

Name: A . Sai Varshith

Roll Number: 2303A51831

Batch - 04

AI Assisted Coding

30-01-2026

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.

The screenshot shows a code editor with Python code for prime number checking. The code defines two functions: `is_prime_naive` and `is_prime_optimized`. The `is_prime_naive` function uses a simple loop from 2 to n to check if n is divisible by any number. The `is_prime_optimized` function uses a similar loop but only goes up to the square root of n . Both functions return `True` if no divisor is found and `False` otherwise. Below the code, the terminal window shows the code being run in a Windows environment. It prompts for a number, prints the results of both methods, and ends with a command to exit the debugger.

```
assignment 5.5.py >-
1  # Task 1: Prime Number Checking
2
3  import math
4
5  # Naive approach (Basic Method)
6  def is_prime_naive(n):
7      if n <= 1:
8          return False
9
10     for i in range(2, n):
11         if n % i == 0:
12             return False
13     return True
14
15
16 # Optimized approach (Checks up to sqrt(n))
17 def is_prime_optimized(n):
18     if n <= 1:
19         return False
20
21     for i in range(2, int(math.sqrt(n)) + 1):
22         if n % i == 0:
23             return False
24     return True
25

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\hp\OneDrive\Desktop\ai> & 'c:\Users\hp\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\hp\vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\lib\debug\launcher' "52617" '--' 'C:\Users\hp\OneDrive\Desktop\ai\assignment_5.5.py'
Enter a number: 5
Naive Method: True
Optimized Method: True
PS C:\Users\hp\OneDrive\Desktop\ai>
```

Explanation:

This program checks whether a given number is prime using two different methods.

- **Naive Method:**

It checks divisibility of the number from 2 to $n-1$.

If any number divides n , it is not prime.

- **Optimized Method:**

It checks divisibility only up to \sqrt{n} because if n has a factor greater than \sqrt{n} , it must also have a corresponding factor smaller than \sqrt{n} .

Time Complexity:

- Naive approach: $O(n)$
- Optimized approach: $O(\sqrt{n})$

Ethical Transparency:

The optimized method improves performance while clearly explaining why fewer iterations are sufficient, ensuring algorithmic transparency.

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.

```

assignment 5.5.py >_
1 def fibonacci(n):
2     # Base case 1: If n is 0, return 0
3     if n == 0:
4         return 0
5
6     # Base case 2: If n is 1, return 1
7     if n == 1:
8         return 1
9
10    # Recursive call: function calls itself
11    return fibonacci(n - 1) + fibonacci(n - 2)
12
13
14
15
16    # Driver code
17 num = int(input("Enter number of terms: "))
18
19 print("Fibonacci Series:")
20 for i in range(num):
21     print(fibonacci(i), end=" ")
22

```

The terminal window shows the execution of the script. It starts with the command PS C:\Users\hp\OneDrive\Desktop\ai> & 'c:\Users\hp\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\hp\.vscode\extensions\ms-python.vscode-pylnk\2025.18.0-win32-x64\handled\lib\vsdebugpy\launcher' '52617' '--' 'c:\Users\hp\OneDrive\Desktop\ai\assignment 5.5.py'. The user is prompted to enter the number of terms, which they type as 5. The output shows the first five Fibonacci numbers: 0 1 1 2 3.

Explanation:

This program calculates Fibonacci numbers using **recursion**, where a function calls itself.

- **Base Case 1:** When $n = 0$, the function returns 0.
- **Base Case 2:** When $n = 1$, the function returns 1.
- **Recursive Case:** For all other values, the function calls itself as `fibonacci(n-1) + fibonacci(n-2)`.

The base cases prevent infinite recursion and ensure correct termination.

Ethical Transparency:

Clear comments and explanations help developers understand recursive behavior and avoid logical or performance errors.

