# Python: Exercise on Lists and Strings

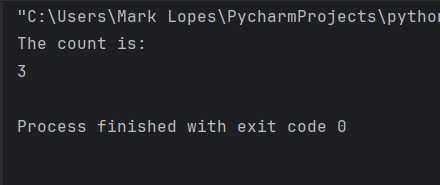
**Name: Mark Lopes**

**Roll No: 9913**

**S.E Comps\_A Batch C**

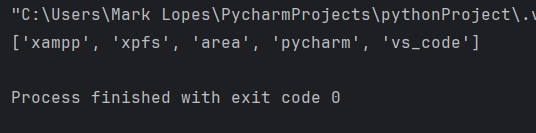
Q1. Given a list of strings, return the count of the number of strings where the string length is 2 or more and the first and last chars of the string are the same.

def count\_string(list):  
 count = 0  
 for string in list :  
 if len(string)>=2 and string[0]==string[-1]:  
 count += 1  
 return count  
  
  
list1=["hello","mam","u","yuy","hf fh"]  
final\_count=count\_string(list1)  
print("The count is:")  
print(final\_count)



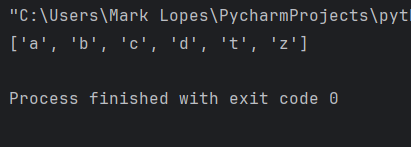
Q2. Given a list of strings, return a list with the strings in sorted order, except group all the strings that begin with &#39;x&#39; first.

def string\_sort(strings\_list):  
 # Initialize empty lists  
 starts\_with\_x = []  
 does\_not\_start\_with\_x = []  
  
 # Separate the strings into two groups  
 for string in strings\_list:  
 if string and string[0] == 'x':  
 starts\_with\_x.append(string)  
 else:  
 does\_not\_start\_with\_x.append(string)  
  
 # Sort both groups  
 starts\_with\_x.sort()  
 does\_not\_start\_with\_x.sort()  
  
 # Combine the sorted lists  
 result = starts\_with\_x + does\_not\_start\_with\_x  
  
 return result  
  
  
list=["xampp","pycharm","vs\_code","xpfs","area"]  
sorted\_list = string\_sort(list)  
print(sorted\_list)



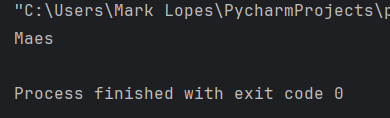
Q3. Given two lists sorted in increasing order, create and return a merged list of all the elements in sorted order. You may modify the passed in lists. Ideally, the solution should work in &quot;linear&quot; time, making a single pass of both lists.

def sort\_both(s1,s2):  
 final\_sort=s1+s2  
 final\_sort=sorted(final\_sort)  
 return final\_sort  
  
  
list1 = ["d","t","c"]  
list2 = ["z","b","a"]  
  
print(sort\_both(list1,list2))



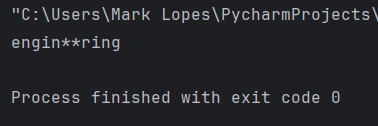
Q4. Given a string s, return a string made of the first 2 and the last 2 chars of the original string, so &#39;spring&#39; yields &#39;spng&#39;. However, if the string length is less than 2, return instead the empty string.

def q4(s):  
 if len(s)<2:  
 return ""  
 string = s[:2] + s[-2:]  
 return string  
  
  
test\_string="Mark Lopes"  
print(q4(test\_string))



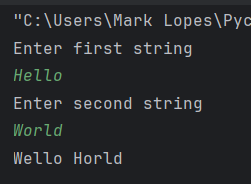
Q5. Given a string s, return a string where all occurences of its first char have been changed to &#39;\*&#39;, except do not change the first char itself.

test\_string = "engineering"  
print(test\_string[0]+test\_string[1:].replace(test\_string[0],'\*'))



Q6. Given strings a and b, return a single string with a and b separated by a space &#39;&lt;a&gt; &lt;b&gt;&#39;, except swap the first 2 chars of each string.

s1 = input("Enter first string\n")  
s2 = input("Enter second string\n")  
  
print(s2[0]+s1[1:]+" "+s1[0]+s2[1:])



Q7. Given a string, find the first appearance of the substring &#39;not&#39; and &#39;bad&#39;. If the &#39;bad&#39; follows the &#39;not&#39; replace the whole &#39;not&#39;...&#39;bad&#39; substring with &#39;good&#39;. Return the resulting string.

def check\_string(s):  
 if (s.find('not')!=-1 and s.find('bad')!=-1):  
 print(s[:s.find('not')]+"good")  
  
  
test\_string=input("Enter first string: ")  
check\_string(test\_string)

