Data Types: data type specifies the type of value a variable can hold and the operations that can be performed on it.

Common Data Types in Python:

1. String (str) - Represents a sequence of characters enclosed in single or double quotes.
- Example:

```
name = "ANURAG"
print(name)  # Output: ANURAG
print(name[2]) # Output: U
```

2. Integer (int) - Represents whole numbers (positive or negative) without decimals. Example:

```
x = 5
y = -6
```

3. Float - Represents real numbers (numbers with decimals). Example: y = 3.14

4. Boolean (bool) - Represents a truth value, either True or False. Example:

```
is_valid = False
is_valid = True
```

5. List (list) - An ordered, mutable collection of elements of same or different data types.
Example:

1. data = ["apple", "banana", "cherry", "Ram", 45, 90]
2. fruits = ["apple", "banana", "cherry"]
   print(fruits[2]) # Output: cherry

Lists are mutable, meaning they can be modified:

Creating a list -> fruits = ["apple", "banana", "cherry"]
Modifying an element -> fruits[1] = "blueberry"

```
print(fruits)  # Output: ['apple', 'blueberry', 'cherry']
```

Adding a new element -> fruits.append("orange")

```
print(fruits)  # Output: ['apple', 'blueberry', 'cherry', 'orange']
```

Removing an element -> fruits.remove("cherry")

print(fruits) # Output: ['apple', 'blueberry', 'orange'] ```

Get list length:

```
my_list = [1, 2, 3, "apple", 4.5]
print(len(my_list)) # Output: 5
```

6. Set (set) - An unordered collection of unique elements of same or different data types. Example:

```
data = {"apple", "banana", "cherry", "Ram", 45, 90}
colors = {"red",green", "blue"}
print(colors) # Output: {'blue', 'red', 'green'}
```

Sets ignore duplicate values:

```
my_set = {1, 2, 2, 3, 3, 4}
print(my_set) # Output: {1, 2, 3, 4}
```

7. Tuple (tuple) - An ordered, immutable collection of elements. Example:

```
fruits = (“apple”, “banana”, “orange”)      # a tuple of strings
my_tuple = (10, 20, 30)                     # a tuple of integers
print(my_tuple[1]) # Output: 20
```

Difference from List: Tuples cannot be modified after creation.

8. Dictionary (dict) - A collection of key-value pairs where keys must be unique. Example:

```
person = {"name": "John", "age": 30, "city": "New York"}
print(person["name"]) # Output: John
```

These fundamental data types help in structuring and organizing data efficiently in Python programming.

Interview Question

Q. Difference b/w List, Set and Tuple

Ans –

Features	List	Set	Tuple
Orders	Ordered (maintains insertion order)	Unordered (order is not maintained)	Ordered (maintains insertion order)
Indexing	Supports indexing	Does not support indexing	Supports indexing
Duplicates	Allows duplicates	Does not allow duplicates	Allows duplicates
Mutability	Mutable (can change elements)	Mutable (can add/remove elements but not modify existing ones)	Immutable (cannot change elements after creation)
Example	fruits = ["apple", "banana", "cherry"]	fruits = {"apple", "banana", "cherry"}	My_tuple = (10, 20, 30) fruits = (“apple”, “banana”, “cherry”)
Use	when you need an ordered, modifiable collection with duplicates allowed.	when you need unique elements and fast membership checking (in operation).	when you need an ordered, immutable collection (e.g., fixed configuration data).