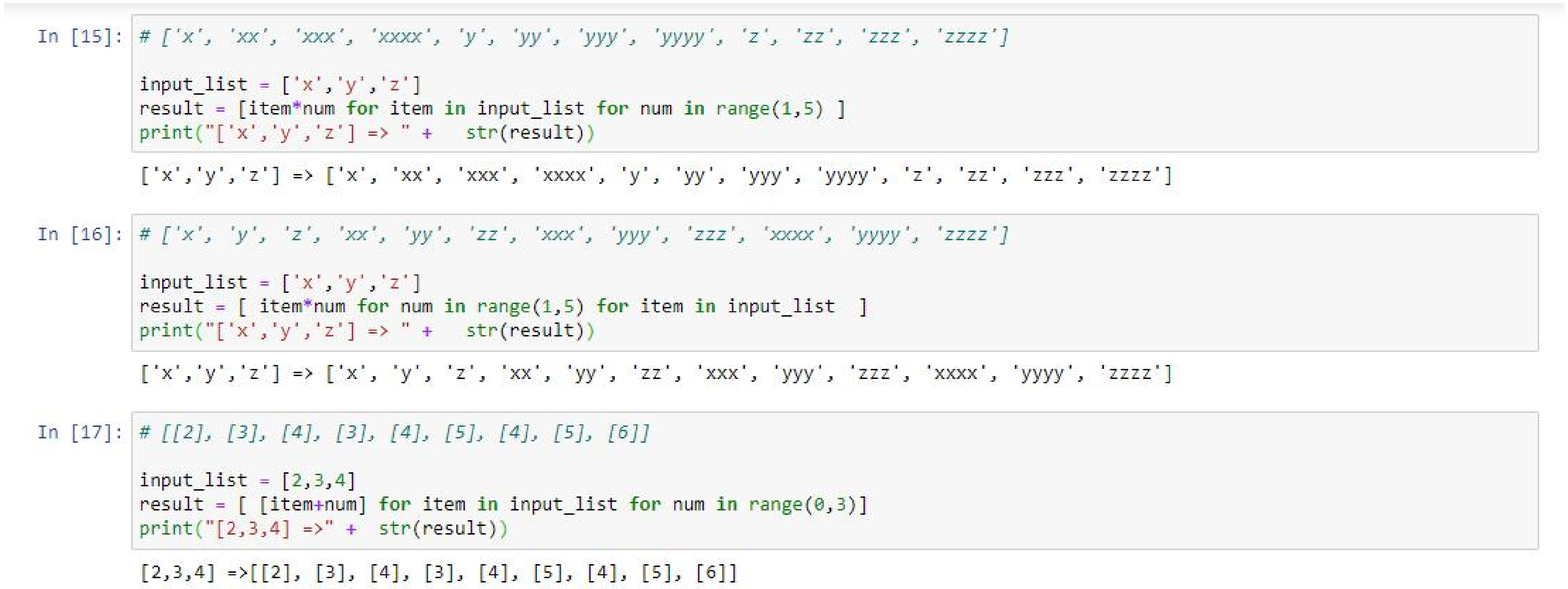
[[2, 3, 4, 5], [3, 4, 5, 6], [4, 5, 6, 7], [5, 6, 7, 8]]

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

In [14]:
```
# Implement List comprehensions to produce the following lists.
# Write List comprehensions to produce the following Lists
# ['A', 'C', 'A', 'D', 'G', 'I', ’L’, ‘ D’]

word = "ACADGILD"
alphabet_list = [ alphabet for alphabet in word ]
print ("ACADGILD => " + str(alphabet_list))
```

```
ACADGILD => ['A', 'C', 'A', 'D', 'G', 'I', 'L', 'D']
```



```
In [15]: # ['x', 'xx', 'xxx', 'xxxx', 'y', 'yy', 'yyy', 'yyyy', 'z', 'zz', 'zzz', 'zzzz']

         input_list = ['x','y','z']
         result = [item*num for item in input_list for num in range(1,5) ]
         print("['x','y','z'] => " +   str(result))

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

```
In [16]: # ['x', 'y', 'z', 'xx', 'yy', 'zz', 'xxx', 'yyy', 'zzz', 'xxxx', 'yyyy', 'zzzz']

         input_list = ['x','y','z']
         result = [ item*num for num in range(1,5) for item in input_list  ]
         print("['x','y','z'] => " +   str(result))

