LAB # 09

LISTING

OBJECTIVE

Exploring list/arrays in python programming.

THEORY

- ✓ A list can store a collection of data of any size unlike arrays.
- ✓ It is a collection which contain **duplicate items**.
- ✓ **Mutable:** items can be modified, replaced, or removed
- ✓ **Ordered:** maintains the order in which items are added
- ✓ **Index-based:** items are accessed using their position (starting from 0)
- ✓ Can store mixed data types (integers, strings, booleans, even other lists)
- ✓ The elements in a list are separated by commas and are enclosed by a pair of **brackets *[ ]*.**

Creating a lists:

A list is a sequence defined by the list class but also have alternative for creation of list without built-in function.

Syntax: Simple list creation – Without Built-in Function

```
#Simple List Creation

thislist = ["Hello!", "17Sw22"]                    # Only Strings
thislist1 = [2,4,6,8]                              # Only Numbers
thislist2 = ['Hi', 22 , "How are you!"]            # Mix Data Types
thislist3 = [3.0, 1, 55]                           # Mix Numbers
thislist4 = []                                     # Empty list

print(thislist)
print(thislist1)
print(thislist2)
print(thislist3)
```

Output:

```
>>> %Run Lab_09.py
['Hello!', '17Sw22']
[2, 4, 6, 8]
['Hi', 22, 'How are you!']
[3.0, 1, 55]
[]
>>>
```

LAB # 09

Syntax: With Built-in function list( ) / list Constructor()

```
list0 = list()                                   # Creates an empty list
list1 = list([1,2,3,4,5])                        # Creates a list with integer elements
list2 = list(["Chinese", "Pakistani", "Turkish"]) # Creates a list with string elements
list3 = list(["17SW22", 22, True, 23, "Male"])   # Creates a mixed list: strings, integers, boolean
list4 = list(["abcd"])                           # Creates a list with a single string element
list5 = list("abcd")                             # Converts string into characters: ['a','b','c','d']
list6 = list(range(2,10))                        # Converts range(2–9) into list: [2,3,4,5,6,7,8,9]

print(list0)
print(list1)
print(list2)
print(list3)
print(list4)
print(list5)
print(list6)
```

Output:

```
>>> %Run Lab_09.py
[]
[1, 2, 3, 4, 5]
['Chinese', 'Pakistani', 'Turkish']
['17SW22', 22, True, 23, 'Male']
['abcd']
['a', 'b', 'c', 'd']
[2, 3, 4, 5, 6, 7, 8, 9]
>>>
```

List Type() and Length()

type(), checks the data type of the list. len(), returns the number of elements in the lis

Example:

```
# Check List Type and Length

mylist = ["Python", 22, 3.5, True]
print(mylist)
print(type(mylist))
print(len(mylist))
```

Output:

```
>>> %Run Lab_09.py
['Python', 22, 3.5, True]
<class 'list'>
4
>>>
```

LAB # 09

Accessing items from the list:

- ✓ An element in a list can be accessed through the ***index operator***, using the following syntax: **myList[index].**
- ✓ List indexes are starts from 0 to len(myList)-1.
- ✓ Negative indexing means beginning from the end, -1 refers to the last item, -2 refers to the second last item etc.

Example:

```
# Accessing the list
a = [10, 20, 30, 40, 50]

print(a[0])        # Prints first element
print(a[-1])       # Prints last element
```

Output:

```
>>> %Run Lab_09.py
10
50
>>>
```

List Slicing [start : end]:

The index operator allows to select an element at the specified index. The slicing operator returns a slice of the list using the syntax ***list[start : end].*** The slice is a sublist from index start to index end – 1.

Example:

```
#Slicing List

exampleList = [10, 20, 30, 40, 50, 60, 70]

print(exampleList[1:4])       # Elements from index 1 to 3 → [20, 30, 40]
print(exampleList[:3])        # Elements from index 0 to 2 → [10, 20, 30]
print(exampleList[3:])        # Elements from index 3 to 5 → [40, 50, 60]
```

Output:

```
>>> %Run task2.py
Slicing 2 to 5 index: [4, 5, 6]
Slicing before 3rd index value: [1, 2, 4]
Slicing after 3rd index value: [5, 6, 7, 8]
```

LAB # 09

List Methods for Adding , Changing and Removing items:
Python has a set of built-in methods that you can use on lists/arrays.

