LAB # 09

SEARCHING & SORTING

EXERCISE

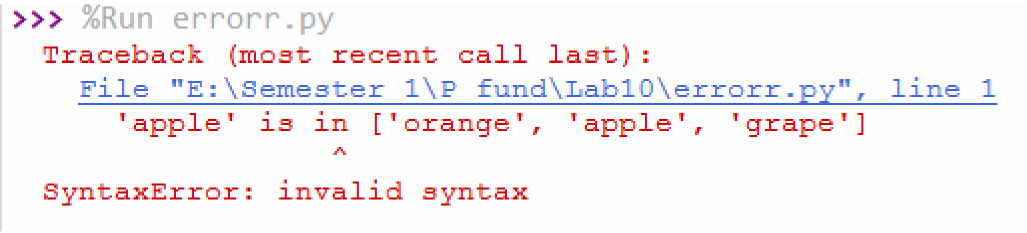
A. Point out the errors, if any, and paste the output also in the following Python programs.

1. Code

```
'apple' is in ['orange', 'apple', 'grape']
```

Output

Print Statement is missing



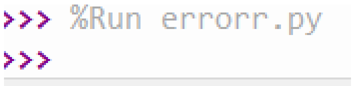
```
>>> %Run errorr.py
  Traceback (most recent call last):
    File "E:\Semester 1\P fund\Lab10\errorr.py", line 1
      'apple' is in ['orange', 'apple', 'grape']
                 ^
  SyntaxError: invalid syntax
```

2. Code

```
def countX(lst, x):
    return lst.count(x)
```

Output:

X is causing an error in the calling function countX



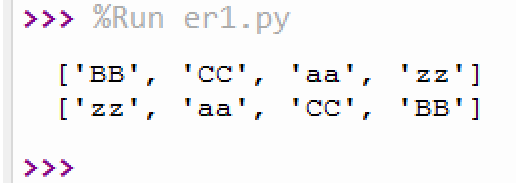
```
>>> %Run errorr.py
>>>
```

What will be the output of the following programs:

1. Code

```
strs = ['aa', 'BB', 'zz', 'CC']
print (sorted(strs))
print (sorted(strs, reverse=True))
```

Output

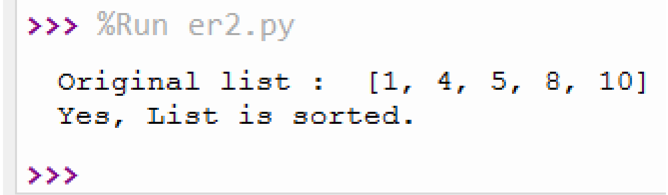


```
>>> %Run er1.py

  ['BB', 'CC', 'aa', 'zz']
  ['zz', 'aa', 'CC', 'BB']

>>>
```

2. Code

```
test_list = [1, 4, 5, 8, 10]
print ("Original list : " , test_list)

# check sorted list

if(test_list == sorted(test_list)):
    print ("Yes, List is sorted.")
else :
    print ("No, List is not sorted.")
```

Output



```
>>> %Run er2.py

  Original list :  [1, 4, 5, 8, 10]
  Yes, List is sorted.

>>>
```

C. Write Python programs for the following:

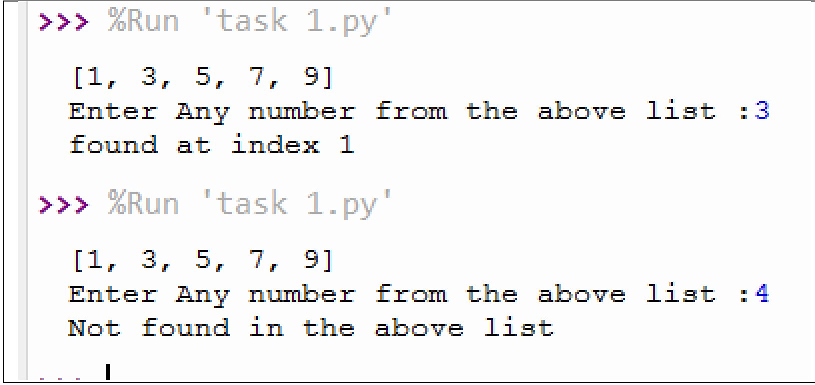
1. Write a program that take function which implements linear search. It should take a list and an element as a parameter, and return the position of the element in the list. The algorithm consists of iterating over a list and returning the index of the first occurrence of an item once it is found. If the element is not in the list, the function should return ‘not found’ message.

Code:

```
list = [1, 3, 5, 7, 9] #list of odd numbers
print(list)
num = int(input("Enter Any number from the above list :")) #selection of number
length = len(list)

def search(list , length, num): # to search the entered number in the list
```

```
    for i in range(0, length):
        if (list[i] == num):
            return i
    return 0

result = search(list, length, num)

if(result == 0):
    print("Not found in the above list")
else:
    print("found at index", result)
```

Output:



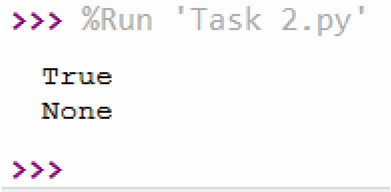
```
>>> %Run 'task 1.py'

  [1, 3, 5, 7, 9]
  Enter Any number from the above list :3
  found at index 1

>>> %Run 'task 1.py'

  [1, 3, 5, 7, 9]
  Enter Any number from the above list :4
  Not found in the above list
```

2. Write a program that create function that takes two lists and returns True if they have at least one common member. Call the function with atleast two data set for searching.

Code:

```
def common_element(list1, list2):
    result1 = False
    for x in list1:
        for y in list2:
            if x == y:
                result = True
                return result
#For list 1
print(common_element([1,2,3,4,5,6,7,8,9,10], [2,4,6,8,10]))
#For list 2
print(common_element([2,4,6,8,10], [1,3,5,7,9]))
```

Output:



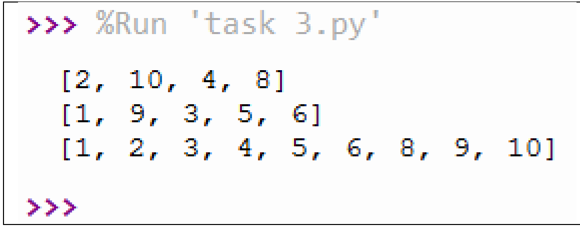
```
>>> %Run 'Task 2.py'

  True
  None

>>>
```

3. Write a program that create function that merges two sorted lists and call two list with random numbers.

Code:

```
def merging(list1, list2):
    print(list1)
    print(list2)
    list3 = list1 + list2
    list3.sort()
    print(list3)
merging([2,10,4,8],[1,9,3,5,6])
```

Output:



```
>>> %Run 'task 3.py'

  [2, 10, 4, 8]
  [1, 9, 3, 5, 6]
  [1, 2, 3, 4, 5, 6, 8, 9, 10]

>>>
```