Key Points on Data Types in Python

What is a Data Type?

- A **data type** is an attribute of data that determines how it is stored and interpreted by a computer system.
- In Python, data types ensure data is stored in the correct format and behaves as expected during operations.

Types of Data in Python

Python offers several built-in data types categorized into five main types:

1. Numeric Data Types

Numeric data types store numbers and are divided into:

- **Integer (`int`)**: Represents whole numbers (e.g., `10` , `-5`).
- **Float (`float`)**: Represents numbers with decimal points (e.g., `10.5` , `-2.3`).
- **Complex (`complex`)**: Represents numbers with both real and imaginary parts (e.g., `10 + 5j`).

2. Sequence Data Types

Sequence types store ordered collections of items:

- **String (`str`)**: Sequence of characters enclosed in single or double quotes (e.g., `"Hello"` or `'World'`).
- **List (`list`)**: Mutable sequences of items, defined using square brackets (e.g., `[1, 2, 3]`).
- **Tuple (`tuple`)**: Immutable sequences of items, defined using parentheses (e.g., `(1, 2, 3)`).

3. Dictionary Data Type

- **Dictionary (`dict`)**: Stores data in key-value pairs, enclosed in curly braces (e.g., `{"a": 1, "b": 2}`).
  - Keys must be unique and immutable (e.g., strings or integers).
  - Values can be any data type.

4. Boolean Data Type

- **Boolean (`bool`)**: Represents `True` or `False` values, often used for logical operations and conditions.

5. Set Data Type

- **Set (`set`)**: Unordered, non-indexed collections of unique items, defined using curly braces (e.g., `{1, 2, 3}`).

How to Check Data Types

Python provides the `type()` function to check the data type of a variable:

```
a = 10          # Integer
b = 2.5         # Float
c = "Python"    # String
d = [1, 2, 3]   # List

print(type(a))  # Output: <class 'int'>
print(type(b))  # Output: <class 'float'>
print(type(c))  # Output: <class 'str'>
print(type(d))  # Output: <class 'list'>
```

Automatic Type Assignment

Python automatically assigns the correct data type based on the value provided during variable assignment:

```
x = 10        # Integer
y = 3.14      # Float
z = "Hello"   # String
```

Examples of Each Data Type

Numeric Examples

```
int_example = 42          # Integer
float_example = 3.14      # Float
complex_example = 1 + 2j  # Complex
```

Sequence Examples

```
string_example = "Hello"
list_example = [1, 2, 3]
```

```
tuple_example = (4, 5, 6)
```

Dictionary Example

```
dict_example = {"name": "John", "age": 30}
print(dict_example["name"])  # Output: John
```

Boolean Example

```
bool_example = True
is_greater = 5 > 3  # True
```

Set Example

```
set_example = {1, 2, 3}
unique_items = set([1, 1, 2, 3])  # {1, 2, 3}
```

Practice

Experiment with these data types to understand their behavior:

1. Create variables of different data types.
2. Use the `type()` function to verify their types.
3. Manipulate and perform operations to see how data types work.