Lists in Python

1. What is a List?

- A list is a data structure used to store multiple items in a single variable.
- A Data Structure is a way of organizing, storing, and managing data in a computer so that it can be accessed and modified efficiently.
- A list is a variable that can store more than one value at the same time.
- Lists can store:
    - Numbers(integers,floats)
    - Strings
    - Booleans
    - Different data types

2. List Syntax

List_name = [elem1, elem2, elem3…elemn]

3. Data Types in a List

a. Numbers

Example 1

```
Numbers = [10, 20, 30]  # a list with three integers
Numbers = [10.25, 20.75, 30.67,67.50]  # a list with four floats
```

b. Strings

Example 2

```
Names = ["Mohamed", "Khadidja", "Ali"]# a list with three strings
```

c. Booleans

Example 3

```
Note=[True, False]
# a list with two booleans
```

c. Different data types together

Example 4

```
data = [1, "Python", True, 3.5]
```

4. Order of Elements (Index)

Each element in a list has a position called an index

```
Names = ["Mohamed", "Fatima", "Ali"]
print(Names[0])  # Mohamed
print(Names[1])  # Fatima
```

Note: Lists are mutable, which means you can change their values after creating them.
Example 4

```
Numbers = [10, 20, 30]
print(Numbers[0])
Numbers[0] = 100
print(Numbers[0])
```

5. Display a List

Method 1

```
for i in range(len(Numbers)):
  print(Numbers[i])
```

Method 2

```
for item in Numbers:
  print(item)
```

Exercise

Write a Python program that does the following:

1. Create a list called fruits that contains: apple, banana, orange
2. Print the entire list
3. Print the first element of the list
4. Change "banana" to "grape"
5. Print the list after the modification

```
fruits = ["apple", "banana", "orange"]
print(fruits)
print(fruits[0])
fruits[1] = "grape"
print(fruits)
```

Output:

```
['apple', 'banana', 'orange']
apple
['apple', 'grape', 'orange']
```

6. List Functions

6.1 append: adds one element to the end of a list.

Syntax:

list_name.append(element)

Example 5

```
Numbers = [1, 2, 3]
print(Numbers)
Numbers.append(4)
print(Numbers)
```

Output:

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

6.2 insert: insert an element at a specific index

Syntax:

list_name.insert(index, element)

Example 6

```
Numbers = [1, 2, 3]
print(Numbers)
Numbers.insert(1, 10)
print(Numbers)
```

Output:

```
[1, 2, 3]
[1, 10, 2, 3]
```

6.3 remove: remove an element by its value

Syntax:

list_name.remove(element)

Example 7

```
numbers = [1, 10, 2, 3]
numbers.remove(10)
print(numbers)
```

Output:

```
[1, 2, 3]
```

6.4 pop(): remove an element by index (default last element)

Syntax:

```
list_name.pop()
list_name.pop(index)
```

Example 8

```
Numbers = [1, 2, 3, 4, 5]
Numbers.pop()
print(Numbers)
Numbers.pop(0)
print(Numbers)
```

Output:

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

6.5 len: returns the number of elements in the list

```
len(list_name)
```

Example 9

```
numbers = [1, 2, 3]
print(len(numbers))
```

Output:

```
3
```

6.6 sort: Sorts the list in ascending order.

Syntax:

```
list_name.sort()
```

Example 10

```
numbers = [3, 1, 4, 2]
numbers.sort()
print(numbers)
```

Output:

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

Sorting in descending order:

```
numbers = [5, 2, 9, 1]
numbers.sort(reverse=True)
print(numbers)
```

Output:

[9, 5, 2, 1]

6.7 reverse: reverse the order of the list

Syntax:

list_name.reverse()

Example 11

```
numbers = [1, 2, 3]
numbers.reverse()
print(numbers)
```

Output:

[3, 2, 1]

6.8 clear: remove all elements from the list

Syntax:

list_name.clear()

6.9 index: find the position (index) of a specific element in a list

Syntax:

list_name.index(value, start, end)

- value: the element you are looking for (required)
- start: the starting index for the search (**optional**)
- end: the ending index for the search (**optional**)

Example 12

```
numbers = [5, 10, 15, 20]
position = numbers.index(15)
print(position)
```

Output:

2

Example 13

```
numbers = [1, 2, 3, 2, 4]
result = numbers.index(2, 2)
print(result)
```

Output:

3

The search starts from index 2, so the first 2 is ignored.

