Day-2

Array-/List

→ collection of items stored at contigous memory locations.

Note:* Python doesn't have built-in support for Array.

→ We use list instead of it

→ Python uses '0' based indexing.

* Lists can be created using list() function or using to square brackets → variable_name = [ ]

→ In python, we need to import array module to use arrays :— as→ import array as arr

IN PYTHON

List	VS	Array
→ can have differen data type of elements.		→ Same data type elements only.
→ No need to import any module.		→ have to import a module called 'array'
→ More flexibility in modification of data.		→ Less flexibility.

→ prefered for shorted sequence of data items.

→ For longer sequence of data items.

→ consume larger memory for easy addition of elements.

→ comparitively more compact in memory size.

List data type Methods :-

→ append() - add an item to the end.

→ insert() - insert an iten in a particular pos.

→ remove() - to remove a particular value.

→ pop() - removes particular item and return (if value passed); else, goes to the last ite

→ remove() - removes all the items from the list; as well as del l[:]

→ index() - to find the index of a particular value which is in first in list.

→ count() - to the repeative times of a particular value.

→ sort() - sort the items of a list.

→ reverse() - reverse the elements as opposite to 'sort()'.

→ copy() - return the shallow copy of the list.

List Comprehension:-

→ provide a concise way to create lists.

→ It is an elegant way to create new lists.

a basic of list comprehension is:-

l_comp = [expression for var in old_list if(condition)]

WHAT IF we try to find squares:-

```
squeres = [ ]
for num in range(5):
    squeres.append (num**2)
```

O/P:- [0, 1, 4, 9, 16]

WHAT IF we use list comprehension:-

```
squeres = [num**2 for num in range(5)]
```

O/P:- [0, 1, 4, 9, 16]

* if we wish, we can add a condition statement also in it.

Here, to find ~~only~~ even number from 1-10.

```
evens = [num for num in range(1,11) if num%2==0]
```

O/P:- [2, 4, 6, 8, 10] //