

## ✓ Lists 1 - Introduction

---

### ✓ Content

1. Motivation behind lists
  2. How to initialize lists?
  3. Indexing lists
  4. Negative indexing
  5. List methods
  6. Iterating over lists
- 

### ✓ Motivation behind Lists

- A python function can have values. Those values have datatypes and can be stored in variables.
- Let's say we have to write a program that stores all the runs made by **Virat Kohli** in all International matches and find out the **minimum**, **maximum**, and **average**.
- Let's say the total number of matches Virat Kohli has played is 370.
- When we actually start writing a program for the problem above, we would have to define 370 variables. That is such a tedious task.
- Here is where **lists** come to our rescue.

values → 4, 4.5, "string", True ...  
(data types) int float string Real ...  
variables → a b c d ...

problem →  
write a program that stores all runs  
made by Virat Kohli in all international  
matches & find out min, max, avg ...

total matches → 370

```
runs1 = 57  
runs2 = 64  
runs3 = 12  
.  
.  
.  
.  
.  
runs370 = 122
```

370 variables  
would be  
very  
difficult to  
manage.

### ✓ Lists

- List is an **ordered** collection of data.

- It is a data structure that can store **multiple** values.
- List have no limit on how many values it can store.
- Creating a list -> [] squared brackets
- They store **comma separated** values.

```
runs_virat = [67, 54, 12, 34, 77, 89, 101]
```

```
runs_virat
```

```
[67, 54, 12, 34, 77, 89, 101]
```

#### ✓ How do I access a value from this list?

- Lists are accessed using indexes.

```
runs_virat[0]
```

```
67
```

```
runs_virat[1]
```

```
54
```

#### ✓ How do I find out the count of total elements in a list?

- Using the `len()` function.

```
len_runs_virat = len(runs_virat)
```

```
len_runs_virat
```

```
7
```

#### ✓ How do I calculate the runs scored by Virat in last match?

```
runs_virat[len(runs_virat) - 1]
```

```
101
```

Python makes it simpler by using **negative indexing**.

```
runs_virat[-1]
```

```
101
```

#### ✓ What does `runs_virat[-len(runs_virat)]` give?

```
runs_virat[-len(runs_virat)] # runs_virat[0]
```

```
67
```



## ✓ Question-1

Given the list `runs = [45, 67, 89, 12, 34, 56, 100]`, print the total runs on odd positions in the list.

**Motivation:** To explain the difference between positions and indexes.

```
runs = [45, 67, 89, 12, 34, 56, 100]
```

```
print(runs[0] + runs[2] + runs[4] + runs[6])
```

268

---

## ✓ List Methods

Methods to add to a list

`[1, 2, 4, 5, 6]` → add an element at the end.  
`list.append(element)`

`[1, 2, 4, 5]` → add an element at index 1  
shift values ahead.  
`list.insert(1, element)`

`[1, 2, 2]` → add multiple values at the end.  
`list.extend([4, 5, 6, 7...])`  
list of values.

## ✓ Can we add more elements to the list?

- Yes, using the `append()` method.

```
# I want to add more runs to this.  
runs = [45, 67, 89, 12, 34, 56, 100]
```

```
runs.append(71)
```

```
runs
```

```
[45, 67, 89, 12, 34, 56, 100, 71]
```

## ✓ Can I add an element at a particular index?

- Yes, using the `insert()` method.

```
# I want to store an element at the zero-th index.  
runs.insert(0, 25)
```

# Every value is moved ahead by 1-space.

runs

```
[25, 45, 67, 89, 12, 34, 56, 100, 71]
```

✓ Let's say we have a variable `new_runs = [130, 69, 92]` and I want to add this to the end of the `runs` list.

- Adding multiple values at once can be done using the `extend()` method.

```
new_runs = [130, 69, 92]
```

```
runs.extend(new_runs)
```

runs

```
[25, 45, 67, 89, 12, 34, 56, 100, 71, 130, 69, 92]
```

- We can also use `+` instead of `extend()`.

```
runs = runs + [200, 250, 300]
```

runs

```
[25, 45, 67, 89, 12, 34, 56, 100, 71, 130, 69, 92, 200, 250, 300]
```

---

## ✓ Iterating over Lists

- `i` -> iterator -> variable used to iterate
- `range(5)` -> iterable -> the collection of values on which we iterate
- `print(i)` -> iteration block -> body of the for loop

```
for i in range(5):  
    print(i)
```

```
0  
1  
2  
3  
4
```

- Lists are also iterable.

```
a = [56, 78, 89, 32, 101, 4]
```

```
for i in a:  
    print(i)
```

```
56  
78  
89  
32  
101  
4
```

---

## ✓ Quiz-1

What will be the output of the following code?

```
my_list = [1, 2, 3, 4, 5]  
i = -1  
while i >= -5:  
    print(my_list[i], end = " ")  
    i -= 1
```

A. 5 4 3 2 1

B. 1 2 3 4 5

C. 5 3 1 2 4

D. 5 4 3 1 2

**Answer:** A

---

## ✓ Question-2

In runs\_virat -> Calculate the sum, average, max and min.

```
runs_virat
    [67, 54, 12, 34, 77, 89, 101]

# Initializing Variables

length_runs = 0
average_runs = 0
sum_runs = 0
max_runs = 0
min_runs = 0

for i in runs_virat:
    sum_runs = sum_runs + i
    length_runs = length_runs + 1

# Length of the list -
length_runs

    7

# Total runs scored -
sum_runs

    434

# Calculating the average runs scored -
average_runs = sum_runs / length_runs
average_runs

    62.0

# Calculating the minimum runs scored -
min_runs = max_runs
for i in a:
    if i < min_runs:
        min_runs = i
min_runs

    4

# Calculating the maximum runs scored -
for i in a:
    if i > max_runs:
        max_runs = i
max_runs

    101
```

✓ We can also use the in-built python functions.

```
sum_runs = sum(runs_virat)
sum_runs

    434

average_runs = sum_runs / len(runs_virat)
average_runs
```

62.0

max(runs\_virat)

101

min(runs\_virat)

12

---

## ✓ Quiz-2

What is the output of the following code?

```
my_list = ["apple", "banana", "orange"]
for i in range(len(my_list)):
    print(f"{my_list[i]} is a fruit.", end = " ")
```

- A. fruit. fruit. fruit.
- B. apple banana orange
- C. is a fruit. is a fruit. is a fruit.
- D. apple is a fruit. banana is a fruit. orange is a fruit.

**Answer:** D

---

## ✓ Question-3

Calculate the sum of runs made by Virat Kohli in all matches with even index.

```
# This will give us all the indices.
for i in range(len(runs_virat)):
    print(i)

0
1
2
3
4
5
6

sum_even_runs = 0
for i in range(len(runs_virat)):
    if i % 2 == 0:
        sum_even_runs = sum_even_runs + runs_virat[i]
sum_even_runs

257
```

---

## ✓ Quiz-3

What is the output of the following code?

```
my_list = [10, 20, 30, 40, 50]
i = 0
while i < len(my_list):
    my_list[i] *= 2
    i += 1
print(my_list)
```

- A. [10, 20, 30, 40, 50]

- B. [1, 2, 3, 4, 5]
- C. [20, 40, 60, 80, 100]
- D. [5, 10, 15, 20, 25]

**Answer:** C

---