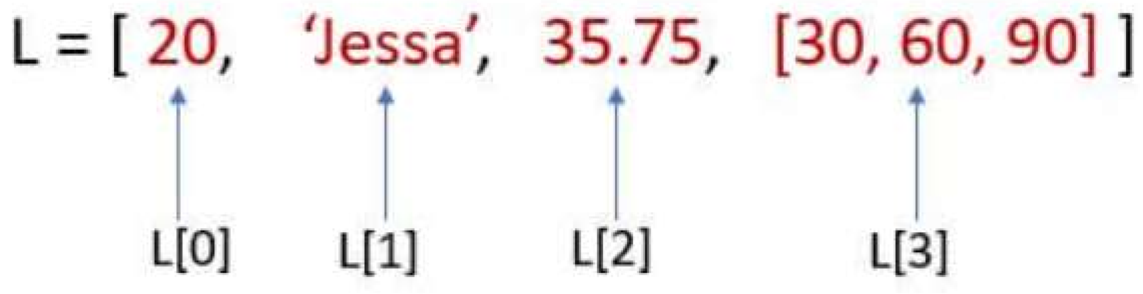
1. Lists

- What are Lists?
- Lists Vs Arrays
- Characterstics of a List
- How to create a list
- Access items from a List
- Editing items in a List
- Deleting items from a List
- Operations on Lists
- Functions on Lists

What are Lists

List is a data type where you can store multiple items under 1 name. More technically, lists act like dynamic arrays which means you can add more items on the fly.



- Why Lists are required in programming?

Array Vs Lists

- Fixed Vs Dynamic Size
- Convenience -> Hetrogeneous
- Speed of Execution
- Memory

```
L=[1,2,3]
print(id(L))
print(id(0))
print(id(1))
print(id(2))
```

```
139720416723104
11126656
11126688
11126720
```

How lists are stored in memory

Characterstics of a List

- Ordered
- Changeble/Mutable
- Hetrogeneous
- Can have duplicates
- are dynamic
- can be nested
- items can be accessed
- can contain any kind of objects in python

```
L=[1,2,3] #mutable means you can change the any time #
L1=[3,2,1]
L==L1 # kisi bi type ka object ko list me add kr shkte he
```

```
False
```

Creating a List

```
# Empty list
print([])
# 1D list  - Also homogenous list
print([1,2,3,4,5])
# 2D iske undar or ak list hotya he
print([1,2,3,[4,5]]) #hetrogeniuos
# 3D
print([[[1,2],[3,4]],[[5,6],[7,8]]]) # homogenious list
# Hetrogenous
print([1,True,5.6,5+6j,'Hello'])
# Using Type conversion
print(list('hello'))
```

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

```
[[[1, 2], [3, 4]], [[5, 6], [7, 8]]]
[1, True, 5.6, (5+6j), 'Hello']
['h', 'e', 'l', 'l', 'o']
```

Accessing Items from a List

```
# Indexing
L=[1,2,3,4]
print(L[0])# poistive indexing ==> go to left se right and start zero se
print(L[-2]) # 2d ke time 2 barket lagana


# Slicing
L=[1,2,3,4,5,6]
print(L[0:3])
print(L[-3:])
print(L[0::2]) # this work also do negativ indexing
print(L[::-1])
```

```
1
3
[1, 2, 3]
[4, 5, 6]
[1, 3, 5]
[6, 5, 4, 3, 2, 1]
```

Adding Items to a List

```
# append
L=[1,2,3,4,5]
#akbar me ak item ko list ke end me add kr skte ho
L.append(True)
print(L)
```

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

```
# extend == Ak sat multiple item ko add kr skte ho
L=[1,2,3,4,4,5]
L.extend([6,4,5])
print(L)
```

```
[1, 2, 3, 4, 4, 5, 6, 4, 5]
```

```
# insert 1 and 2 ke bich me 100 add krna he to insert function use kr skte ho
L=[1,2,3,4,5]
L.insert(6,100)
```

```
print(L)

    [1, 2, 3, 4, 5, 100]
```

Editing items in a List

Deleting items from a List

```
# del
```

```
# remove
```

```
# pop
```

```
# clear
```

Operations on Lists

- Arithmetic
- Membership

[ ] ↳ *3 cells hidden*

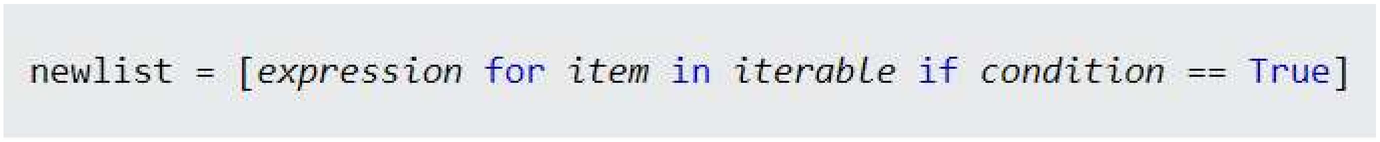
List Functions

[ ] ↳ *6 cells hidden*

List Comprehension

List Comprehension provides a concise way of creating lists.

newlist = [expression for item in iterable if condition == True]



```
newlist = [expression for item in iterable if condition == True]
```

Advantages of List Comprehension

- More time-efficient and space-efficient than loops.
- Require fewer lines of code.
- Transforms iterative statement into a formula.

[ ] ↳ *8 cells hidden*

2 ways to traverse a list

- itemwise
- indexwise

Zip

The zip() function returns a zip object, which is an iterator of tuples where the first item in each passed iterator is paired together, and then the second item in each passed iterator are paired together.

If the passed iterators have different lengths, the iterator with the least items decides the length of the new iterator.

```
# Write a program to add items of 2 lists indexwise

L1 = [1,2,3,4]
L2 = [-1,-2,-3,-4]
```

Disadvantages of Python Lists

- Slow
- Risky usage
- eats up more memory

List Programs

[ ] ↳ *7 cells hidden*