



Subrata Saha

# LIST



# List

- built-in data types in Python used to store collections of data
- Lists are used to store multiple items in a single variable.
- list is created by placing all the items (elements) inside square brackets `[]`, separated by commas.
- List items are ordered, changeable, and allow duplicate values.
- It can grows and shrink dynamically

# List

- It can have any number of items and they may be of different types (integer, float, string etc.).
- Empty list  
`my_list = []`
- List of integers  
`my_list = [1, 2, 3]`
- List with mixed data types  
`my_list = [1, "Hello", 3.4]`
- List can be created using constructor  
`my_list = list()`

# List Index

- The index operator [ ] is used to access an item in a list.
- Index value starts from 0.
- Python allows negative indexing for its sequences. The index of -1 refers to the last item, -2 to the second last item and so on.
- `mylist=[2,4,6,8,10]`
- `print(mylist[0])` Output: 2
- `print(mylist[2])` 6
- `print(mylist[-1])` 10
- `print(mylist[-3])` 6

# Nested list

- A list can also have another list as an item. This is called a nested list.

```
my_list = ["Techno", [8, 4, 6], [2.5, 3.8]]
```

- `mylist=["Techno",[2,4,6,8]]`

```
print(mylist[0])
```

Output: Techno

```
print(mylist[1])
```

[2, 4, 6, 8]

```
print(mylist[0][2])
```

C

```
print(mylist[1][2])
```

6

# Slicing in list

- `my_list = [1,2,3,4,5,6,7,8]`
  - `print(my_list[2:5])`
  - `print(my_list[:5])`
  - `print(my_list[5:])`
  - `print(my_list[:])`
- Output:
- `[3, 4, 5]`
  - `[1, 2, 3, 4, 5]`
  - `[6, 7, 8]`
  - `[1, 2, 3, 4, 5, 6, 7, 8]`
- 
- `print(my_list[:-5])`
  - `print(my_list[-5:])`
- `[1, 2, 3]`  
`[4, 5, 6, 7, 8]`
- 
- `print(my_list[-2:-5:-1])`
  - `print(my_list[::-1])`
- `[7, 6, 5]`  
`[8, 7, 6, 5, 4, 3, 2, 1]`

# Modify List Elements

- `even = [2, 4, 6, 8, 10]`

- `even[0] = 12`

- `print(even)`

Output:

`[12, 4, 6, 8, 10]`

- `even[1:4] = [14, 16, 18]`

- `print(even)`

`[12, 14, 16, 18, 10]`

# Concatenate(+) and repeat(\*)

- `odd = [1, 3, 5]`  
`odd1 = odd + [7, 9]`  
`print(odd1)`  
`odd2 = odd * 3`  
`print(odd2)`

Output:

`[1, 3, 5, 7, 9]`

`[1, 3, 5, 1, 3, 5, 1, 3, 5]`



# Add new item

- To add single item at the end of a list **append()** method is used whereas to add several items **extend()** method is used.

- `odd = [1, 3, 5]`

```
odd.append(7)
```

```
print(odd)
```

Output:

```
[1, 3, 5, 7]
```

- Not only integer we can add any object even any list

```
odd.extend([9, 11, 13])
```

```
print(odd)
```

```
[1, 3, 5, 7, 9, 11, 13]
```

# Insert()

- Inserts element at the desired position.
- Syntax:

`insert(position, value)`

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

`my_list.insert(2,5)`

`print(my_list)`

Output:

`[1, 2, 5, 3, 4]`

- `my_list.insert(0,7)`

`print(my_list)`

`[7, 1, 2, 5, 3, 4]`

- `my_list.insert(-2,9)`

`print(my_list)`

`[7, 1, 2, 5, 9, 3, 4]`

# Delete Elements from List

- `my_list = [1,2,3,4,5,6]`
- `# delete one item`  
`del my_list[2]`  
`print(my_list)`  
Output: `[1, 2, 4, 5, 6]`
- `# delete multiple items`  
`del my_list[2:4]`  
`print(my_list)`  
`[1, 2, 6]`
- `# delete entire list`  
`del my_list`  
`print(my_list)`  
Error: List not defined

# Delete Elements from List

- **remove()** removes the first matching value/object. It does not do anything with the indexing.
- `myList = [1, 2, 3, 2]`  
`myList.remove(2)`  
`print(myList)`  
Output:  
`[1, 3, 2]`
- **pop()** removes the item from end or from a specific index and returns it.
- `my_list = [1,2,3,4]`
- `x=my_list.pop()`
- `print(x)`  
4
- `x=my_list.pop(1)`
- `print(x)`  
2
- `print(my_list)`  
`[1, 3]`

# Delete Elements from List

- `clear()` method removes all elements from a list.
- ```
my_list = [1,2,3,4]
```

```
my_list.clear()
```

```
print(my_list)
```

Output:  
[]
- Items can be deleted from list by assigning an empty list to a slice of elements.
- ```
my_list = [1,2,3,4,5,6]
```

```
my_list[2:5] = []
```

```
print(my_list)
```

[1, 2, 6]

# Loop Through a List

- `my_list = [1,2,3,4]`
- `for i in my_list:`  
    `print(i)`

Output:

1  
2  
3  
4

- `for i in range(len(my_list)):`  
    `print(my_list[i])`

1  
2  
3  
4

# List functions

Method	Description
max()	Returns highest element from a list
min()	Returns smallest element from a list
sum()	Sums up the numbers in the list
all()	Returns true if all element are true or if list is empty
any()	return true if any element of the list is true. if list is empty, return false
sorted()	Returns a new sorted list, original list is not sorted.
list()	Converts an sequence to a list

# List Methods

Method	Description
copy()	Returns a copy of the list
count()	Returns the number of elements with the specified value
index()	Returns the index of the first element with the specified value
reverse()	Reverses the order of the list
sort()	Sorts the list. To sort in descending order, pass argument <b>reverse=True</b>