# TITLE: LIST AND TUPLE

**OBJECTIVE**:

* To understand list and tuple data types in python.

**THEORY**:

List:

A list in python is referred as the collection of items. Different types of data like: integers, strings and even other lists can be stored in list. Lists are used for storing multiple items in a single variable and these are created using square brackets.

1. Syntax: variable\_name=[‘item\_1’, ‘item\_2’, …….]
2. #Example:
3. myList=[‘Bagmati’, ‘Janakpur’, ‘Karnali’, ‘Koshi’]

Notes:

* Lists are ordered, i.e. the items in lists have defined order and the order will not change. Some list methods changes the list order but in general the order of items will not change.
* Lists allows duplicate entries.
* Lists are mutable, i.e. the list items can be modified.

List Operations:

1. Modifying elements in a list: To modify the elements in a list.

1. Syntax: variable\_name[index]=new\_value

2. fruits = ['apple', 'orange', 'banana']

3. fruits[1] = 'kiwi' # Changes the second element from 'orange' to 'kiwi'

1. Adding elements in a list: This adds new elements in the list.

1. Syntax: my\_list.append(new\_element)

2. fruits = ['apple', 'orange', 'banana']

3. fruits.append('grape') # Adds 'grape' to the end of the list

1. Syntax: my\_list.insert(index, new\_element)

2. fruits = ['apple', 'orange', 'banana']

3. fruits.insert(1, 'kiwi') # Inserts 'kiwi' at index 1

1. Removing elements from a list: To remove any element from the list.

1. Syntax: my\_list.remove(element)

2. fruits = ['apple', 'orange', 'banana']

3. fruits.remove('kiwi') # Removes Kiwi from the list

1. Sorting a list permanently using sort() method: Sort the list permanently.
2. sorted\_list = sorted(my\_list)
3. unsorted\_numbers = [3, 1, 4, 1, 5, 9, 2]
4. sorted\_numbers = sorted(unsorted\_numbers) # Creates a sorted copy
5. Sorting a list temporarily using sorted() function: Temporarily sort the list.
6. sorted\_list = sorted(my\_list)
7. unsorted\_numbers = [3, 1, 4, 1, 5, 9, 2]
8. sorted\_numbers = sorted(unsorted\_numbers) # Creates a sorted copy
9. Printing list in reverse order
10. my\_list.reverse()
11. numbers = [1, 2, 3, 4, 5]
12. numbers.reverse() # Reverses the order of the list
13. Finding the length of a list
14. length = len(my\_list)
15. fruits = ['apple', 'orange', 'banana']
16. num\_fruits = len(fruits) # Returns the length of the list
17. Avoiding index errors:
18. index = 5
19. if index < len(my\_list):
20. element = my\_list[index]
21. print(element)
22. else:
23. print("Index out of range.")
24. Using range() to make a list of numbers.
25. my\_list = list(range(start, stop, step))
26. even\_numbers = list(range(2, 11, 2)) # Generates [2, 4, 6, 8, 10]
27. Looping through list
28. for item in my\_list:
29. # Code to process each item
30. fruits = ['apple', 'kiwi', 'orange', 'banana']
31. for fruit in fruits:
32. print(fruit) # Prints each fruit in the list
33. Slicing a list
34. sublist = my\_list[start:end]
35. numbers = [1, 2, 3, 4, 5]
36. subset = numbers[1:4] # Creates a sublist [2, 3, 4]
37. print(subset)
38. Looping through slice
39. for item in my\_list[start:end]:
40. # Code to process each item in the slice
41. numbers = [1, 2, 3, 4, 5]
42. for num in numbers[2:5]:
43. print(num) # Prints elements in the slice [3, 4, 5]
44. Copying a list
45. new\_list = my\_list.copy()
46. fruits = ['apple', 'kiwi', 'orange', 'banana']
47. fruits\_copy = fruits.copy() # Creates a copy of the fruits list
48. print(fruits\_copy)