## **Python List Methods**

### **1. append()**

* **Description**: Adds a single item to the end of the list.
* **Syntax**: list.append(item)
* **Parameters**: item - The item to be added to the end of the list.
* **Return Type**: None

**Example**:

my\_list = [1, 2, 3]

my\_list.append(4)

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

### **2. extend()**

* **Description**: Extends the list by appending elements from an iterable (e.g., another list).
* **Syntax**: list.extend(iterable)
* **Parameters**: iterable - An iterable whose elements are added to the list.
* **Return Type**: None

my\_list = [1, 2, 3]

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

# Result: [1, 2, 3, 4, 5]

### **3. insert()**

* **Description**: Inserts an item at a specified position in the list.
* **Syntax**: list.insert(index, item)
* **Parameters**:
  + index - The position at which to insert the item.
  + item - The item to be inserted.
* **Return Type**: None

**Example**:

my\_list = [1, 2, 3]

my\_list.insert(1, 'a')

# Result: [1, 'a', 2, 3]

### **4. remove()**

* **Description**: Removes the first occurrence of a specified item from the list.
* **Syntax**: list.remove(item)
* **Parameters**: item - The item to be removed.
* **Return Type**: None

**Example**:

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

my\_list.remove(2)

# Result: [1, 3, 2]

### **5. pop()**

* **Description**: Removes and returns an item at a specified position (or the last item if no index is specified).
* **Syntax**: list.pop([index])
* **Parameters**: index (optional) - The position of the item to be removed. If omitted, the last item is removed.
* **Return Type**: The item that was removed.

**Example**:

my\_list = [1, 2, 3]

popped\_item = my\_list.pop()

# Result: [1, 2, 3] -> [1, 2], popped\_item = 3

### **6. clear()**

* **Description**: Removes all items from the list, leaving it empty.
* **Syntax**: list.clear()
* **Parameters**: None
* **Return Type**: None

**Example**:

my\_list = [1, 2, 3]

my\_list.clear()

# Result: []

### **7. index()**

* **Description**: Returns the index of the first occurrence of a specified item.
* **Syntax**: list.index(item, [start], [end])
* **Parameters**:
  + item - The item to search for.
  + start (optional) - The starting index to search from.
  + end (optional) - The ending index to search up to.
* **Return Type**: int

**Example**:

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

index\_of\_2 = my\_list.index(2)

# Result: index\_of\_2 = 1

### **8. count()**

* **Description**: Returns the number of occurrences of a specified item in the list.
* **Syntax**: list.count(item)
* **Parameters**: item - The item to count.
* **Return Type**: int

**Example**:

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

count\_of\_2 = my\_list.count(2)

# Result: count\_of\_2 = 2

### **9. sort()**

* **Description**: Sorts the items of the list in ascending order.
* **Syntax**: list.sort(key=None, reverse=False)
* **Parameters**:
  + key (optional) - A function to be used for sorting (e.g., lambda x: x).
  + reverse (optional) - If True, the list is sorted in descending order.
* **Return Type**: None

**Example**:

my\_list = [3, 1, 2]

my\_list.sort()

# Result: [1, 2, 3]

### **10. reverse()**

* **Description**: Reverses the order of items in the list.
* **Syntax**: list.reverse()
* **Parameters**: None
* **Return Type**: None

**Example**:

my\_list = [1, 2, 3]

my\_list.reverse()

# Result: [3, 2, 1]

### **11. copy()**

* **Description**: Returns a shallow copy of the list.
* **Syntax**: list.copy()
* **Parameters**: None
* **Return Type**: A new list that is a copy of the original list.

**Example**:

my\_list = [1, 2, 3]

copy\_list = my\_list.copy()

# Result: copy\_list = [1, 2, 3]