part-2

Python Data Types

Numeric Data Types

- **Integers**: Whole numbers (e.g., 7, -90, 365)
- **Floating-Point Numbers**: Numbers with decimal points (e.g., 7.8, -90.5, 2.5)
- **Complex Numbers**: Numbers with real and imaginary parts (e.g., 8 + 6j)

Type Conversion

- You can use type conversion functions to convert between numeric data types:
  - `int()`: Convert to integer
  - `float()`: Convert to floating-point number
  - `complex()`: Convert to complex number
  - Conversion will truncate the decimal part when converting from float to int
  - Converting an integer to a float will add a `.0` to the end

Boolean Data Types

- Represents `True` or `False` values
- Can be used in comparison operations (e.g., `num > c`)

String Data Types

- Represented within single or double quotes
- Can perform indexing and slicing operations
- Indexing starts at 0
- Negative indexing starts from the end
- Supports string manipulation methods (e.g., `replace()`, `find()`)

Sequence Data Types

Lists

- Ordered collection of items
- Can store different data types
- Supports operations like appending, popping, and reversing

Tuples

- Ordered collection of items
- Immutable (cannot be modified after creation)
- Supports indexing and slicing like lists

Ranges

- Represents a sequence of numbers
- Created using the `range()`function

Mapping Data Types

Dictionaries

- Unordered collection of key-value pairs
- Keys must be unique
- Supports adding, modifying, and retrieving values

Set Data Types

- Unordered collection of unique items
- Supports set operations like union, intersection, and difference

Binary Data Types

- `bytes`: Immutable sequence of bytes
- `bytearray`: Mutable sequence of bytes

- `memoryview`: Provides a view into an object's memory

Type Conversion Functions

- `int()`, `float()`, `complex()`
- `str()`, `list()`, `tuple()`
- `bin()`, `hex()`, `oct()`
- `chr()`, `ord()`

Key Takeaways

- Python has a variety of built-in data types to represent different kinds of data
- You can perform type conversions using various functions
- Data types have specific properties and support different operations
- Understanding data types is crucial for effective Python programming

Data Type	Description	Example
Integer	Whole numbers	`7`, `-90`, `365`
Float	Numbers with decimal points	`7.8`, `-90.5`, `2.5`
Complex	Numbers with real and imaginary parts	`8 + 6j`
Boolean	`True` or `False` values	`True`, `False`
String	Textual data	`"Python Programming"`, `'simply learn'`
List	Ordered collection of items	`[1, 0, 1, "America", 2.05]`
Tuple	Ordered, immutable collection	`(1, 2, 3, 4)`
Dictionary	Unordered collection of key-value pairs	`{1: "tiger", 2: "lion", 3: "dog"}`
Set	Unordered collection of unique items	`{1, 2, 3, 4}`

Bytes	Immutable sequence of bytes	`b"hello"`