**Python Notes Based on the Questions:**

**1. Basic Data Types:**

* **Integer (int):** Whole numbers (e.g., 10, -5, 0).
* **Float (float):** Decimal numbers (e.g., 3.14, -2.5).
* **String (str):** Sequences of characters (e.g., "Hello", 'Python').
* **Boolean (bool):** True or False values.
* **Type Checking:** Use the type() function (e.g., type(variable)).

**2. String Manipulation:**

* **Reversing a String:** Use slicing with a negative step (e.g., string[::-1]).
* **Uppercase Conversion:** Use the .upper() method (e.g., string.upper()).

**3. Lists:**

* **Creating a List:** Use square brackets [] (e.g., my\_list = [1, 2, 3]).
* **Adding Elements:** Use the .append() method (e.g., my\_list.append(6)).
* **Removing Elements:** Use the .remove() method (e.g., my\_list.remove(3)) or del my\_list[index] or .pop(index)
* **List comprehensions:** A concise way to create lists.

**4. Conditional Statements:**

* **if Statements:** Execute code based on a condition.
* **if/else Statements:** Execute different code blocks based on a condition.
* **if/elif/else Statements:** For multiple conditions.

**5. Loops:**

* **for Loop:** Iterate over a sequence (e.g., for i in range(10):).
* **while Loop:** Repeat a block of code while a condition is true.

**6. Functions:**

* **Defining Functions:** Use the def keyword (e.g., def add(a, b):).
* **Return Statement:** Use return to send a value back from a function.

**7. Dictionaries:**

* **Creating Dictionaries:** Use curly braces {} and key-value pairs (e.g., my\_dict = {"name": "Alice", "age": 30}).
* **Accessing Values:** Use the key in square brackets (e.g., my\_dict["name"]).

**8. File Handling:**

* **Reading Files:** Use the open() function with 'r' mode and .read() or .readlines().
* **Writing Files:** Use the open() function with 'w' mode and .write().
* **Context Managers:** Use with open(...) as file: to close files automatically.

**9. List Comprehensions:**

* **Syntax:** [expression for item in iterable if condition].
* **Example:** squares = [x\*\*2 for x in range(1, 11)].

**10. Error Handling:**

* **try/except Blocks:** Handle exceptions (e.g., try: ... except ZeroDivisionError: ...).

**11. Object-Oriented Programming (OOP):**

* **Classes:** Use the class keyword to define a blueprint for objects.
* **Attributes:** Variables associated with a class or object.
* **Methods:** Functions associated with a class.
* **Objects:** Instances of a class.

**12. Modules and Packages:**

* **Modules:** Python files containing code.
* **Packages:** Directories containing modules and an \_\_init\_\_.py file.
* **Importing:** Use the import keyword (e.g., import module, from module import function).

**13. Regular Expressions:**

* **re Module:** Use the re module for regular expression operations.
* **Pattern Matching:** Use functions like re.findall().
* **Email Pattern Example:** r"[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}"