

M.AKHIL REDDY

2303A52315

BATCH 45

Ass 5.5

Task Description #1 (Transparency in Algorithm Optimization)

Task: Use AI to generate two solutions for checking prime

numbers:

- Naive approach(basic)
- Optimized approach

Prompt:

“Generate Python code for two prime-checking methods and explain how the optimized version improves performance.”

Expected Output:

- Code for both methods.
- Transparent explanation of time complexity.
- Comparison highlighting efficiency improvements

```
1.5.py | 2.5.py | 3.5.py | 5.5.py
1 def is_prime(n):
2     while i * i <= n:
3         if n % i == 0 or n % (i + 2) == 0:
4             return False
5         i += 6
6
7     return True
8
9
10 num = int(input("Enter a number: "))
11
12 if is_prime(num):
13     print(f"{num} is a prime number")
14 else:
15     print(f"{num} is not a prime number")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/3.5.py"
Enter a number: 19
19 is not a Harshad number.

PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/5.5.py"
Enter a number: 17
17 is a prime number

Task Description #2 (Transparency in Recursive Algorithms)

Objective: Use AI to generate a recursive function to calculate

Fibonacci numbers.

Instructions:

1. Ask AI to add clear comments explaining recursion.
2. Ask AI to explain base cases and recursive calls.

Expected Output:

- Well-commented recursive code.
- Clear explanation of how recursion works.
- Verification that explanation matches actual execution.

```
EXPLORER File Edit Selection View Go ... 1.5.py 2.5.py 3.5.py 5.5.py akhils ai coding

AKHILS AI CODING
1.5.py
2.5.py
3.5.py
5.5.py

22
23 def is_fibonacci(n):
24     if n < 0:
25         return False
26
27     a, b = 0, 1
28     while a < n:
29         a, b = b, a + b
30
31     return a == n
32
33
34 # Example usage
35 num = int(input("Enter a number: "))
36
37 if is_fibonacci(num):
38     print(f"{num} is a Fibonacci number.")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
+ & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe ...
+ ~~
+ CategoryInfo : ObjectNotFound: (:)[], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding> & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/5.5.py"
Enter a number: 11
11 is not a Fibonacci number.

PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding>
```

Use AI to generate a Python program that reads a file and

processes data.

Prompt:

“Generate code with proper error handling and clear explanations for each exception.”

Expected Output:

- Code with meaningful exception handling.
- Clear comments explaining each error scenario.
- Validation that explanations align with runtime behavior.

The screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left lists files: 1.5.py, 2.5.py, 3.5.py, and 5.5.py. The 5.5.py file is open in the editor. The code reads a file named 'new.txt' and prints its contents. The terminal at the bottom shows the output of running the script, which includes an error message about a missing file.

```

43     def read_file(file_path):
44         try:
45             with open(file_path, 'r') as file:
46                 data = file.readlines()
47                 # Process data (for example, print each line)
48                 for line in data:
49                     print(line.strip())
50             except FileNotFoundError:
51                 print(f"Error: The file at {file_path} was not found.")
52             except IOError:
53                 print("Error: An I/O error occurred while reading the file.")
54
55     read_file('C:\\\\Users\\\\achyu\\\\Desktop\\\\sujith hacker\\\\new.txt')
56

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/5.5.py"
Enter a number: 11
11 is not a Fibonacci number.
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/5.5.py"
Error: The file at C:\\\\Users\\\\achyu\\\\Desktop\\\\sujith hacker\\\\new.txt was not found.
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding>

Use an AI tool to generate a Python-based login system.

Analyze: Check whether the AI uses secure password handling practices.

Expected Output:

- Identification of security flaws (plain-text passwords, weak validation).
- Revised version using password hashing and input validation

The screenshot shows the Visual Studio Code interface. The Explorer sidebar lists files: 1.5.py, 2.5.py, 3.5.py, and 5.5.py. The 5.5.py file is open in the editor. The code defines a login_system function that takes stored_username and stored_password as arguments. It prompts the user for a username and password, then checks if they match the stored values. The terminal at the bottom shows the output of running the script, which includes an error message about a missing file.

```

57     # generate a code for a login system for password handling
58
59     def login_system(stored_username, stored_password):
60         username = input("Enter username: ")
61         password = input("Enter password: ")
62
63         if username == stored_username and password == stored_password:
64             print("Login successful")
65         else:
66             print("Login failed. Incorrect username or password.")
67
68
69     login_system("admin", "password123")

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

11 is not a Fibonacci number.
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/5.5.py"
Error: The file at C:\\\\Users\\\\achyu\\\\Desktop\\\\sujith hacker\\\\new.txt was not found.
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding & C:/Users/achyu/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/achyu/OneDrive/Desktop/akhils ai coding/5.5.py"
Enter username: admin
Enter password: 123
Login failed. Incorrect username or password.
PS C:\Users\achyu\OneDrive\Desktop\akhils ai coding>

Use an AI tool to generate a Python script that logs user

activity (username, IP address, timestamp).

Analyze: Examine whether sensitive data is logged unnecessarily or insecurely.

Expected Output:

- Identified privacy risks in logging.
- Improved version with minimal, anonymized, or masked logging.

- Explanation of privacy-aware logging principles.

The screenshot shows a VS Code interface with the following details:

- EXPLORER:** Shows files 1.5.py, 2.5.py, 3.5.py, and 5.5.py under the AKHILS AI CODING folder, along with a user_activity.log file.
- CODE EDITOR:** Displays the content of 5.5.py. The code defines two functions: log_user_activity and log_user_activity_privacy_aware. The log_user_activity function logs the full IP address and timestamp. The log_user_activity_privacy_aware function logs the IP address with its last two octets masked (e.g., "254.789.192.168.1" becomes "254.789.192.xxx.xxx") and only logs the date part of the timestamp.
- TERMINAL:** Shows the command line output of running the script. It includes the error message "Error: The file at C:/Users/achyu/Desktop/akhils ai coding&new.txt was not found.", followed by a password prompt "Enter password: 123", and a failure message "Login failed. Incorrect username or password."