- index() finds the position of an element in a list
- It returns the first occurrence only
- It causes an error if the element is not found
- Always check with in to avoid errors

Example

```
if 100 in numbers:
   print(numbers.index(100))
```

6.10 Slicing

Slicing allows you to extract a part of a list by specifying start, end, and step

new_list = list[start:end:step]

Example 14

```
numbers = [10, 20, 30, 40, 50]
print(numbers[0:3])
```

Output:

[10, 20, 30]

```
print(numbers[2:])
```

Output:

[30, 40, 50]

Extract first 4 elements with step 2

```
print(numbers[:4:2])
```

Output:

[10, 30]

Exercise

Write a Python program that does the following:

1. Create a list called numbers with the values: 4, 8, 2, 6
2. Add the number 10 to the end of the list
3. Insert the number 5 at index 2
4. Remove the number 8 from the list

5. Remove the last element from the list
6. Sort the list in ascending order
7. Print the final list

```
# 1. Create the list
numbers = [4, 8, 2, 6]

# 2. Add 10 to the end
numbers.append(10)

# 3. Insert 5 at index 2
numbers.insert(2, 5)

# 4. Remove the number 8
numbers.remove(8)

# 5. Remove the last element
numbers.pop()

# 6. Sort the list
numbers.sort()

# 7. Print the final list
print(numbers)
```

7. Nested List

A nested list is a list that contains other lists as its elements.
Syntax:

List_name = [[..., ...], [..., ...], ...]

Example:

nested_list = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

- nested_list is the main list
- Each element inside it is a sublist

7.1 Accessing Elements in a Nested List

To access an element, we use two indices
Syntax:

list_name[row][column]

Example:

```
Matrix = [[10, 20], [30, 40]]
print(Matrix[0])      # [10, 20]
print(Matrix[0][1])   # 20
print(Matrix[1][0])   # 30
```

Modifying element:

```
Matrix[1][1] = 99
print(Matrix)
```

7.2 Looping Through a Nested List

a. **Printing row by row:**

```
Matrix = [[10, 20], [30, 40], [50, 60]]
for row in Matrix:
   print(row)
```

b. **Using nested for loops:**

Method 1

```
for i in range(len(Matrix)):
  for j in range(len(Matrix[i])):
      print(Matrix[i][j])
```

Method 2

```
for row in Matrix:
  for ele in row:
    print(ele)
```

7.3 Sum of All Elements

```
Sum = 0
for row in Matrix:
  for element in row:
    Sum += element
print(Sum)
```

7.4 Finding the Maximum Element

Example

```
matrix = [[3, 8, 2], [5, 1, 9]]
max_value = matrix[0][0]
for row in matrix:
   for number in row:
      if number > max_value:
         max_value = number
print(max_value)
```

7.5 Syntax – f-string formatting for decimals

f"{variable:.nf}"

- variable → the variable or value you want to format
- n → number of decimal places
- f → means float (decimal number)

Exercise

We have the following nested list:

```
grades = [
   [12, 15, 14],
   [10, 11, 13],
   [16, 18, 17]
]
```

1. Add the grade 20 to the end of the first inner list.
2. Insert the grade 9 at the beginning of the second inner list.
3. Insert a new inner list [11, 12, 13] as a new subject at the end of the nested list.
4. Remove the grade 11 from the second inner list.
5. Arrange the grades of each inner list in ascending order.
6. Calculate the total of the grades in each inner list.
7. Determine the highest grade in each inner list.

Solution

```
grades = [
   [12, 15, 14],
   [10, 11, 13],
   [16, 18, 17]
]
print(grades)

grades[0].append(20)
print(grades)

grades[1].insert(0, 9)
print(grades)

grades.append([11, 12, 13])
print(grades)

grades[1].remove(11)
```

```
print(grades)

for subject in grades:
    subject.sort()
print(grades)

print("=========Sum: Solution 1==========")
for subject in grades:
   print(sum(subject))

print("======Sum: Solution 2======")
Sum = 0
for i in range(len(grades)):
     for j in range(len(grades[i])):
          Sum += grades[i][j]
     print(Sum)
     Sum = 0

print("=========max: Solution 1==========")
for subject in grades:
    print(max(subject))

print("========max: Solution 2==========")
for i in range(len(grades)):
    max = grades[i][0]
    for j in range(len(grades[i])):
        if grades[i][j] > max:
            max = grades[i][j]
    print("the max of ", i+1, "is: ", max)
```

Solution Exercise 4

```
import statistics

Numbers = []
for i in range(8):
   num = int(input("enter your number "))
   Numbers.append(num)

print(Numbers)

Max = Numbers[0]
for i in range(len(Numbers)):
    if Numbers[i] > Max:
       Max = Numbers[i]
print(Max)

Min = Numbers[0]
```

```
for i in range(len(Numbers)):
    if Numbers[i] < Min:
        Min = Numbers[i]
print(Min)

Avg = 0
for i in range(len(Numbers)):
    Avg += Numbers[i]
Avg = Avg / len(Numbers)
print(Avg)

print(max(Numbers))
print(min(Numbers))
print(statistics.mean(Numbers))

Small = []
Large = []
for elem in Numbers:
    if elem > 10:
        Large.append(elem)
    else:
        Small.append(elem)
print(Large)
print(Small)
```