PAGE NO:

Assignment:

- How to add many elements at once to a list.
- How to use the delete method.
- How to check if an element is existing in a list.
- Research on tuples.

How to add very elements at once into a list.

- This method is ~~type~~ HOW TO ADD VERY ELEMENTS AT ONCE TO A LIST

1) Using the extend() method:

This method is specifically designed to add elements from an iterable (like another list, tuple, or string) to the end of the existing list.

- It modifies the original list in-place, making it ~~difficult~~ efficient for appending multiple elements.

Python code

```
my_list = [1, 2, 3]
elements_to_add = [4, 5, 6]
my_list.extend(elements_to_add)
print(my_list)
```

Output: [1, 2, 3, 4, 5, 6]

2) Using the (+) operator (concatenation):

This approach creates a new list by combining an existing list with another iterable.

- The original list remains unchanged.

Python Code:

print(new_list)
#output: [1,2,3,4,5,6].
print(my_list) #output: [1,2,3] [original list remains unchanged]

3) Using a loop with append(). [for loop].
- This method iterates through the elements you want to add and appends each one individually using the append() method.
    - It provides more flexibility for specific modifications within a loop.

Python.

```
my_list = [1,2,3]
elements_to_add = [4,5,6].
    for element in elements_to_add.
        my_list.append(element)
print(my_list) # output: [1,2,3,4,5,6]
```

(short notes)

- If you want to efficiently add elements from another iterable to the end of an existing list and modify the original list use extend().
    - If you need to create new list combining elements from multiple sources, use concatenation (+)
    - If you require more

1) Deleting Variables

- To remove a Variable from the current scope (local or global), use `del` followed by the Variable name. This removes the reference to the object and any further attempts to use the Variable name will result in a NameError.

Python Code:

```
X = 10
del X
# print(X) # This will raise a NameError since X is deleted
```

2) Deleting Elements from lists.

- `del` can be used with list indexing to remove specific elements at a particular index or within a slice.
- Important: This modifies the original list in-place.

Python.

```
my_list = [1, 2, 3, 4, 5]
# Delete the element at index 2 (removing the value 3)
del my_list[2].
print(my_list) #output [1, 2, 4, 5].

# Delete a slice (elements from index 1 to index 3, excluding
del my_list [1:3]
print (my_list) # output [1, 5]
```