Lab #9

In Chapter 13 and 20, foundational concepts about exceptions and unit testing is explained.
Chapter 13 introduces what exceptions are and how to handle errors using "try", "except", and "raise". It explained different types of standard exceptions and best practices for catching and managing them. I found the section on exception flow control and cleanup especially helpful for understanding how to prevent my programs from crashing unexpectedly.
Chapter 20 focused on unit testing — starting with "assert" statements and defensive programming, then progressing into writing structured tests with "pytest". I learned how to design testable functions, write tests before coding (Test-First Development), and interpret pytest failure reports. The breakdown of automated testing and the role of "pytest" really clarified how to test code effectively and consistently. These chapters gave me the tools to make my programs more reliable and easier to debug.

The "Runtime Stack" slides explain how function calls are managed in memory during a program's execution. I learned that everytime a function is called, a new stack frame is pushed onto the stack, storing that functions parameters and local variables. When the function finishes, its frame is propped off the stack. This system keeps track of the program's control flow and helps explain how nested function calls are executed in order. The slides also highlight the difference between the stack (for function calls) and the heap (for object storage like lists and variables), reinforcing how Python stores and accesses data behind the scenes.