List

- List items are `ordered`, `changeable`, and `allow duplicate values`.
- List items are indexed.
- Changeable, means that we can change, add, and remove items in a list after it has been created.
- Allow duplicates, means lists can have items with the same value
- Starts with []

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

Methods

append()

Adds an `element` at the `end of the list`.

```
lst = [1, 2, 3]
lst.append(4)
print(lst)
```

extend()

Adds `elements of a list` to the `end of the current list`.

```
lst = [1, 2, 3]
lst.extend([5, 6])
print(lst)
```

insert()

Adds an `element` at a `specified position`.

```
lst = [1, 2, 3]
lst.insert(1, 'a')
print(lst)
```

remove()

Removes the `first occurrence` of the element with the specified value.

```
lst = [1, 2, 3, 'a']
lst.remove('a')
print(lst)
```

pop()

Removes the element at the `specified position`, or the `last item`.

```
lst = [1, 2, 3]
lst.pop()
print(lst)
```

clear()

Removes `all the elements` from the list.

```
lst = [1, 2, 3]
lst.clear()
print(lst)
```

reverse()

`Reverses the order` of the list.

```
lst = [1, 2, 3, 'a']
lst.reverse()
print(lst)
```

len()

Returns `number of items` in sequence. Similar to that of tuple

```
lst = [1, 2, 3]
print(len(lst))
```

min()

Returns the `minimum value` in sequence. Similar to that of tuple

```
lst = [1, 2, 3]
print(min(lst))
```

max()

- Returns the `maximum value` in sequence. Similar to that of tuple

```
lst = [1, 2, 3]
print(max(lst))
```

in

- Finds if the element `is in the list`. Similar to that of tuple

```
lst = [1, 2, 3]
print(1 in lst)
print(6 not in lst)
```

Slicing

- Slicing `[start:stop:step]` Similar to that of tuple
- **stop is exclusive**, **start defaults to 0**, **step defaults to 1**
- If start > stop or start > len(), empty list is returned

```
lst = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
print(lst[1:4])
print(lst[3:])
print(lst[:3])
print(lst[::2])
print(lst[1:5:2])
print(lst[0:6:-1])
print(lst[::-1])
```

List in a List

Similar to that of tuple

```
lst = [1, 2, 3, (4, 5), [6, 7], 8, 9]
print(lst[2])
print(lst[-2])
print(lst[3])
print(lst[4])

lst1 = [1, 2, 3, 4, 5, 6, 7, 8]
lst2 = [1, [2, 3, 4], [5, 6, 7], [8,]]
lst3 = [1, [2, 3, (4,), 5], [6, 7, 8]]

print(4 in tup1)
print(4 in tup2)
print(4 in tup3)
```

Looping

You can iterate through a list in 2 ways Similar to that of tuple

1st Way

```
lst = [3, 2, 1, 5, 7]
for i in lst:
        print(i)
```

2nd Way

```
lst = [3, 2, 1, 5, 7]
for i in range(len(lst)):
        print(i, ":", lst[i])
```