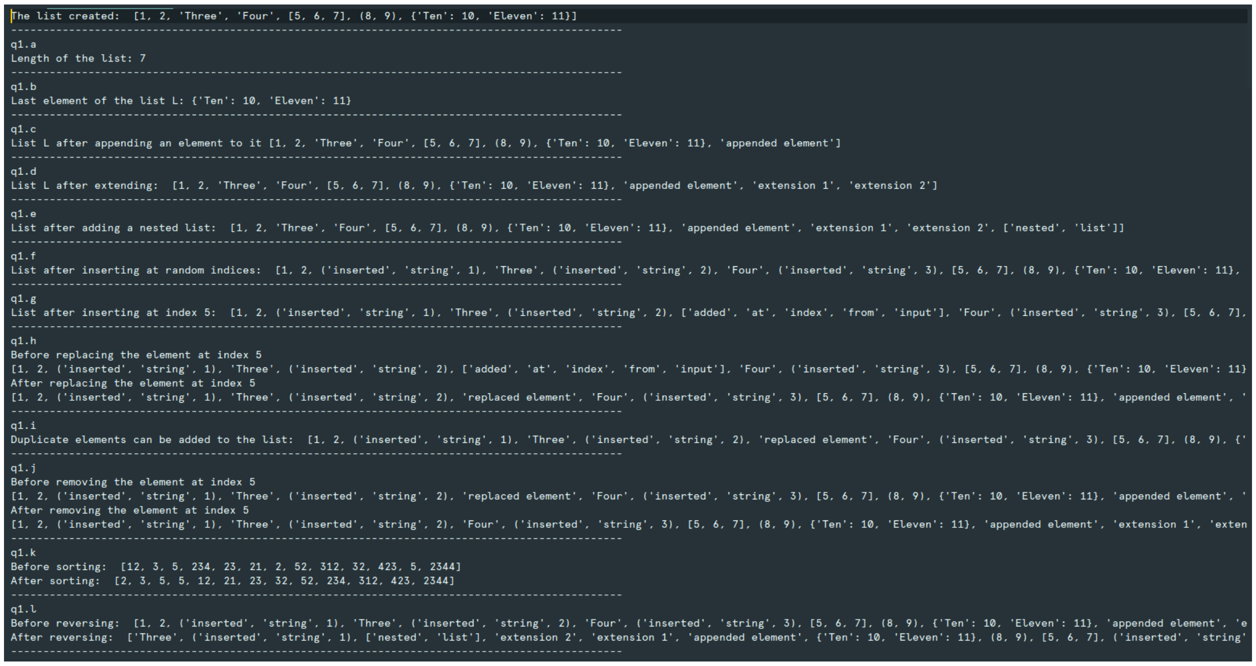
Data Science - Lab 2

Student Name	Venkata Sai Manoj Boganadham
Roll no	197121
Section	A
Type	Assignment
Subject	DS Lab

1. Write a python program to create a list with n number of items (where n should be at least 6) with different types (integer, float, string) and perform the following functions: (List Methods)

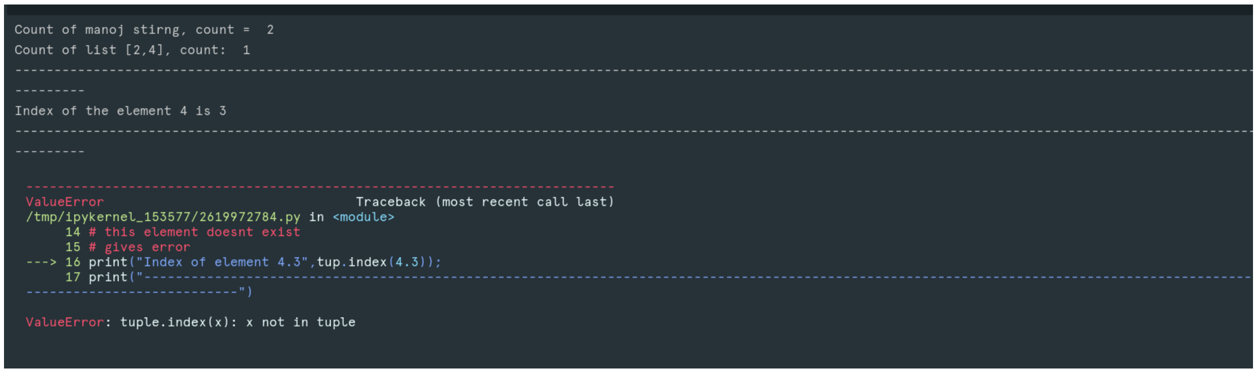
```
# q1
# List with 10 items
L = [1, 2, 'Three', 'Four', [5,6,7], (8,9), {'Ten': 10, 'Eleven': 11}];
print("The list created: ", L);
print("-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------")

# q1.a - Get the length of the list L
print("q1.a");
print("Length of the list:", len(L))
print("-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------")


# q1.b - Access the last element of the list using the negative index
print("q1.b");
print("Last element of the list L:", L[-1]);
print("-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------")


# q1.c - Add one item to a list using the append() method
print("q1.c");

L.append('appended element');
print("List L after appending an element to it", L);
print("-------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
------")


# q1.d - Add several items using the extend method
print("q1.d");

L.extend(["extension 1", "extension 2"]);
print("List L after extending: ", L);
print("-------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
------")


# q1.e - Add a list as an item to the existing list (nested list)
print("q1.e");

L.append(["nested", "list"]);
print("List after adding a nested list: ", L);
print("-------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
------")


# q1.f - Use the index operator to access the items at various location within the list
# adding at the indices 2, 4 ,6
print("q1.f");

L.insert(2, ("inserted", "string", 1));
L.insert(4, ("inserted", "string", 2));
L.insert(6, ("inserted", "string", 3));
print("List after inserting at random indices: ", L);
print("-------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
------")


# q1.g - Add an element to the list at specified index using the insert() method
# give the input 5
print("q1.g");

idx = int(input("Enter the integer 5:"));
L.insert(idx, ["added","at","index","from","input"]);
print("List after inserting at index 5: ", L);
print("-------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
------")


# q1.h - Replace an existing element from the list at a specified location
# replacing the element at the index 5
print("q1.h");
print("Before replacing the element at index 5");
print(L);
L[5] = "replaced element";
print("After replacing the element at index 5");
print(L);
print("--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------")


# q1.i - Add duplicate elements to the list
print("q1.i");
L.append(L[2]);
L.append(L[3]);
print("Duplicate elements can be added to the list: ", L);
print("--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------")