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

```
In [17]: # [[2], [3], [4], [3], [4], [5], [4], [5], [6]]

         input_list = [2,3,4]
         result = [ [item+num] for item in input_list for num in range(0,3)]
         print("[2,3,4] =>" +  str(result))

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

In [18]:
```
# [[2, 3, 4, 5], [3, 4, 5, 6], [4, 5, 6, 7], [5, 6, 7, 8]]

input_list = [2,3,4,5]
result = [ [item+num for item in input_list] for num in range(0,4)  ]
print("[2,3,4,5] =>" +  str(result))
```

```
[2,3,4,5] =>[[2, 3, 4, 5], [3, 4, 5, 6], [4, 5, 6, 7], [5, 6, 7, 8]]
```

In [19]:
```
# # [(1, 1), (2, 1), (3, 1), (1, 2), (2, 2), (3, 2), (1, 3), (2, 3), (3, 3)]

input_list=[1,2,3]
result = [ (b,a) for a in input_list for b in input_list]
print("[1,2,3] =>" +  str(result))
```

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

(3)Implement a function longestWord() that takes a list of words and returns the longest one.

In [21]:
```python
# Implement a function longestWord() that takes a list of words and returns the longest one.

def longestWord(words):
    word_len = []
    for n in words:
        word_len.append((len(n), n))
    word_len.sort()
    return word_len[-1][1]

print(longestWord(['Delhi', 'Maharashtra', 'goa', 'MP', 'UP']))
```

```
Maharashtra
```

##TASK 2###

(1.1) Write a Python Program(with class concepts) to find the area of the triangle using the below formula.

area = (s(s-a)(s-b)(s-c)) * 0.5

Function to take the length of the sides of triangle from user should be defined in the parent class and function to calculate the area should be defined in subclass.

In [23]:
```python
class Triangle:

    def __init__(self,a,b,c):
        self.a = float(a)
        self.b = float(b)
        self.c = float(c)

    def area(self):
        s=(self.a + self.b + self.c)/2
        return((s*(s-self.a)*(s-self.b)*(s-self.c))**0.5)

a =input("Enter the value of a = ")
b =input("Enter the value of b = ")
c =input("Enter the value of c = ")
t = Triangle(a, b, c)

print("area : {}".format(t.area()))
```

(1.2) Write a function filter_long_words() that takes a list of words and an integer n and returns the list of words that are longer than n.

In [27]:
```
def filter_long_words(guess, number):

    new_list = []

    for i in range(len(guess)):
        if len(guess[i]) > number:
            new_list.append(guess[i])

    print (new_list)

list1 = input("Enter list of words: ")
list2 = list1.split(",")

def number():
    global list, integer1
    integer = input("Enter the value of n : ")
    integer1 = int(integer)
    filter_long_words(list2, integer1)

number()
```

```
Enter list of words: hello,my,name,is,Shubham
Enter the value of n : 3
['hello', 'name', 'Shubham']
```

(2.1) Write a Python program using function concept that maps list of words into a list of integers representing the lengths of the corresponding words.

Hint: If a list [ab,cde,erty] is passed on to the python function output should come as [2,3,4]

Here 2,3 and 4 are the lengths of the words in the list.

In [28]:
```
listOfWords =  ['Apple','Banana','Cranberry','Jackfruit','Watermelon']

listOfInts = []

for i in range(len(listOfWords)):
    listOfInts.append(len(listOfWords[i]))

print ("List of words:"+str(listOfWords))
print ("List of wordlength:"+str(listOfInts))
```

```
List of words:['Apple', 'Banana', 'Cranberry', 'Jackfruit', 'Watermelon']
List of wordlength:[5, 6, 9, 9, 10]
```

(2.2) Write a Python function which takes a character (i.e. a string of length 1) and returns True if it is a vowel, False otherwise.

In [29]:

```python
def my_vowel(char):
    vowels = ('a', 'e', 'i', 'o', 'u')
    if char not in vowels:
        return False
    return True


if __name__ == "__main__":
    print (my_vowel('a'))
    print (my_vowel('b'))
```

```
True
False
```