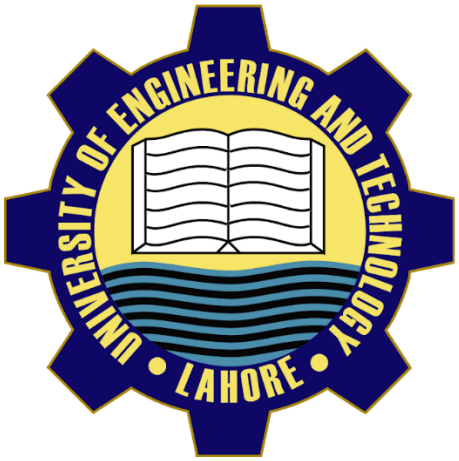
MCT-242 : COMPUTER PROGRAMMING-I

using Python 3.9



Prepared By:

Mr. Muhammad Ahsan Naeem

YouTube Playlist

https://www.youtube.com/playlist?list=PLWF9TXck7O_wMDB-VriREZ6EvwkWLNB7q

LAB 17: LIST OPERATIONS AND LIST METHODS

List Operations:

The two list operations known as list concatenation (+ Operator) and repetition (* Operator) along with list comparison are explained in the table below:

Expression	Result
`[5,3,"Hello"]+[2,4]`	`[5,3,"Hello",2,4]`
`[1]*5`	`[1,1,1,1,1]`
`[1,5,2]*3`	`[1,5,2,1,5,2,1,5,2]`
`[2,5,3]==[2,5,3]`	`True`
`[2,5]>[2,3,7]`	`True`

Empty List:

We can also define an empty list i.e. a list with no element inside it. The use of such list will be clear soon. There are two ways to define an empty list.

```
a=[]
b=list()
```

Built-in List Methods:

There are few built-in very useful methods in list class for its processing and operations. We will explore these step by step:

Method	Description	Example	Output

`append(x)`	Appends x at the end of the list.	`a=[5,4]`
		`a.append(6)`
		`print(a)`
			`[5, 4, 6]`

`insert(i,x)`	Inserts x at i-th index	`a=[5,4,1]`
		`a.insert(1,6)`
		`print(a)`
			`[5, 6, 4, 1]`

`pop(i)`	Removes the list element at index i.	`a=[5,4,1]`
		`a.pop(1)`
		`print(a)`
			`[5, 1]`

`remove(x)`	Removes the element x from list.	`a=[5,4,1]` `a.remove(1)`	`[5, 4]`

		`print(a)`	

`count(x)`	Gives the number of time x I present in the list	`a=[5,4,4,1]`
		`b=a.count(4)`
		`print(b)`
			`2`

`index(x)`	Returns the index of x inside list. If there are multiple x in list it will return the index of first one.	`a = [3,6,10,6]` `print(a.index(6))`	`1`

`clear()`	Clears all entries of a list	`a = [3,6,10,6]`
		`a.clear()`
		`print(a)`
			`[]`

`sort`	Sorts list in ascending order.	`a = [3,4,1,7,8,2]`
		`a.sort()`
		`print(a)`
			`[1, 2, 3, 4, 7, 8]`

	Can also sort list in reverse order.	`a = [3,4,1,7,8,2]`
		`a.sort(reverse=True)`
		`print(a)`
			`[8, 7, 4, 3, 2, 1]`

`reverse()`	Reverses a list	`a = [3,6,10,7,8,2]`
		`a.reverse()`
		`print(a)`
			`[2, 8, 7, 10, 6, 3]`

`extend`	Extends a list at the end of another list similar to + operator	`a = [3,6,10]`
		`b=[4,5]`
		`b.extend(a)`
		`print(b)`
			`[4, 5, 3, 6, 10]`

Built-in Python Functions for List:

Some Python built-in functions for list are given here:

Method	Description	Example	Output

`len(a)`	Returns the length of the list.	`a = [3,6,10,6]`
		`b=len(a)`
		`print(b)`
			`4`

`max(a)`	Returns the maximum element of the list	`a = [3,6,10,6]` `print(max(a))`	`10`
`min(a)`	Returns the minimum element of the list	`a = [3,6,10,6]` `print(min(a))`	`3`

`sorted(a)`	a is the list which will get sorted in ascending order	`a = [3,4,1,7,8,2]`
		`b=sorted(a)`
		`print(b)`
			`[1, 2, 3, 4, 7, 8]`

	List a can also be sorted in reverse order.	`a = [3,4,1,7,8,2]`
		`b=sorted(a,reverse=True)`
		`print(b)`
			`[8, 7, 4, 3, 2, 1]`