# q1.j - Remove the item at the given index from the list using pop() method
# Removing the element at the index 5
print("q1.j");

print("Before removing the element at index 5");
print(L);
L.pop(5);
print("After removing the element at index 5");
print(L);
print("--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------")


# q1.k - Sort the elements of the given list in a specific ascending or descending order
print("q1.k");
L1 = [12,3,5,234,23,21,2,52,312,32,423,5,2344];
print("Before sorting: ", L1);
L1.sort();
print("After sorting: ", L1);
print("--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------")


# q1.l - Reverse the elements using the reverse method
print("q1.l");
print("Before reversing: ", L);
L.reverse();
print("After reversing: ", L);
print("--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------")
```



```
The list created:  [1, 2, 'Three', 'Four', [5, 6, 7], (8, 9), {'Ten': 10, 'Eleven': 11}]
-----------------------------------------------------------------------------------------------
q1.a
Length of the list: 7
-----------------------------------------------------------------------------------------------
q1.b
Last element of the list L: {'Ten': 10, 'Eleven': 11}
-----------------------------------------------------------------------------------------------
q1.c
List L after appending an element to it [1, 2, 'Three', 'Four', [5, 6, 7], (8, 9), {'Ten': 10, 'Eleven': 11}, 'appended element']
-----------------------------------------------------------------------------------------------
q1.d
List L after extending:  [1, 2, 'Three', 'Four', [5, 6, 7], (8, 9), {'Ten': 10, 'Eleven': 11}, 'appended element', 'extension 1', 'extension 2']
-----------------------------------------------------------------------------------------------
q1.e
List after adding a nested list:  [1, 2, 'Three', 'Four', [5, 6, 7], (8, 9), {'Ten': 10, 'Eleven': 11}, 'appended element', 'extension 1', 'extension 2', ['nested', 'list']]
-----------------------------------------------------------------------------------------------
q1.f
List after inserting at random indices:  [1, 2, ('inserted', 'string', 1), 'Three', ('inserted', 'string', 2), 'Four', ('inserted', 'string', 3), [5, 6, 7], (8, 9), {'Ten': 10, 'Eleven': 11},
-----------------------------------------------------------------------------------------------
q1.g
List after inserting at index 5:  [1, 2, ('inserted', 'string', 1), 'Three', ('inserted', 'string', 2), ['added', 'at', 'index', 'from', 'input'], 'Four', ('inserted', 'string', 3), [5, 6, 7],
-----------------------------------------------------------------------------------------------
q1.h
Before replacing the element at index 5
[1, 2, ('inserted', 'string', 1), 'Three', ('inserted', 'string', 2), ['added', 'at', 'index', 'from', 'input'], 'Four', ('inserted', 'string', 3), [5, 6, 7], (8, 9), {'Ten': 10, 'Eleven': 11}
After replacing the element at index 5
