Chapter 2: Python Data Types:

In Python, data types represent the type or category of data that can be stored and manipulated within a program. Python supports several built-in data types, each serving a different purpose. Here are the main data types in Python:

Numeric Types:

- **int**: Integer data type represents whole numbers, positive or negative, without any decimal point.

Ex:

```
x = 10
y = -5
```

- **float**: Float data type represents floating-point numbers, which include decimal points.

Ex:

```
pi = 3.14
temperature = 98.6
```

- **complex**: Complex data type represents complex numbers in the form a + bj, where a and b are real numbers, and j is the imaginary unit.

Ex:

```
z = 2 + 3j
```

Sequence Types:

- **str**: String data type represents sequences of characters enclosed within single quotes (') or double quotes (").

Ex:

```
name = 'Ajay'
message = "Hello, world!"
```

- **list**: List data type represents ordered collections of items enclosed within square brackets ([]). Lists can contain elements of different data types.

Ex:

```
numbers = [1, 2, 3, 4, 5]
fruits = ['apple', 'banana', 'orange']
```

- **tuple**: Tuple data type represents immutable ordered collections of items enclosed within parentheses (()). Tuples can contain elements of different data types.

Ex:

```
coordinates = (3, 5)
rgb_colors = (255, 0, 0)
```

- **range**: Range data type represents a sequence of numbers generated by the range() function.

Mapping Type:

- **dict**: Dictionary data type represents unordered collections of key-value pairs enclosed within curly braces ({}). Each key-value pair in a dictionary maps the key to its corresponding value.

Ex:

```
person = {'name': 'Alice', 'age': 30}
```

Set Types:

- **set**: Set data type represents unordered collections of unique elements enclosed within curly braces ({}). Sets do not allow duplicate elements.

Ex:

```
vowels = {'a', 'e', 'i', 'o', 'u'}
```

- **frozenset**: Frozenset data type represents immutable sets, similar to sets but cannot be modified after creation.

Boolean Type:

- **bool**: Boolean data type represents the truth values True and False, used for logical operations and comparisons.

Ex:

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

Binary Types:

- **bytes**: Bytes data type represents immutable sequences of bytes, typically used to store binary data.
- **bytearray**: Bytearray data type represents mutable sequences of bytes, similar to bytes but can be modified.

None Type:

- **None**: None data type represents the absence of a value or a null value in Python.

Ex:

```
result = None
```

Understanding and utilizing these data types effectively is essential for writing Python programs that are efficient, readable, and maintainable. Each data type has its own set of operations and methods for manipulation, and choosing the appropriate data type depends on the specific requirements and context of your code.