**Python List Methods**

**1. append()**

**Description:**

The append() method adds a single element to the end of the list. It modifies the original list.

**Syntax:**

python

Copy code

list\_name.append(element)

**Parameters:**

* element: The element to be added to the list.

**Return Type:**

* None (modifies the list in place)

**Example:**

python

Copy code

my\_list = [1, 2, 3]

my\_list.append(4)

# my\_list becomes [1, 2, 3, 4]

**2. extend()**

**Description:**

The extend() method adds all elements of an iterable (like a list or a tuple) to the end of the list.

**Syntax:**

python

Copy code

list\_name.extend(iterable)

**Parameters:**

* iterable: An iterable like a list, tuple, set, etc., whose elements are added to the list.

**Return Type:**

* None (modifies the list in place)

**Example:**

python

Copy code

my\_list = [1, 2, 3]

my\_list.extend([4, 5])

# my\_list becomes [1, 2, 3, 4, 5]

**3. insert()**

**Description:**

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

**Syntax:**

python

Copy code

list\_name.insert(index, element)

**Parameters:**

* index: The position at which the element should be inserted.
* element: The element to be inserted.

**Return Type:**

* None (modifies the list in place)

**Example:**

python

Copy code

my\_list = [1, 2, 4]

my\_list.insert(2, 3)

# my\_list becomes [1, 2, 3, 4]

**4. remove()**

**Description:**

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

**Syntax:**

python

Copy code

list\_name.remove(element)

**Parameters:**

* element: The element to be removed.

**Return Type:**

* None (modifies the list in place)

**Example:**

python

Copy code

my\_list = [1, 2, 3, 2, 4]

my\_list.remove(2)

# my\_list becomes [1, 3, 2, 4]

**5. pop()**

**Description:**

The pop() method removes and returns the element at the specified position. If no index is specified, it removes and returns the last element.

**Syntax:**

python

Copy code

list\_name.pop(index)

**Parameters:**

* index (optional): The position of the element to remove. Defaults to the last element if not specified.

**Return Type:**

* The removed element.

**Example:**

python

Copy code

my\_list = [1, 2, 3, 4]

removed\_element = my\_list.pop(2)

# removed\_element is 3, my\_list becomes [1, 2, 4]

**6. clear()**

**Description:**

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

**Syntax:**

python

Copy code

list\_name.clear()

**Parameters:**

* None

**Return Type:**

* None (modifies the list in place)

**Example:**

python

Copy code

my\_list = [1, 2, 3]

my\_list.clear()

# my\_list becomes []

**7. index()**

**Description:**

The index() method returns the index of the first occurrence of a specified element in the list. If the element is not found, it raises a ValueError.

**Syntax:**

python

Copy code

list\_name.index(element, start, end)

**Parameters:**

* element: The element whose index is to be found.
* start (optional): The starting position in the list to search.
* end (optional): The ending position in the list to search.

**Return Type:**

* The index of the element.

**Example:**

python

Copy code

my\_list = [1, 2, 3, 2, 4]

index\_of\_element = my\_list.index(2)

# index\_of\_element is 1

**8. count()**

**Description:**

The count() method returns the number of occurrences of a specified element in the list.

**Syntax:**

python

Copy code

list\_name.count(element)

**Parameters:**

* element: The element to count.

**Return Type:**

* The number of occurrences of the element.

**Example:**

python

Copy code

my\_list = [1, 2, 3, 2, 4]

count\_of\_element = my\_list.count(2)

# count\_of\_element is 2

**9. sort()**

**Description:**

The sort() method sorts the elements of the list in ascending or descending order. By default, it sorts in ascending order.

**Syntax:**

python

Copy code

list\_name.sort(key=None, reverse=False)

**Parameters:**

* key (optional): A function that serves as a key for sorting.
* reverse (optional): If True, the list is sorted in descending order.

**Return Type:**

* None (modifies the list in place)

**Example:**

python

Copy code

my\_list = [4, 2, 3, 1]

my\_list.sort()

# my\_list becomes [1, 2, 3, 4]

**10. reverse()**

**Description:**

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

**Syntax:**

python

Copy code

list\_name.reverse()

**Parameters:**

* None

**Return Type:**

* None (modifies the list in place)

**Example:**

python

Copy code

my\_list = [1, 2, 3, 4]

my\_list.reverse()

# my\_list becomes [4, 3, 2, 1]

**11. copy()**

**Description:**

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

**Syntax:**

python

Copy code

list\_name.copy()

**Parameters:**

* None

**Return Type:**

* A new list that is a shallow copy of the original list.

**Example:**

python

Copy code

my\_list = [1, 2, 3]

new\_list = my\_list.copy()

# new\_list is [1, 2, 3]