TOPSTechnologies

Collections, functions and Modules

Presented for :

TOPs Technologies

Presented by :

Sumit B Yadav

Que 1

A list is one of the most commonly used data structures in Python, allowing you to store multiple items in a single variable.

Creating a List

You can create a list by placing elements inside square brackets [], separated by commas.

```
# Examples of lists
empty_list = []  # An empty list
numbers = [1, 2, 3, 4, 5]  # A list of integers
fruits = ["apple", "banana", "cherry"]  # A list of strings
mixed_list = [1, "hello", 3.14, True]  # A mixed list with different types
```

Accessing Elements in a List

You can access list elements using their index. Python uses zero-based indexing, so the first element is at index 0.

```
fruits = ["apple", "banana", "cherry"]

# Access elements by index
print(fruits[0])  # Output: apple
print(fruits[1])  # Output: banana
print(fruits[2])  # Output: cherry
```

Accessing with Negative Indexing

You can use negative indices to access elements from the end of the list.

```
# Negative indexing
print(fruits[-1])  # Output: cherry (last element)
print(fruits[-2])  # Output: banana
print(fruits[-3])  # Output: apple (first element)
```

Que. 2

Indexing in Lists

Indexing in Python lists allows you to access individual elements. Python supports positive indexing (starting from the beginning) and negative indexing (starting from the end).

Positive Indexing

Positive indexing starts from 0 for the first element and increments by 1 for each subsequent element.

```
# A list of colors
colors = ["red", "green", "blue", "yellow"]

# Access elements using positive indices
print(colors[0])  # Output: red (1st element)
print(colors[1])  # Output: green (2nd element)
print(colors[2])  # Output: blue (3rd element)
print(colors[3])  # Output: yellow (4th element)
```

Negative Indexing

Negative indexing starts from -1 for the last element and decreases as you move toward the beginning.

Que. 3

list slicing is a technique used to access a range of elements from a list. It allows you to extract specific portions of a list by specifying start, stop, and step indices.

```
list[start:stop:step]
```

- start: The index where the slice starts (inclusive). Defaults to 0 if omitted.
- stop: The index where the slice ends (exclusive). Mandatory unless slicing backward with step.
- step: The interval between elements. Defaults to 1 if omitted.

Example -

```
my_list = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

# Extract elements from index 2 to 5 (exclusive)
print(my_list[2:5])  # Output: [2, 3, 4]
```