`sum(a)`	All elements of list a are summed up.	`a = [3,4,1,7,8,2]`
		`b=sum(a)`
		`print(b)`
			`25`

`all(a)`	Returns True if all values inside the list are True. Only 0 and False are treated as not True, any other value is treated as True. In other words, if there is one or more False or 0 inside the list, it will return False.	`a=[2,3,5,'Python']`
		`b=[1,2,0]`
		`print(all(a))`
		`print(all(b))`
			`True`
			`False`

`any(a)`	Returns True if there is at least one True (any thing other than 0 or False) in the list. Returns False when all values inside the list are False or 0.	`a=[0,3,5,False]`
		`b=[0,False]`
		`print(any(a))`
		`print(any(b))`
			`True`
			`False`

Other than above built-in Python functions, there is a Python keyword or statement named as **`del`** which can be applied on any variable to delete it and likewise can be applied on a list. We can also specify a list slice instead of complete list to delete specific elements from the list.

`del`	Another way to delete list element via its index	`a=[5,4,1]`
		`del a[2]`
		`print(a)`
			`[5, 4]`

Filling list with user's input values:

You can see the use of **`append()`** method as adding an element at the end of the existing list. One possible use is shown in the code below where user is allowed to enter as many numbers as he wants and enters -1 to finish the process. An empty list is defined first, and each new entry is appended to it until -1 is entered.

```
numList=[]
while True:
    x=eval(input("Enter Next number (-1 to exit):"))
    if x==-1:
        break
    numList.append(x)

print("You entered these numbers:")
print(numList)
```

Tasks:

[1] We will define following list of 20 elements in our program:

```
a = [7,3,6,10,6,7,-2,7,5,-8,23,12,-22,3,6,-5,7,5,10,-20]
```

And will do the following tasks on it. We have seen detail of filling a list with user's input numbers. But for this task we will not take user's data and will define the above list in program. The purpose is to understand and use the methods and functions given in last lab session.
Secondly, you are not allowed to use `for` loop or `while` loop for the below tasks. The first task is solved for your guidance.

a. **Find the index of maximum value inside the list.**
   From table of last lab session, we can see that we have `max()` function for the maximum value of list and `index()` method to find index of any element. We can use both of them as shown here:

   ```
   print(a.index(max(a)))
   ```

   This will display 10 which is the index of 23, the maximum value in the list.

b. **Find how many numbers are less than 7 in the list.**
   - Sort the list. The original list must not be overwritten i.e. the sorted list must be a new list keeping the original list as there.
   - Find the index of 7
   - Relate index of 7 in sorted list with the number of elements less than 7

c. **Find how many numbers are greater than 7 in the list.**
   First Approach:
   - Sort list in reverse order. The original list must not be overwritten i.e. the sorted list must be a new list keeping the original list as there.
   - Find the index of 7
   - Relate index of 7 in reverse sorted list with the number of elements greater than 7

   Second Approach:
   - Find number of elements less than 7 as in part b.
   - Count number of times 7 is in the list.
   - Find total number of elements i.e., length of the list.
   - Relate above three number to find the number of elements greater than 7

d. **Find the index of second largest number in the list. If there are two or more instances of the largest value, the second largest can be considered same as the first largest. (later, we will see the case where the second largest means the one after all occurrences of first largest).**
   - Sort the list. The original list must not be overwritten i.e., the sorted list must be a new list keeping the original list as there.

- Find the second last number in the sorted list.
- Find the index of above number from original list.

e. **Replace the minimum value (only the first instance in case there are multiple) by the maximum value.**
   - Find the minimum and the maximum value.
   - Find index of the minimum value
   - Replace that index value with maximum value.

[2] Write a program that will ask user to enter numbers and will then find and display the average, the highest and the lowest values. User can enter as many numbers as he wants and will enter `-1` to end the process. Use one `while` loop to take in the entries as shown above and then use **only** one `for` loop to calculate the three outputs.

[3] Write a program that will first create a list of 20 random numbers (1 to 50). One `for` or `while` loop will do this part. Then the program should create a new list that will contain all prime numbers in the first list. Using a function `isPrime` is up to you.

