Name: Syed Mansoor ul Hassan Bukhari

Course: Artificial Intelligence

Assignment: 01

List Methods in Python

In Python, lists are a versatile and widely used data structure that allows you to store a collection of items. Lists are mutable, meaning you can change their contents after they are created. Python provides several built-in methods to manipulate lists efficiently. Below, we'll explore some commonly used list methods, provide a brief explanation for each, and show examples of how to use them.

1. `append()`

Explanation: The `append()` method adds an element to the end of the list.

Example:

```
fruits = ['apple', 'banana', 'cherry']

fruits.append('orange')

print(fruits)

# Output: ['apple', 'banana', 'cherry', 'orange']
```

2. `insert()`

Explanation: The `insert()` method inserts an element at a specified position in the list.

Example:

```
fruits = ['apple', 'banana', 'cherry']

fruits.insert(1, 'orange')  # Insert 'orange' at index 1

print(fruits)

# Output: ['apple', 'orange', 'banana', 'cherry']
```

3. `remove()`

Explanation: The `remove()` method removes the first occurrence of a specified element from the list.

Example:

fruits = ['apple', 'banana', 'cherry', 'banana']

fruits.remove('banana')

print(fruits)

 # Output: ['apple', 'cherry', 'banana']

4. `pop()`

Explanation: The **`pop()`** method removes and returns the element at a specified index. If no index is specified, it removes and returns the last element.

Example:

fruits = ['apple', 'banana', 'cherry']

removed_fruit = fruits.pop(1) # Remove and return the element at index 1

print(removed_fruit) # Output: 'banana'

print(fruits)

 # Output: ['apple', 'cherry']

5. `clear()`

Explanation: The **`clear()`** method removes all elements from the list, resulting in an empty list.

Example:

fruits = ['apple', 'banana', 'cherry']

fruits.clear()

print(fruits)

 # Output: []

6. `index()`

Explanation: The `index()` method returns the index of the first occurrence of a specified element in the list.

Example:

fruits = ['apple', 'banana', 'cherry']

index_of_cherry = fruits.index('cherry')

print(index_of_cherry)

 # Output: 2

7. `count()`

Explanation: The `count()` method returns the number of times a specified element appears in the list.

Example:

fruits = ['apple', 'banana', 'cherry', 'banana']

banana_count = fruits.count('banana')

print(banana_count)

 # Output: 2

8. `sort()`

Explanation: The **`sort()`** method sorts the elements of the list in ascending order by default. You can also sort in descending order by passing **`reverse=True`** as an argument.

Example:

numbers = [3, 1, 4, 2]

numbers.sort()

print(numbers)

 # Output: [1, 2, 3, 4]

numbers.sort(reverse=True)

print(numbers)

 # Output: [4, 3, 2, 1]

9. `reverse()`

Explanation: The `reverse()` method reverses the order of the elements in the list.

Example:

fruits = ['apple', 'banana', 'cherry']

fruits.reverse()

print(fruits)

 # Output: ['cherry', 'banana', 'apple']

10. `copy()`

Explanation: The `copy()` method returns a shallow copy of the list.

Example:

fruits = ['apple', 'banana', 'cherry']

fruits_copy = fruits.copy()

print(fruits_copy)

 # Output: ['apple', 'banana', 'cherry']

11. `extend()`

Explanation: The `extend()` method adds all elements of an iterable (e.g., another list) to the end of the list.

Example:

fruits = ['apple', 'banana', 'cherry']

more_fruits = ['orange', 'grape']

fruits.extend(more_fruits)

print(fruits)

 # Output: ['apple', 'banana', 'cherry', 'orange', 'grape']

These are some of the essential list methods in Python that you will frequently use. Each method serves a specific purpose and helps in managing and manipulating lists effectively. Understanding these methods will make your programming tasks easier and more efficient.