Lists - 1D :

class __ :

def func (self , A, B):

    ___ {write code}

    ___

    ___

⇒ **Announcement :**

⇒ A problem solving session on sunday 11 am (optional)

⇒ **Topics :**

→ Intro to Lists
→ Indexing
→ Append
→ Insert
→ -ve indexing
→ Operations
→ Problem solving

Intro to Lists:

→ Class Teacher :

→ Avg. no. of students in a class : 50

i) 50 registers for 50 students
ii) 1 register for 50 students ✓✓

Advantages of option 2:

i) less resource
ii) Easy to maintain
iii) Easy to analyse
iv) Perform operatn easily

⇒ Sachin :

Total matches = 463

i) addtn ii) Avg runs

runs 1 = 100
runs 2 = 50
.
.
.
runs 463 = 200

$$avg = \frac{runs1 + runs2 + \dots + runs463}{463}$$

★ Lists :

List definition done :

runs = [100, 63, 200, 150, 100, 99, 98, 99]

name of list ⇒ runs

type(runs) ⇒ list

★ Indexing :

S.no.	Roll . no.
0	1
1	2
2	3
3	4
4	5
5	6
6	7

runs = [50, 20, 31, 150, 200]

index = 0 1 2 3 4

size of runs ⇒ 5

index of last element ⇒ 4 ⇒ 5-1

⇒ N-1

⇒ Access the element of a list:

⇒ runs[index]

★ -ve indexing :

⇒ topper from first ⇒ ~~1~~ 0
topper from last ⇒ -1
second topper from last ⇒ -2

+ve index ⇒ 0 1 2 3 4 5 6

runs = [0, 99, 0, 99, 200, 150, 163]

-ve index : -7 -6 -5 -4 -3 -2 -1

len(runs) ⇒ 7

+ve index ⇒ [0, 7) 7 excluded
-ve index ⇒ [-1, -7] all inclusive

⇒ if size of list is N. ⇒ 7

range(N) ⇒ 0, 1, 2, 3, 4, 5, 6

★ Updating a list :

runs = [1, 50, 150, 200]

centurian = 250

Two ways of adding :

i) Adding at last of list (append)
ii) Adding at an index (insert)

⇒ runs.append(centurian)

⇒ runs : [1, 50, 150, 200, 250]

⇒ This power of list that it can be updated is known as **Mutability**

★ Insert : insert(index, data)

index: 0 1 2 3

l = [6, 8, 9, 7]

⇒ l.insert(1, 33)

=> l = [6, 33, 8, 9, 7]

index : 0 1 2 3 4