[1, 2, ('inserted', 'string', 1), 'Three', ('inserted', 'string', 2), 'replaced element', 'Four', ('inserted', 'string', 3), [5, 6, 7], (8, 9), {'Ten': 10, 'Eleven': 11}, 'appended element', '
-----------------------------------------------------------------------------------------------
q1.i
Duplicate elements can be added to the list:  [1, 2, ('inserted', 'string', 1), 'Three', ('inserted', 'string', 2), 'replaced element', 'Four', ('inserted', 'string', 3), [5, 6, 7], (8, 9), {'
-----------------------------------------------------------------------------------------------
q1.j
Before removing the element at index 5
[1, 2, ('inserted', 'string', 1), 'Three', ('inserted', 'string', 2), 'replaced element', 'Four', ('inserted', 'string', 3), [5, 6, 7], (8, 9), {'Ten': 10, 'Eleven': 11}, 'appended element', '
After removing the element at index 5
[1, 2, ('inserted', 'string', 1), 'Three', ('inserted', 'string', 2), 'Four', ('inserted', 'string', 3), [5, 6, 7], (8, 9), {'Ten': 10, 'Eleven': 11}, 'appended element', 'extension 1', 'exten
-----------------------------------------------------------------------------------------------
q1.k
Before sorting:  [12, 3, 5, 234, 23, 21, 2, 52, 312, 32, 423, 5, 2344]
After sorting:  [2, 3, 5, 5, 12, 21, 23, 32, 52, 234, 312, 423, 2344]
-----------------------------------------------------------------------------------------------
q1.l
Before reversing:  [1, 2, ('inserted', 'string', 1), 'Three', ('inserted', 'string', 2), 'Four', ('inserted', 'string', 3), [5, 6, 7], (8, 9), {'Ten': 10, 'Eleven': 11}, 'appended element', 'e
After reversing:  ['Three', ('inserted', 'string', 1), ['nested', 'list'], 'extension 2', 'extension 1', 'appended element', {'Ten': 10, 'Eleven': 11}, (8, 9), [5, 6, 7], ('inserted', 'string'
-----------------------------------------------------------------------------------------------
```

2. Write a Python program to create a tuple with n different data types and implement the two methods: count() and index().

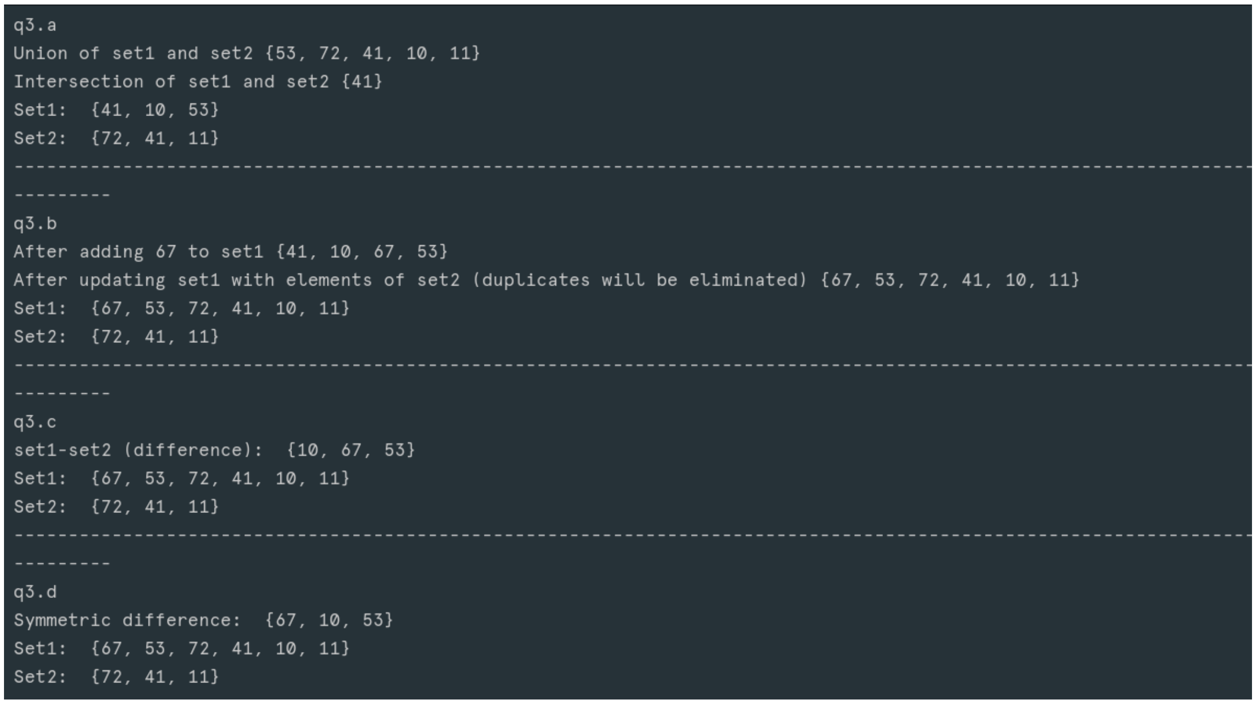
```
# create a tuple
tup =(1,"manoj", [2,4], 4, "manoj");
# printing the count of an item
print("Count of manoj stirng, count = ", tup.count("manoj"));
print("Count of list [2,4], count: ", tup.count([2,4]));
print("-----------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
------")


# printing the index of an element in the tuple
# this element exists
print("Index of the element 4 is", tup.index(4));
print("-----------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
------")

# this element doesnt exist
# gives error
print("Index of element 4.3",tup.index(4.3));
print("-----------------------------------------------------------------------------
-----------------------------------------------------------------------------------------
------")
```



```
Count of manoj stirng, count =  2
Count of list [2,4], count:  1
---------------------------------------------------------------------------------------------------------
---------
Index of the element 4 is 3
---------------------------------------------------------------------------------------------------------
