Lab 4: Python Lists

Objectives

In this lab, we aim to get hands-on experience with Python lists, one of the most versatile data structures in Python. By the end of the lab, you will be able to:

- Create, access, and modify lists efficiently.
- Use loops and conditional statements to work with list elements.
- Perform common list operations like summing, multiplying, sorting, and reversing items.
- Identify and remove duplicates from lists.
- Find specific elements such as the largest, smallest, second largest, and second smallest numbers.
- Work with both strings and numbers in lists, including sorting numerically.
- Apply built-in methods like append, remove, pop, insert, sort, max, and min.
- Merge, copy, and manipulate multiple lists.
- Solve practical problems such as filtering even numbers, checking for prime numbers, or adding prefixes to items in a list.

Theory

Lists in Python are a flexible way to store multiple values in a single variable. They can hold numbers, strings, or even other lists, making them incredibly useful for a wide range of programming tasks.

We can create and access lists using indices, and slice them to work with sublists. Lists can be modified dynamically with methods like append(), insert(), remove(), and pop().

Loops are essential for processing lists. With for or while loops, we can go through each element to perform operations like calculating a sum, multiplying numbers, or filtering items based on conditions. Conditional statements (if, elif, else) allow us to make decisions inside loops, such as checking for prime numbers or removing certain elements.

Python also provides a set of built-in functions and list methods for common tasks: len() to get the size of a list, sum() to add numbers, max() and min() to find extreme values, and sort() or reverse() for ordering elements. We can also merge lists, remove duplicates, or work with nested lists to handle more complex data structures.

By mastering these techniques, we can efficiently manage and manipulate collections of data, making Python lists an essential tool for programming.

Program:

1. Write a Python Program to print all the items in a list.

```
list = ['apple', 'banana', 'cherry'] print(list)
```

Output:
['apple', 'banana', 'cherry']

2. Use range function to print all the even numbers from 1 to 10.

```
for i in range(1,11):
if i % 2 == 0:
        print(i, end = ' ')
```

Output:
2 4 6 8 10

3. Write a Python Program to get the largest and smallest number in a list without using builtin functions.

```
list = [7, 5, 3, 9, 4, 6] list.sort()
print("\nSorted list:", list) print("Smallest
element:",(list[0]))
print("Largest element:",(list[-1]))
```

Output:
Sorted list: [3, 4, 5, 6, 7, 9]
Smallest element: 3
Largest element: 9

4. Write a Python program to find the second smallest and second largest numbers in a list.

```
list = [7, 5, 3, 9, 4, 6] list.sort()
print("\nSorted list:", list) print("Second
smallest element:",(list[1]))
print("Second largest element:",(list[-2]))
```

Output:
Sorted list: [3, 4, 5, 6, 7, 9]
Second smallest element: 4
Second largest element: 7

5. Write a Python program to sum all the items in a list.

```
list = [9, 5, 3, 8, 4, 6] sum = 0 for i in list:
sum += i print("\nSum of all items in the
list:", sum)
```

Output:
Sum of all items in the list: 35

6. Write a Python program to multiply all the items in a list.

```
list = [1, 5, 2, 3, 4, 6]
multiply = 1 for i in
list:
    multiply *= i print("\nProduct of all items in
the list:", multiply)
```

Output:
Product of all items in the list: 720

7. Write a Python program to check if a list is empty or not.

```
list = [] if
not list:
   print("The list is empty.") else:
   print("The list is not empty.")
```

Output:
The list is empty.

8. Write a Python program to print the numbers of a specified list after removing even numbers from it.

```
list =[2, 5, 4, 1, 8, 7]
for i in list:     if i %
2 != 0:
       print(i, end=' ')
```

Output:
5 1 7

9. Write a Python Program to print a specified list after removing the 0th, 4th and 5th element. Sample list: ['Red', 'Green', 'White', 'Black', 'Pink', 'Yellow'] Expected Output: ['Green', 'White', 'Black']

```
colors = ['Red', 'Green', 'White', 'Black', 'Pink', 'Yellow'] new_colors
= [ ]

for i in range(len(colors)):     if i
!= 0 and i != 4 and i != 5:
new_colors.append(colors[i])
print(new_colors)
```

Output:
['Green', 'White', 'Black']

10. Write a Python Program to check if each number is prime in a given list of numbers. Print the prime numbers if any, otherwise print "no prime numbers".

```
num_list = [2, 4, 7, 9, 12, 11]
prime_numbers = [] for num in
num_list:     if num > 1:        for i in
range(2, int(num**0.5) + 1):            if
(num % i) == 0:
              break
else:
          prime_numbers.append(num) if
prime_numbers:
   print("Prime numbers in the list:", prime_numbers)
```

Output:
Prime numbers in the list: [2, 7, 11]

11. Write a Python program to remove duplicates from a list.

```
original_list = [7, 5, 3, 5, 9, 7, 2]
print("Original list:", original_list) new_list
= []

for item in original_list:     if item not in
new_list:          new_list.append(item)
print("List after removing duplicates:", new_list)
```

