**Python Data Types & Typecasting**

* Python distinguishes data types like **integers, floats, strings, Booleans**.
* Typecasting allows conversion between types (e.g., int to float, float to string, etc.).
* Boolean conversion: 0 becomes False, non-zero becomes True.

**2. Expressions & Variables**

* Expressions use operators to perform calculations (e.g., +, -, \*, /, //).
* Python respects **BODMAS** (order of operations).
* Variables store and manipulate data using the = operator.
* Variables can be reassigned and combined in expressions.

**3. Strings & String Operations**

* Strings are sequences of characters, immutable, and enclosed in quotes.
* Operations include indexing, slicing, concatenation, and escape sequences (\n, \t, etc.).
* Python provides built-in **string methods** (e.g., .replace(), .lower(), .find()).

**4. Tuples, Lists, Dictionaries, and Sets**

* **Tuples**: Immutable, ordered collections; support indexing and slicing.
* **Lists**: Mutable, ordered collections; support operations like append, delete, etc.
* **Dictionaries**: Key-value pairs with unique, immutable keys.
* **Sets**: Unordered collections of unique elements; support operations like union, intersection.

**5. Conditional Statements**

* Use if, elif, and else to create logic based on comparisons (==, !=, <, >).
* Boolean logic (and, or, not) helps combine conditions.

**6. Loops**

* **For loops** iterate over sequences (lists, strings, etc.).
* **While loops** repeat as long as a condition is true.
* The range() function generates sequences of numbers for loops.

**7. Functions & Scope**

* Functions are reusable blocks of code using def, can take parameters and return values.
* Built-in functions include len(), sum(), sorted().
* Scope:
  + **Local**: Defined inside functions.
  + **Global**: Defined outside functions.

**8. Exception Handling**

* Prevent crashes using try, except, else, finally.
* Helps handle runtime errors gracefully.

**9. Object-Oriented Programming**

* **Objects** are instances of **classes**, which define attributes and methods.
* \_\_init\_\_ initializes object attributes.
* Methods operate on object data via self.

**10. File Handling**

* Use open() to **read ('r')**, **write ('w')**, or **append ('a')** to files.
* Context managers (with open(...)) ensure proper file handling.
* \n indicates a new line in files.

**11. Pandas Library**

* **Pandas** is used for **data manipulation** via **DataFrames** and **Series**.
* Supports operations like reading files, filtering, grouping, and summarizing.
* Key methods: .head(), .mean(), .unique(), etc.

**12. NumPy Library**

* Provides fast operations on **multi-dimensional arrays (ndarrays)**.
* Supports indexing, slicing, vector math, and matrix operations.
* Functions: .dtype, .shape, .size, dot(), etc.

**13. APIs & Web Interaction**

* **APIs** let software components communicate, often using HTTP protocols like **GET**, **POST**, **PUT**, and **DELETE**.
* Python’s requests library handles HTTP interactions.
* **REST APIs** return data in formats like JSON.

**14. Web Scraping**

* Use **Beautiful Soup** and **requests** to extract data from websites.
* HTML is parsed into trees; elements are located using find\_all, tags, and attributes.
* Tables can be read using pandas.read\_html().

**15. File Formats**

* Python supports formats like .txt, .csv, .json, .xml, .xlsx.
* Use appropriate libraries (e.g., **Pandas** for CSV, **json** module for JSON).