Name: Muhammad Haris

Roll no: PIAIC177845

1. append(x)

Description: Adds an element to the end of the list.

Syntax: list.append(x)

Parameters:

- x: The element to be added to the list.

Return Type: None (modifies the original list in place)

Example:

```
my_list = [1, 2, 3]
my_list.append(4)
print(my_list)  # Output: [1, 2, 3, 4]
```

content_copy Use code with caution.Python

2. extend(iterable)

Description: Appends the elements of an iterable (e.g., another list, tuple, string) to the end of the list.

Syntax: list.extend(iterable)

Parameters:

- iterable: An iterable containing elements to be added to the list.

Return Type: None (modifies the original list in place)

Example:

```
my_list = [1, 2, 3]
```

```
my_list.extend([4, 5, 6])

print(my_list)  # Output: [1, 2, 3, 4, 5, 6]
```

content_copy Use code with caution.Python

3. insert(i, x)

Description: Inserts an element at a specific index in the list.

Syntax: list.insert(i, x)

Parameters:

- i: The index at which to insert the element.
- x: The element to be inserted.

Return Type: None (modifies the original list in place)

Example:

```
my_list = [1, 2, 3]

my_list.insert(1, 4)

print(my_list)  # Output: [1, 4, 2, 3]
```

content_copy Use code with caution.Python

4. remove(x)

Description: Removes the first occurrence of a specific element from the list.

Syntax: list.remove(x)

Parameters:

- x: The element to be removed.

Return Type: None (modifies the original list in place)

Example:

```
    my_list = [1, 2, 3, 2]

my_list.remove(2)

print(my_list)  # Output: [1, 3, 2]
```

content_copy Use code with caution.Python

5. pop(i=-1)

Description: Removes and returns the element at the specified index. If no index is provided, removes and returns the last element.

Syntax: list.pop(i)

Parameters:

- i (optional): The index of the element to be removed. Defaults to -1 (last element).

Return Type: The element that was removed from the list.

Example:

```
    my_list = [1, 2, 3]

removed_element = my_list.pop()  # Removes and returns the last element

print(removed_element)  # Output: 3

print(my_list)  # Output: [1, 2]


removed_element = my_list.pop(0)  # Removes and returns the first element

print(removed_element)  # Output: 1

print(my_list)  # Output: [2]
```

content_copy Use code with caution.Python

6. clear()

Description: Removes all elements from the list.

Syntax: list.clear()

Parameters: None

Return Type: None (modifies the original list in place)

Example:

```
my_list = [1, 2, 3]

my_list.clear()

print(my_list)  # Output: []
```

content_copy Use code with caution.Python

7. index(x, start=0, end=None)

Description: Returns the index of the first occurrence of a specific element in the list.

Syntax: list.index(x, start=0, end=None)

Parameters:

- x: The element to search for.
- start (optional): The starting index for the search. Defaults to 0.
- end (optional): The ending index for the search. Defaults to the end of the list.

Return Type: The index of the first occurrence of the element. Raises a ValueError if the element is not found.

Example:

```
my_list = [1, 2, 3, 2]
```

```
index_of_2 = my_list.index(2)

print(index_of_2)  # Output: 1


index_of_2_again = my_list.index(2, 2)

print(index_of_2_again)  # Output: 3
```

content_copy Use code with caution.Python

8. count(x)

Description: Returns the number of occurrences of a specific element in the list.

Syntax: list.count(x)

Parameters:

- x: The element to count.

Return Type: The number of occurrences of the element.

Example:

```
    my_list = [1, 2, 3, 2]

count_of_2 = my_list.count(2)

print(count_of_2)  # Output: 2
```

content_copy Use code with caution.Python

9. sort(reverse=False, key=None)

Description: Sorts the elements of the list in ascending order.

Syntax: list.sort(reverse=False, key=None)

Parameters:

- reverse (optional): If True, sorts the list in descending order. Defaults to False.
- key (optional): A function that specifies a custom sorting criterion.

Return Type: None (modifies the original list in place)

Example:

```
my_list = [3, 1, 4, 2]

my_list.sort()

print(my_list)  # Output: [1, 2, 3, 4]



my_list.sort(reverse=True)

print(my_list)  # Output: [4, 3, 2, 1]
```

content_copy Use code with caution.Python

10. reverse()

Description: Reverses the order of the elements in the list.

Syntax: list.reverse()

Parameters: None

Return Type: None (modifies the original list in place)

Example:

```
my_list = [1, 2, 3]

my_list.reverse()

print(my_list)  # Output: [3, 2, 1]
```

content_copy Use code with caution.Python

11. copy()

Description: Returns a shallow copy of the list. Changes to the original list will not affect the copy, and vice versa.

Syntax: list.copy()

Parameters: None

Return Type: A new list object.

Example:

```
    my_list = [1, 2, 3]
copied_list = my_list.copy()
copied_list.append(4)
print(my_list)  # Output: [1, 2, 3]
print(copied_list)  # Output: [1, 2, 3, 4]
```

content_copy Use code with caution.Python

12. len(list)

Description: Returns the number of elements in the list.

Syntax: len(list)

Parameters:

- list: The list for which you want to find the length.

Return Type: An integer representing the length of the list.

Example:

```
my_list = [1, 2, 3]
list_length = len(my_list)
print(list_length)  # Output: 3
```

content_copy Use code with caution.Python

13. min(list)

Description: Returns the minimum element in the list.

Syntax: min(list)

Parameters:

- list: The list for which you want to find the minimum value.

Return Type: The minimum element in the list.

Example:

```
my_list = [3, 1, 4, 2]
minimum_value = min(my_list)
print(minimum_value)  # Output: 1
```

content_copy Use code with caution.Python

14. max(list)

Description: Returns the maximum element in the list.

Syntax: max(list)

Parameters:

- list: The list for which you want to find the maximum value.

Return Type: The maximum element in the list.

Example:

```
    my_list = [3, 1, 4, 2]

maximum_value = max(my_list)

print(maximum_value)  # Output: 4



content_copy Use code with caution.Python
```

15. sum(list)

Description: Returns the sum of all elements in the list.

Syntax: sum(list)

Parameters:

- list: The list for which you want to calculate the sum.

Return Type: An integer representing the sum of all elements in the list.

Example:

```
    my_list = [1, 2, 3, 4]

total_sum = sum(my_list)

print(total_sum)  # Output: 10



content_copy Use code with caution.Python
```

16. any(list)

Description: Returns True if at least one element in the list is True or evaluates to True. Returns False if all elements are False or the list is empty.

Syntax: any(list)

Parameters:

- list: The list to check for truthiness.

Return Type: A boolean value (True or False).

Example:

```
    my_list = [0, False, "", 1]

has_truthy_value = any(my_list)

print(has_truthy_value)  # Output: True



my_list = [0, False, "", None]

has_truthy_value = any(my_list)

print(has_truthy_value)  # Output: False
```

content_copy Use code with caution.Python

17. all(list)

Description: Returns True if all elements in the list are True or evaluate to True. Returns False if at least one element is False or the list is empty.

Syntax: all(list)

Parameters:

- list: The list to check for truthiness.

Return Type: A boolean value (True or False).

Example:

```
    my_list = [1, True, "hello"]
```

```
are_all_truthy = all(my_list)

print(are_all_truthy)  # Output: True


my_list = [1, True, "", "hello"]

are_all_truthy = all(my_list)

print(are_all_truthy)  # Output: False
```