Sample Output is:

```
A random List: [5, 16, 28, 12, 44, 13, 40, 35, 21, 37, 24, 31,
27, 13, 34, 19, 41, 33, 29, 34]
List of Prime numbers: [5, 13, 37, 31, 13, 19, 41, 29]
```

Using range() to fill list element in a sequence:

Filling a list with elements following a particular sequence can be done using `range()` and `list()` function. For example, if you want to fill a list with integers 0 to 9, this can be directly done as:

```
a=list(range(10))
print(a)
```

Guess the output of following code and verify:

```
a=list(range(100,0,-10))
print(a)
```

Basically `list()` is a built-in Python function that converts other data types to a list. This can be used on string as:

```
x="Hello"
a=list(x)
print(a)
```

The output will be:

```
['H', 'e', 'l', 'l', 'o']
```

This method can also be used to convert tuple (not covered yet) to a list as:

```
x=(43,22,'Yes')
a=list(x)
print(a)
```

And can covert a set (not covered yet) to a list as:

```
x={4,2,7}
a=list(x)
print(a)
```

Slicing a list:

We can access a single element of a list using its index inside square brackets. We can also access a part of the list using the similar format. This is known as list slicing. To access a slice of list named as **a**, we can use following format:

```
a[start:stop:steps]
```

The above format is very similar to **range()** format. The last parameter **steps** is optional having the default value of 1. So, if **a** is having elements from index 1 to 20 and we want to access its elements with index starting from 4 to 10, we can do it as:

```
a=list(range(1,21))
print(a[4:11])
```

Note carefully that to access index till 10 we need to pass the **stop** parameter as 11.

If we don't specify **stop** and **steps** parameters, list will be sliced from start to the end of the list. For example, to access list elements from index 4 to the last element, it can be done as:

```
print(a[4:])
```

Now if we write:

```
print(a[:5])
```

This is going to access element from **start** to index 4 and here will be the output:

```
[1, 2, 3, 4, 5]
```

Now think about the output of following and verify:

```
print(a[15:])
print(a[::-1])
```

Functions in random module for list:

Here are useful functions in random module that work on list:

Function	Description	Example	Sample Output

`choice(a)`	Picks a random item from list **a**.	`from random import choice`
		`a=[5,4,1]`
		`print(choice(a))`
			`5`

`sample(a,n)`	Picks a group of **n** random items from list **a**.	`from random import sample`
		`a = [3,6,10,6]`
		`print(sample(a,2))`
			`[10,3]`

`shuffle(a)`	Shuffles the items of list **a**	`from random import shuffle`
		`a = [3,6,10,6]`
		`shuffle(a)`
		`print(a)`
			`[6, 6, 3, 10]`

In task 3 of this lab session, you were asked to create a list of 20 random numbers between 1 to 50. At that time there was possibility of repeated numbers. Now, think about generating the same list ensuring that no two numbers will be same using above functions.

Tasks:

[4] Result of Computer Programming-I: Program will ask the user for the marks of the students in subject Computer Programming-I. Program will first ask for the number of students in the class and then will ask for marks of every individual. Passing criteria is 40 marks. Your program should display the summary of result that includes:

- Average Marks
- No. of Failing Students
- Highest Marks obtained along with the roll number.
- Lowest Marks obtained along with the roll number.

The result should be displayed in a box at the center of blank window (clearing the previous screen).

Sample Output is:

```
This Program displays the result of Computer Programing-I
Enter total students in class: 5
Enter Marks of roll no. 1 : 72
Enter Marks of roll no. 2 : 62
Enter Marks of roll no. 3 : 81
Enter Marks of roll no. 4 : 32
Enter Marks of roll no. 5 : 54


****************************************
* Result of Computer Programming-I     *
****************************************
* Average Marks: 60.2                  *
* Failing Students: 1                  *
* Highest Marks:81 Obtained by Roll No. :3   *
```

```
* Lowest Marks: 32 Obtained by Roll No. :4   *
*********************************************
```

[5] Write a program that will create a list named as **`List1`** that contains 20 random numbers between 1 and 10. It then creates another list named as **`List2`** with 10 elements such that the first element is the number of 1's in **`List1`**, second element is number of 2's and so on till the 10^{th} element is number of 10's in **`List1`**. Display both lists on screen.

[6] Write a program that asks the user for an integer and then creates and displays a list that consists of all factors of that integer.

[7] Write a program that rotates the elements of a list so that the element at the first index moves to the second index, the element in the second index moves to the third index, etc., and the element in the last index moves to the first index. You can define original list within program by yourself.

[8] Write a program that generates 100 random integers that are either 0 or 1. Then find the longest run of zeros i.e., the largest number of consecutive zeros. For instance, the longest run of zeros in [1,0,1,1,0,0,0,0,1,0,0] is 4.

[9] Write a program that removes any repeated items from a list so that each item appears at most once. For instance, the list [1,1,2,3,4,3,0,0] would become [1,2,3,4,0].