Assignment 4

Name : Om Lokhande
PRN No. - 123B1B181

1.What exactly is [] ?

Ans. - In Python [] is used for List declaration.

2. In a list of values stored in a variable called word, how would you assign the value hello as the third value? (Assume [2, 4, 6, 8, 10] are in word.)

Ans. - hello can be assign as a third value by simply accessing the index of third element and assign value hello to it .

Word = [2, 4 ,6, 8, 10]

Word[2] = “hello”

After this operation, the word list will look like this:

Word = [2,4, hello,8,10]

3. What is the value of word[int(int(’3’ * 2) / 11)]?

Ans. - i) '3' * 2’ repeats the string '3' twice, resulting in '33'.

ii) int('33') converts the string '33' to the integer 33.

iii) 33 / 11 gives 3.0.

iv) int(3.0) converts 3.0 to the integer 3.

v) word[3] accesses the element at index 3 in the list, which is 'd'.

4. What is the value of word[-1]?

Ans. - As negative indexing starts from last elemetnt

The index of last element is -1, for second last it is -2, for third last it is -3 and so on. So, word[-1] accesses the last element, which is 'd'.

5. What is the value of word[:2]?

Ans. - word[:2] slices the list from the start (index 0) up to, but not including, index 2.

6. What is the value of bacon.index(‘cat’)?

Ans. - The value of bacon.index('cat') is 1, as 'cat' first appears at index 1.

Index method Is used to find index of an element

7. How does bacon.append(99) change the look of the list value in bacon?

Ans. - After calling bacon.append(99), the list will look like this: [3.14, 'cat', 11, 'cat', True, 99].

As append method is used to put element at the last of the list

8. How does bacon.remove('cat') change the look of the list in bacon?

Ans. - After calling bacon.remove('cat'), the list will look like this: [3.14, 11, 'cat', True]. The first occurrence of 'cat' is removed.

9. What are the list concatenation and list replication operators?

Ans. - List Concantenation – List Concantenation means combining elements of two lists into the single List. + operator is used for concantenation

Eg. Lst1 = [1, 2, 3]

Lst2 = [4, 5, 6]

Lst3 = Lst1 + Lst2

print(Lst3) // OUPTUT : [1, 2, 3, 4, 5,6]

List Replication - It is a way to make a new list by copying the items in an existing list multiple times. * operator is used for list replication.

Lst = [1, 2, 3]

New_lst = Lst*3

print(New_lst) // OUTPUT : [1, 2, 3, 1, 2, 3, 1, 2, 3]

10. What is difference between the list methods append() and insert()?

Ans. – append() methos add the element at the end of the list whereas you can insert element at any position you want using insert() method.

Eg. my_list = [1, 2, 3]

my_list.append(4)

print(my_list) // OUTPUT : [1, 2, 3, 4]

```
my_list.insert(1, 5)
print(my_list)    // OUTPUT : [1, 5,  2, 3, 4]
```

11. What are the two methods for removing items from a list?

Ans. - remove(): Removes the first occurrence of a specified element from the list.

pop(): Removes and returns the element at a specified index. If no index is specified, it removes and returns the last element in the list.

12. Describe how list values and string values are identical.

Ans. - Both lists and strings are sequence types in Python. This means they are ordered collections of elements, where each element has a specific position (index).

You can access elements in both lists and strings using their index, starting from 0.

13. What's the difference between tuples and lists?

Ans. – The main difference between Lists and Tuples is that lists are mutable whereas tuples are Immutable i.e we cannot chage the tuples once they are made but list can be chaged

We can add or remove element element from list as per our need.

14. How do you type a tuple value that only contains the integer 42?

Ans. - my_tuple = (42,)

15. How do you get a list value's tuple form? How do you get a tuple value's list form?

Ans. – Converting List into tuples

```
my_list = [1, 2, 3]
my_tuple = tuple(my_list)
print(mu_tuple)   // OUTPUT : (1, 2, 3)
```

Converting tuple into list

```
my_tuple = (1, 2, 3)
```

```
my_list = list(my_tuple)
print(my_list)      // OUTPUT : [1, 2, 3]
```

16. Variables that contain list values are not necessarily lists themselves. Instead, what do they contain?

Ans. - Variables that "contain" list values actually hold references to the list objects, not the lists themselves.

17. How do you distinguish between copy.copy() and copy.deepcopy()?

Ans. - copy.copy(): Creates a shallow copy of an object, meaning it copies the top-level object but keeps references to nested objects. Changes to nested objects affect both the original and the copy.

copy.deepcopy(): Creates a deep copy of an object, meaning it duplicates the entire object and all nested objects. Changes to nested objects in the copy do not affect the original object.