Output:
Original list: [7, 5, 3, 5, 9, 7, 2]
List after removing duplicates: [7, 5, 3, 9, 2]

12. Write a Python Program to merge two lists and removes all duplicates from the combined list.

```
list_1 = ['banana'] list_2 =
['apple', 'cherry']
list_1.extend(list_2)
print("Extended list:", list_1)
```

Output:
Extended list: ['banana', 'apple', 'cherry']

13. Write a Python Program to print a list in a reverse order.

```
list = [7, 5, 3, 9, 4, 6]
print("REVERSED =",list[::-1])
```

Output:
REVERSED = [6, 4, 9, 3, 5, 7]

14. Write a Python program to sort a given list of strings (numbers) numerically. Original list: ['12', '45', '7', '0', 100', '200', '-12', '-500'] Sort the said list of strings(numbers) numerically: [-500, -12, 0, 4, 7, 12, 45, 100, 200]

```
num = ['12', '45', '7', '0', '100', '200', '-12', '-500']
print("Original list:", num) num =
[int(x) for x in num] num.sort(key =
int) print("Sorted list numerically:",
num)
```

Output:
Original list: ['12', '45', '7', '0', '100', '200', '-12', '-500']
Sorted list numerically: [-500, -12, 0, 7, 12, 45, 100, 200]

15. Apply various functions like append, remove, pop, insert, sort, max, min in a list.

```
list = [7, 5, 3, 10, 11] list.append(9)
print("List after appending 9:", list)
list.remove(5) print("List after
removing 5:", list) list.pop(0)
print("List after popping the first element:", list)
list.insert(1, 4) print("List after inserting 4
at index 1:", list) list.sort() print("List after
sorting:", list)
max(list) print("Maximum element in the list:",
max(list)) min(list) print("Minimum element in
the list:", min(list))
```

Output:
List after appending 9: [7, 5, 3, 10, 11, 9]
List after removing 5: [7, 3, 10, 11, 9]
List after popping the first element: [3, 10, 11, 9]
List after inserting 4 at index 1: [3, 4, 10, 11, 9]
List after sorting: [3, 4, 9, 10, 11]
Maximum element in the list: 11
Minimum element in the list: 3

16. Write a python program to copy a content of one list to another.

```
original_list = [1, 2, 3, 4]
new_list = original_list.copy()

print("Original:", original_list)
print("Copy:", new_list)
```

Output:
Original: [1, 2, 3, 4]
Copy: [1, 2, 3, 4]

17. Write a Python Program to sort a list of lists by a given index of the inner list.

```
data = [[2, 3, 5], [0, 1, 4],  [6, 2, 0] ]

data.sort(key=lambda x: x[0])
print("Sorted list:", data)
```

Output:
Sorted list: [[0, 1, 4], [2, 3, 5], [6, 2, 0]]

18. Write a Python program to generate and print a list of the first and last 5 elements where the values are square numbers between 1 and 15 (both included).

```
square_numbers = [ ]

for i in range(1, 16):
   square_numbers.append(i ** 2)

first_5 = square_numbers[:5]
last_5 = square_numbers[-5:]

print("List of square numbers:", square_numbers)
print("First 5 elements:", first_5)
print("Last 5 elements:", last_5)
```

Output:
List of square numbers: [1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225]
First 5 elements: [1, 4, 9, 16, 25]
Last 5 elements: [121, 144, 169, 196, 225]

19. Write a Python program to insert a given string at the beginning of all items in a list. Sample list : [1,2,3,4], string : emp Expected output : ['empl 'emp2', 'emp3', 'emp4']

```
list = [1, 2, 3, 4] string
= 'emp' for i in
range(len(list)):
    list[i] = string + str(list[i])
print("Modified list:", list)
```

Output:
Modified list: ['emp1', 'emp2', 'emp3', 'emp4']

20. Write a Python program to insert n values in a list and find those values in a list that are greater than a specified number.

```
n = int(input("How many numbers? ")) compare_number =
int(input("Enter a number to compare: ")) greater_values = []

for i in range(n):
    num = int(input(f"Enter number {i+1}: "))     if num >
compare_number:         greater_values.append(num) print("Numbers
greater than", compare_number, "are:", greater_values)
```

Output:
Numbers greater than 67 are: [89]

Conclusion:

In this lab, we explored Python lists in depth and learned how to use them effectively in different scenarios. We practiced creating, modifying, and accessing list elements, and used loops and conditional statements to perform various operations. We also learned how to handle duplicates, sort lists numerically and alphabetically, and manipulate nested lists. Through these exercises, we not only strengthened our understanding of Python list operations but also developed practical skills for solving real-world problems efficiently. Working with lists helps us organize data, perform calculations, and implement logic in a clear and structured way, which is an essential skill for any Python programmer.