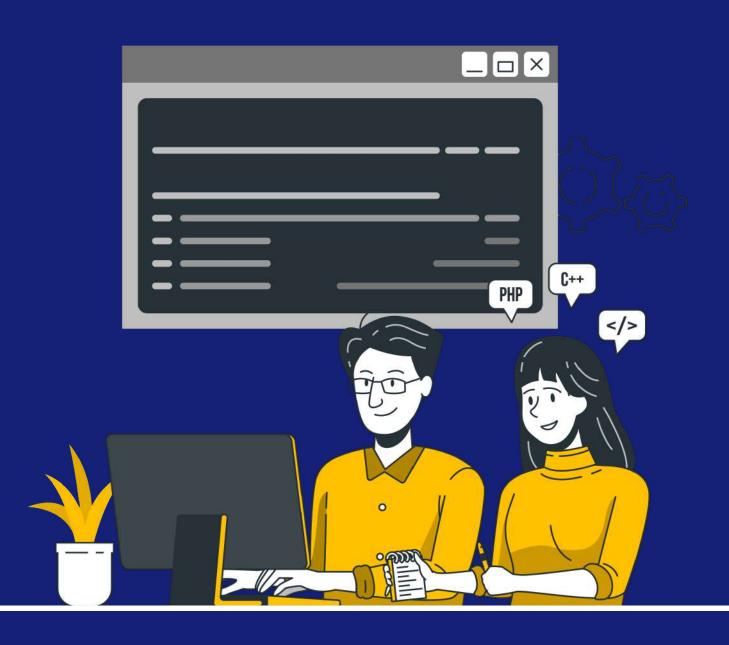


Assignment

File handling, Exception Handling and Multitasking in Python





- 1. Write a code to read the contents of a file in Python.
- 2. Write a code to write to a file in Python.
- 3. Write a code to append to a file in Python.
- 4. Write a code to read a binary file in Python.
- 5. What happens if we don't use `with` keyword with `open` in python?
- 6. Explain the concept of buffering in file handling and how it helps in improving read and write operations.
- 7. Describe the steps involved in implementing buffered file handling in a programming language of your choice.
- 8. Write a Python function to read a text file using buffered reading and return its contents.
- 9. What are the advantages of using buffered reading over direct file reading in Python?
- 10. Write a Python code snippet to append content to a file using buffered writing.
- 11. Write a Python function that demonstrates the use of close() method on a file.
- 12. Create a Python function to showcase the detach() method on a file object.
- 13. Write a Python function to demonstrate the use of the seek() method to change the file position.
- 14. Create a Python function to return the file descriptor (integer number) of a file using the fileno() method.
- 15. Write a Python function to return the current position of the file's object using the tell() method.
- 16. Create a Python program that logs a message to a file using the logging module.
- 17. Explain the importance of logging levels in Python's logging module.
- 18. Create a Python program that uses the debugger to find the value of a variable inside a loop.
- 19. Create a Python program that demonstrates setting breakpoints and inspecting variables using the debugger
- 20. Create a Python program that uses the debugger to trace a recursive function.
- 21. Write a try-except block to handle a ZeroDivisionError.
- 22. How does the else block work with try-except?
- 23. Implement a try-except-else block to open and read a file.
- 24. What is the purpose of the finally block in exception handling.
- 25. Write a try-except-finally block to handle a ValueError.



- 26. How multiple except blocks work in Python?
- 27. What is a custom exception in Python?
- 28. Create a custom exception class with a message.
- 29. Write a code to raise a custom exception in Python.
- 30. Write a function that raises a custom exception when a value is negative.
- 31. What is the role of try, except, else, and finally in handling exceptions.
- 32. How can custom exceptions improve code readability and maintainability?
- 33. What is multithreading?
- 34. Create a thread in Python.
- 35. What is the Global Interpreter Lock (GIL) in Python?
- 36. Implement a simple multithreading example in Python.
- 37. What is the purpose of the 'join()' method in threading?
- 38. Describe a scenario where multithreading would be beneficial in Python.
- 39. What is multiprocessing in Python?
- 40. How is multiprocessing different from multithreading in Python?
- 41. Create a process using the multiprocessing module in Python.
- 42. Explain the concept of Pool in the multiprocessing module.
- 43. Explain inter-process communication in multiprocessing.