Daily Diary - June 17, 2024

Python File Handling

Today, I explored the fundamentals of file handling in Python. I learned about various file modes, such as reading ('r'), writing ('w'), appending ('a'), and working with binary files ('b'). Key file methods like open(), read(), write(), and close() were covered. Using the with statement ensures proper file handling, which automatically closes the file after the block of code is executed.

Practical Scenarios

- Reading and writing files: Implemented functions to read entire files, write to files, and append content to existing files.
- Error handling: Used try-except blocks to handle potential errors like FileNotFoundError and IOError.
- Working with binary files: Practiced writing and reading binary data to and from files.

Exception and Error Handling

The focus was on managing exceptions and errors in Python programs to make them more robust and user-friendly. Key points included:

- Types of exceptions: Differentiated between built-in exceptions like ValueError, TypeError, and custom exceptions.
- Using try, except, else, and finally blocks: Structured my code to handle exceptions gracefully, ensuring resources are properly released and necessary cleanup is performed.
- Raising exceptions: Learned how to raise exceptions deliberately using the raise statement for specific conditions.
- Creating custom exceptions: Defined custom exception classes to handle unique error situations in my projects.

Working with Random, Datetime, and Math Modules

The day also included working with some essential Python modules:

Random module

- Generated random numbers and made use of functions like randint(), choice(), and shuffle(). This module is crucial for creating random data and simulating random events in programs.

Datetime module

- Worked with dates and times, performing operations like getting the current date/time, formatting dates, and performing arithmetic with dates.

Functions such as datetime.now(), timedelta, and strftime() were very useful.

Math module

- Utilized mathematical functions for more complex calculations, such as finding square roots with sqrt(), calculating the power of numbers with pow(), and using constants like pi.