---------

 ---------------------------------------------------------------------------
 ValueError                                Traceback (most recent call last)
 /tmp/ipykernel_153577/2619972784.py in <module>
      14 # this element doesnt exist
      15 # gives error
 ---> 16 print("Index of element 4.3",tup.index(4.3));
      17 print("----------------------------------------------------------------------------------------------
 --------------------------")

 ValueError: tuple.index(x): x not in tuple
```

3. Write a Python program to create two sets (S1 and S2) with n number of different elements [add elements to the sets S1 and S2, such that there are at least 2 common elements between them] and perform the following functions: (Set Methods)

```
# Creating the sets
set1 = {10,41,53}
set2 = {11,41,72}


# 3 a.Perform union and intersection
print("q3.a");
# perfoming the union
print("Union of set1 and set2", set1.union(set2));
# performing the intersection
print("Intersection of set1 and set2", set1.intersection(set2));
print("Set1: ", set1);
print("Set2: ", set2);
print("-----------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
------")


# 3 b.Add elements using add () and update () methods
print("q3.b")
# performing add()
set1.add(67)
print("After adding 67 to set1", set1)
# performing the update()
set1.update(set2)
print("After updating set1 with elements of set2 (duplicates will be eliminated)", set1)
print("Set1: ", set1);
print("Set2: ", set2);
print("-----------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
------")



# 3 c.Perform S1 – S2
print("q3.c")
print("set1-set2 (difference): ", set1.difference(set2))
print("Set1: ", set1);
print("Set2: ", set2);
print("-----------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
------")



# 3 d.Find the Symmetric Difference of S1 and S2
print("q3.d")
print("Symmetric difference: ",set1.symmetric_difference(set2))
print("Set1: ", set1);
print("Set2: ", set2);
print("-----------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
------")
```



```
q3.a
Union of set1 and set2 {53, 72, 41, 10, 11}
Intersection of set1 and set2 {41}
Set1:  {41, 10, 53}
Set2:  {72, 41, 11}
------------------------------------------------------------------------------------------------------------------
---------
q3.b
After adding 67 to set1 {41, 10, 67, 53}
After updating set1 with elements of set2 (duplicates will be eliminated) {67, 53, 72, 41, 10, 11}
Set1:  {67, 53, 72, 41, 10, 11}
Set2:  {72, 41, 11}
------------------------------------------------------------------------------------------------------------------
---------
q3.c
set1-set2 (difference):  {10, 67, 53}
Set1:  {67, 53, 72, 41, 10, 11}
Set2:  {72, 41, 11}
------------------------------------------------------------------------------------------------------------------
---------
q3.d
Symmetric difference:  {67, 10, 53}
Set1:  {67, 53, 72, 41, 10, 11}
Set2:  {72, 41, 11}
```