Module 15: Advanced Python

1. Printing on Screen

print() function: This is your primary tool for displaying text, variables, and results to the console. It's incredibly versatile

Formatting Outputs: Making your output clear and readable is important.

1. Reading Data from Keyboard

input() function: This function pauses your program, displays a prompt to the user, and waits for them to type something and press Enter. Whatever the user types is returned as a string.

Converting User Input: Since input() always returns a string, you often need to convert the input to the appropriate data type (e.g., int() for integers, float() for floating-point numbers) if you plan to perform numerical operations or other type-specific actions

1. Opening and Closing Files

open() function: This function is used to open a file. It takes the file path and a "mode" as arguments.

Modes: 'r' (read): Opens for reading (default). 'w' (write): Opens for writing. Creates a new file if it doesn't exist or truncates (empties) the file if it does. 'a' (append): Opens for writing. Appends to the end of the file if it exists, or creates a new file. 'r+' (read/write): Opens for both reading and writing. 'w+' (write/read): Opens for both writing and reading. Truncates the file.

close() method: After you're done working with a file, it's essential to close it using the close() method. This releases the file resource, ensures all buffered data is written, and prevents potential data corruption or resource leaks. Using with open(...) as file: is the preferred way, as it automatically handles closing the file.

1. Reading and Writing Files

Once a file is open, you can perform operations to read its contents or write new data to it.

read() : Reads the entire content of the file as a single string. readline() : Reads a single line from the file. readlines() : Reads all lines from the file and returns them as a list of strings, where each string is a line.

1. Exception Handling

Introduction to Exceptions: An exception is an event that disrupts the normal flow of a program. It's Python's way of signaling that something went wrong

try block: Contains the code that might raise an exception. except block: Catches and handles specific types of exceptions that occur in the try block. You can have multiple except blocks for different exception types. finally block: Contains code that will always be executed, regardless of whether an exception occurred or was handled. It's often used for cleanup operations like closing files.

Multiple Exceptions and Custom Exceptions: You can handle different types of exceptions separately. You can also define your own custom exception classes to represent specific error conditions in your application, making your error messages more meaningful.

1. Class and Object

Class: A blueprint or a template for creating objects. It defines the common properties (attributes) and behaviors (methods) that all objects of that type will have. Object: An instance of a class. It's a concrete entity created from the class blueprint, with its own unique set of attribute values. Attributes: Variables that belong to an object (or class) and store data. They represent the characteristics or state of an object. Methods: Functions that belong to an object (or class) and define its behavior. They represent the actions an object can perform.

Local and Global Variables:

Local Variables: Defined inside a method (or function) and are only accessible within that method. Global Variables: Defined outside any class or function and can be accessed from anywhere in the program. Within a method, if you want to modify a global variable, you need to explicitly declare it using the global keyword

1. Inheritance

Inheritance is a core OOP principle that promotes code reusability and establishes a hierarchical relationship between classes.

Single Inheritance: A class inherits from only one base class. Multilevel Inheritance: A class inherits from another class, which in turn inherits from a third class (A -> B -> C). Multiple Inheritance: A class inherits from multiple base classes. Hierarchical Inheritance: Multiple classes inherit from a single base class. Hybrid Inheritance: A combination of two or more types of inheritance

super() function: This built-in function allows a child class to call methods or access attributes of its parent class. It's commonly used in the \_\_init\_\_ method of a child class to properly initialize the parent class's attributes.

1. Method Overloading and Overriding

Method Overloading: (Not directly supported in Python in the same way as some other languages like Java/C++). In Python, if you define multiple methods with the same name in the same class, the last one defined will overwrite the previous ones. However, you can achieve similar functionality using default arguments, variable-length arguments ( args , kwargs ), or type checking within a single method. The idea is to have a single method that can handle different numbers or types of arguments

Method Overriding: This occurs when a child class provides its own specific implementation for a method that is already defined in its parent class. The child class's method "overrides" the parent's method when called on an object of the child class. This allows for specialized behavior in subclasses while maintaining a common interface

1. SQLite3 and PyMySQL

Database Connectivity: These are modules that allow your Python code to interact with specific types of databases.

SQLite3: Python has built-in support for SQLite, a lightweight, file-based database. It's great for small-scale applications or local data storage because it doesn't require a separate server process. PyMySQL: A Python library that allows you to connect to and interact with MySQL databases. MySQL is a popular relational database management system often used for larger, more complex applications

Establish a connection to the database. Create a "cursor" object, which is used to execute SQL commands. Execute SQL queries (e.g., CREATE TABLE , INSERT , SELECT , UPDATE , DELETE ). Fetch results from SELECT queries. Commit (save) or rollback (undo) changes to the database.

1. Search and Match Functions

re module: Python's built-in module for working with regular expressions.

re.search() : Scans through a string looking for the first location where the regular expression pattern produces a match. It returns a match object if a match is found, otherwise None .re.search() : Scans

through a string looking for the first location where the regular expression pattern produces a match. It returns a match object if a match is found, otherwise None .

Difference between search and match : The key difference is where they look for the pattern. re.match() only looks at the beginning of the string, while re.search() scans the entire string. re.search() is generally more commonly used when you want to find a pattern anywhere within a string.