

# Creating and Accessing a List

|             |  |          |  |
|-------------|--|----------|--|
| Name        |  | Date     |  |
| Class / Sec |  | Roll no. |  |

## 1 · AIM

To write a Python program that creates a **list**, accesses its elements by **index**, and uses Python's built-in functions to find the length, largest, smallest and average of the values it holds.

## 2 · SOFTWARE AND LIBRARIES USED

| TOOL             | WHAT IT IS FOR                                                                 |
|------------------|--------------------------------------------------------------------------------|
| Python 3         | The programming language the program is written in                             |
| Jupyter Notebook | The environment used to write and run the code (IDLE or Colab also acceptable) |
| No extra library | Lists are built into Python — nothing to import and nothing to install         |

## 3 · THEORY

A **list** stores several values in a single variable, written inside square brackets and separated by commas. Each value has a position number called its **index**, and Python always counts from **0**, not 1.

| marks[0] | marks[1] | marks[2] | marks[3] | marks[4] | marks[5]         |
|----------|----------|----------|----------|----------|------------------|
| 78       | 65       | 88       | 54       | 91       | 72               |
| first    |          |          |          |          | last · also [-1] |

| FUNCTION   | WHAT IT RETURNS    | ON THIS LIST |
|------------|--------------------|--------------|
| len(marks) | How many items     | 6            |
| max(marks) | The largest value  | 91           |
| min(marks) | The smallest value | 54           |
| sum(marks) | All values added   | 448          |

**Counting starts at 0:** A list of **6** items has indexes **0 to 5**. There is no marks[6] — asking for it raises an **IndexError**. This is the single commonest mistake beginners make with lists.

## 4 · PROGRAM

```
# Create a list of marks
marks = [78, 65, 88, 54, 91, 72]

print("The list      :", marks)
print("Number of items:", len(marks))

# Access individual elements by index
print("\nFirst mark (index 0) :", marks[0])
print("Fourth mark (index 3) :", marks[3])
print("Last mark (index -1):", marks[-1])

# Take a slice - a section of the list
print("\nItems 1 to 3 :", marks[1:4])
```

```
# Built-in functions
print("\nHighest :", max(marks))
print("Lowest  :", min(marks))
print("Total   :", sum(marks))
print("Average :", round(sum(marks) / len(marks), 2))
```

**Note on the slice:** `marks[1:4]` gives items 1, 2 and 3 — it **stops before** index 4. The first number is included, the second is not. Getting this wrong is a classic off-by-one error.

## 5 • EXPECTED OUTPUT

The list : [78, 65, 88, 54, 91, 72]  
Number of items: 6

First mark (index 0) : 78  
Fourth mark (index 3) : 54  
Last mark (index -1): 72

Items 1 to 3 : [65, 88, 54]

Highest : 91  
Lowest : 54  
Total : 448  
Average : 74.67

[ PASTE SCREENSHOT HERE ]

*Paste a screenshot of YOUR program running, with the code and output visible together*

## 6 • CHECKING THE OUTPUT

Every result can be worked out by hand. Show the arithmetic in your file.

| RESULT                           | WORKING                                                                            |
|----------------------------------|------------------------------------------------------------------------------------|
| <b>len = 6</b>                   | Six values in the list, so the valid indexes are <b>0 to 5</b>                     |
| <b>marks[3] = 54</b>             | Counting from 0: 78 is [0], 65 is [1], 88 is [2], <b>54 is [3]</b>                 |
| <b>marks[-1] = 72</b>            | Negative indexes count <b>backwards</b> from the end. [-1] is always the last item |
| <b>marks[1:4] = [65, 88, 54]</b> | Items 1, 2 and 3. It <b>stops before</b> 4 — three items, not four                 |
| <b>Total = 448</b>               | 78 + 65 + 88 + 54 + 91 + 72 = <b>448</b>                                           |
| <b>Average = 74.67</b>           | 448 ÷ 6 = 74.666... rounded to <b>74.67</b> by round(..., 2)                       |

**Try this deliberately:** Add a line asking for `marks[6]` and run it. Python raises **IndexError: list index out of range**. Seeing the error once is the fastest way to remember that six items means indexes 0 to 5.

## 7 • OBSERVATION AND RESULT

Two or three sentences in your own words. Say what a list is and what you learned about indexing.

---



---



---

**Example observation:** A list stores several values in one variable, and each value has an index starting at 0 — so this list of 6 marks has indexes 0 to 5. Negative indexing gave the last item without needing to know the length. The built-in functions `len()`, `max()`, `min()` and `sum()` worked directly on the list, and the average of 74.67 matched the hand calculation of 448 divided by 6.

**Worth knowing:** Python has no built-in **average()** function, which is why the program divides `sum` by `len`. NumPy provides **`np.mean()`** instead — that is what Program 4 uses.

## 8 · VIVA QUESTIONS FOR THIS PROGRAM

| LIKELY QUESTION                                           | SHAPE OF A STRONG ANSWER                                                                       |
|-----------------------------------------------------------|------------------------------------------------------------------------------------------------|
| What index does the first item of a list have?            | <b>0</b> . Python counts from zero, so a list of 6 items has indexes 0 to 5                    |
| What does <code>marks[-1]</code> give you?                | The <b>last</b> item. Negative indexes count backwards from the end                            |
| Why does <code>marks[1:4]</code> return only three items? | A slice <b>stops before</b> the second number. It gives indexes 1, 2 and 3                     |
| What happens if you ask for <code>marks[6]</code> ?       | Python raises an <b><code>IndexError</code></b> — there is no index 6 in a list of six items   |
| How did you find the average?                             | <b><code>sum(marks) divided by len(marks)</code></b> . Python has no built-in average function |

## 9 · HOW THIS ENTRY IS MARKED

### FULL MARKS

**Positive, negative and slice** indexing all demonstrated · all four built-in functions used · commented code · a screenshot of **your own** output · an observation explaining that indexing starts at **0**

### LOSES MARKS

Only prints the list and stops · no screenshot, or someone else's · says the first item is at index **1** · thinks `marks[1:4]` returns four items · cannot say what an `IndexError` is

**Before you submit:** Use **your own** numbers — your marks, or your family's ages. Keep at least six values so the slicing is worth showing, and work the average out by hand to check it matches what Python printed.