

PYTHON LIST  
CLASS XI  
(MODULE-2)  
BY

Mrs. SUJATA PRADHAN,  
PGT(SS), Computer Science,  
AECS, ANUPURAM



**Lists & Operations on list:**

**Lists**

List is an ordered sequence of items. Values in the list are called elements / items. It can be written as a list of comma-separated items (values) between **square brackets [ ]**. Items in

the list can be of different data types  
and List is mutable.

## Operations on list:



Indexing



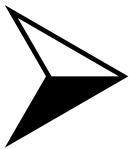
Slicing



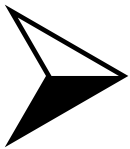
Concatenation



Repetitions



Updating



Membership



Comparison

- 

Indexing

- Slicing

- 

## **Concatenation & Repetition**

**Concatenation** is done by + operator.

Concatenation is supported by sequence data types(string, list, tuple).

Concatenation is done between the **same data types** only.

### **Concatenating two list objects**

```
l1=[1,2]
```

```
l2=[3,4]
```

```
print (l1+l2)
```

Output:

[1, 2, 3, 4]

**Repetition** :Sequences datatypes support a **repetition** operator \*. The repetition operator \* will make multiple copies of that particular object and combines them together. When \* is used with an integer it performs multiplication but with list, tuple or strings it performs a repetition.

### **Repetition operator on List**

```
l1=[1,2,3]
```

```
print (l1 * 3)
```

Output:

```
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

The concatenation and repetition operators are supported only by **sequence datatypes** . Both concatenation and repetition always

result in a **new object**. Concatenation is done only between the same datatypes .

- **Membership** Operation

**Membership** Operators are the operators, which are used to check whether a value/variable exists in the sequence. This operator returns either True or False, if a value/variable is found in the **list**, it returns True otherwise it returns False.

```
x = ["apple", "banana"]
```

```
print("banana" in x)
```

```
# returns True because a sequence with the  
value "banana" is in the list
```

```
x = ["apple", "banana"]
```

```
print("pineapple" not in x)
```

```
# returns True because a sequence with the  
value "pineapple" is not in the list
```



## Traversing a List

We can access each element of the list or traverse a list using a **for** loop or a **while** loop.

- **List using While loop**

The **while** loop in Python is used to iterate over a block of code as long as the test expression (condition) is

**Syntax:**

**while (condition):**

**body of while**



## Basic List Operations

- Updating List values
- The list elements can also be deleted by using the **del** keyword. Python also provides us the **remove()** method if we do not know which element is to be deleted from the list.
- `list = [1, 2, 3, 4, 5, 6]`
- `print(list)`
- `# It will assign value to the value to the second in dex`
- `list[2] = 10`
- `print(list)`
- `# Adding multiple-element`
- `list[1:3] = [89, 78]`
- `print(list)`
- `# It will add value at the end of the list`
- `list[-1] = 25`
- `print(list)`

- Deletion of list values
- List operations
- SUMMARY

## **Operations on list**

- ❖ **Concatenation & Repetition**
- ❖ **Membership Operation**
- ❖ **List traversal**
- ❖ **List updation**
- ❖ **List comparison**

# Thank You