Method	Description
append()	Adds an element at the end of the list
copy()	Returns a shallow copy of the list
count()	Returns the number of elements with the specified value
index()	Returns the index of the first element with the specified value
insert()	Adds an element at the specified position
pop()	Removes the element at the specified position (default last)
remove()	Removes the first item with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list in ascending order (use reverse=True for descending)
clear()	Removes all elements from the list
extend()	Adds all elements of another list (or iterable) to the end of the list
del keyword	Deletes an element or the whole list (not a method, but often used)

List Operations – Add Elements

Example:

```
# Python Program: List Operations – Add Elements

# Initial list
mylist = [10, 20, 30, 40]
print("Original List:", mylist)

# ================================
# 1️⃣ Adding Elements
# ================================

# append(): Adds an element at the end
mylist.append(50)
print("After append(50):", mylist)

# extend(): Adds multiple elements at the end
mylist.extend([60, 70, 80])
print("After extend([60, 70, 80]):", mylist)

# insert(): Adds an element at a specific position
mylist.insert(2, 25)  # Insert 25 at index 2
print("After insert(2, 25):", mylist)
```

LAB # 09

Output:

```
>>> %Run Lab_09.py
Original List: [10, 20, 30, 40]
After append(50): [10, 20, 30, 40, 50]
After extend([60, 70, 80]): [10, 20, 30, 40, 50, 60, 70, 80]
After insert(2, 25): [10, 20, 25, 30, 40, 50, 60, 70, 80]
>>>
```

List Operations – Update Element

Example:

```
# ================================
# 2️⃣ Updating Elements
# ================================

# Initial list
mylist = [10, 20, 30, 40]
print("Original List:", mylist)

# Change element at specific index
mylist[1] = 22
print("After updating index 1 to 22:", mylist)
```

Output:

```
>>> %Run Lab_09.py
Original List: [10, 20, 30, 40]
After updating index 1 to 22: [10, 22, 30, 40]
>>>
```

List Operations – Removing Element

Example:

```
# ================================
# 3️⃣ Removing Elements
# ================================

# Initial list
mylist = [10, 20, 30, 40, 50, 60]
print("Original List:", mylist)

# remove(): Remove first occurrence of value
mylist.remove(30)
print("After remove(30):", mylist)
```

LAB # 09

```
# pop(): Remove element at specific index
mylist.pop(3)  # Removes element at index 3
print("After pop(3):", mylist)

# pop() without index removes last element
mylist.pop()
print("After pop():", mylist)

# del statement: Remove element at specific index
del mylist[0]
print("After del mylist[0]:", mylist)
```

Output:

```
>>> %Run Lab_09.py
Original List: [10, 20, 30, 40, 50, 60]
After remove(30): [10, 20, 40, 50, 60]
After pop(3): [10, 20, 40, 60]
After pop(): [10, 20, 40]
After del mylist[0]: [20, 40]
>>>
```

Iterating Over Lists

Example:

```
# Python Program: Iterating Over a List

fruits = ["Apple", "Banana", "Cherry", "Mango"]

# Using a for loop to iterate over the list
print("List of Fruits:")
for fruit in fruits:
    print(fruit)
```

Output:

```
>>> %Run Lab_09.py
List of Fruits:
Apple
Banana
Cherry
Mango
>>>
```

LAB # 09

EXERCISE

A. Point out the errors, if any, in the following Python programs.

1. Code

```
Def max_list( list ):
    max = list[ 0 ]
    for a is in list:
        elif a > max:
            max = a
    return max
print(max_list[1, 2, -8, 0])
```

Output

2. Code

```
motorcycles = {'honda', 'yamaha', 'suzuki'}
print(motorcycles)
del motorcycles(0)
print(motorcycles)
```

Output:

3. Code

```
Def dupe_v1(x):
  y = []
  for i in x:
    if i not in y:
      y(append(i))
      return y

a = [1,2,3,4,3,2,1]
print a
print dupe_v1(a)
```

Output:

LAB # 09

B. What will be the output of the following programs:

1. Code

```
list1= [1,2,4,5,6,7,8]
print("Negative Slicing:",list1[-4:-1])
x = [1, 2, 3, 4, 5, 6, 7, 8, 9]
print("Odd number:", x[::2])
```

Output

2. Code

```
def multiply_list(elements):
    t = 1
    for x in elements:
        t*= x
    return t
print(multiply_list([1,2,9]))
```

Output

3. Code

```
def add(x,lst=[] ):
    if x not in lst:
        lst.append(x)
        return lst

def main():
    list1 = add(2)
    print(list1)
    list2 = add(3, [11, 12, 13, 14])
    print(list2)
main()
```

Output

LAB # 09

C. Write Python programs for the following:

1. Write a program that store the names of a few of your friends in a list called 'names'. Print each person's name by accessing each element in the list, one at a time.

2. Write a program that make a list that includes at least four people you'd like to invite to dinner. Then use your list to print a message to each person, inviting them to dinner. But one of your guest can't make the dinner, so you need to send out a new set of invitations. Delete that person on your list, use ***del statement*** and add one more person at the same specified index, use the ***insert( )*** method. Resend the invitation.

3. Write a program that take list = [30, 1, 2, 1, 0], what is the list after applying each of the following statements? Assume that each line of code is independent.

- list.append(40)
- list.remove(1)
- list.pop(1)
- list.pop()
- list.sort()
- list.reverse()

4. Write a program to define a function called 'printsquare' with no parameter, take first 7 integer values and compute their square and stored all square values in the list.