Task Description #3 (Transparency in Error Handling)

Task: 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.

```

1 Welcome  day 1.py M  assignment 5.5.py U  task1.py M  assignment.py U  day 1.py (Index)  task1.py (Index)  assign-2(2303A51118)-copilot.py  t2.jpeg  D v  ...
2
3 def read_file(filename):
4     print("File Content:\n", data)
5
6     except FileNotFoundError:
7         # If file does not exist
8         print("Error: File not found.")
9
10    except PermissionError:
11        # If permission is denied
12        print("Error: Permission denied.")
13
14    except Exception as e:
15        # For any other unexpected errors
16        print("Unexpected error occurred:", e)
17
18
19
20
21
22 # Driver code
23 file_name = input("Enter file name: ")
24 read_file(file_name)
25
26

```

TERMINAL

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\hp\OneDrive\Desktop\ai> c:; cd 'c:\Users\hp\OneDrive\Desktop\ai'; & 'c:\Users\hp\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\hp\.vscode\extensions\ms-python.debugger-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '63992' ...
PS C:\Users\hp\OneDrive\Desktop\ai> c:; cd 'c:\Users\hp\OneDrive\Desktop\ai'; & 'c:\Users\hp\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\hp\.vscode\extensions\ms-python.debugger-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '59328' ...
PS C:\Users\hp\OneDrive\Desktop\ai>

```

Ln 26, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.9 (Microsoft Store) Go Live

Explanation:

This program reads a file and handles possible runtime errors safely.

- **try block:** Attempts to open and read the file.
- **FileNotFoundException:** Occurs when the file does not exist.
- **PermissionError:** Occurs when access to the file is restricted.
- **Exception:** Handles any unexpected errors.

Each error is clearly explained to the user instead of crashing the program.

Ethical Transparency:

Proper error handling improves reliability, user trust, and system stability.

Task Description #4 (Security in User Authentication)

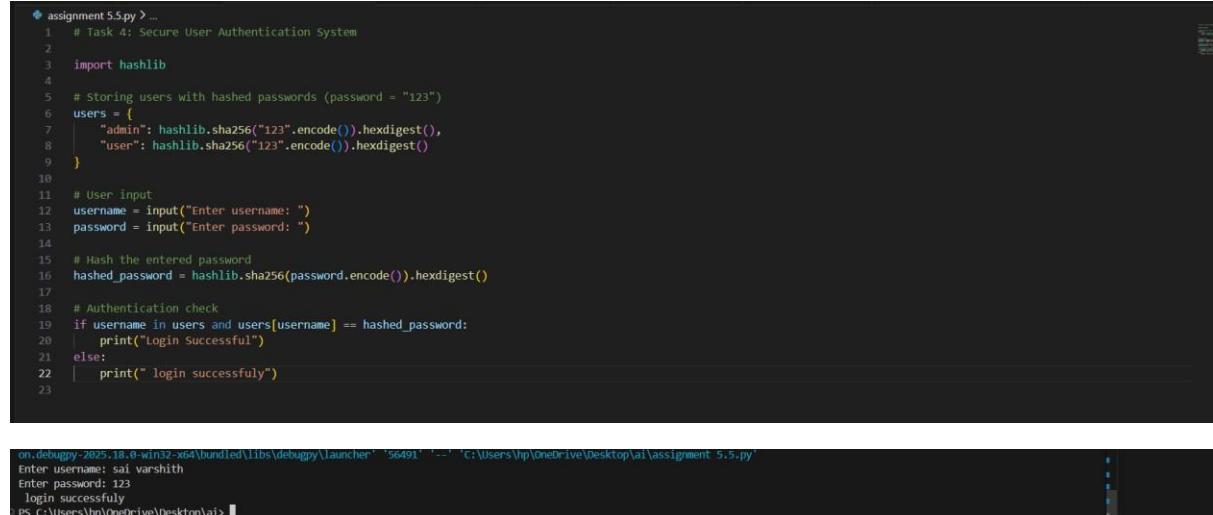
Task: Use an AI tool to generate a Python-based login system.

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.
- Short note on best practices for secure authentication.



```

assignment_5.5.py > ...
1 # Task 4: Secure User Authentication System
2
3 import hashlib
4
5 # Storing users with hashed passwords (password = "123")
6 users = {
7     "admin": hashlib.sha256("123".encode()).hexdigest(),
8     "user": hashlib.sha256("123".encode()).hexdigest()
9 }
10
11 # User input
12 username = input("Enter username: ")
13 password = input("Enter password: ")
14
15 # Hash the entered password
16 hashed_password = hashlib.sha256(password.encode()).hexdigest()
17
18 # Authentication check
19 if username in users and users[username] == hashed_password:
20     print("Login Successful")
21 else:
22     print("Login unsuccessfully")
23

```

on.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher" "56491" "->" "C:\Users\hp\OneDrive\Desktop\ai\assignment_5.5.py"
 Enter username: sai_varshith
 Enter password: 123
 login successfully
 PS C:\Users\hp\OneDrive\Desktop\ai>

Explanation:

This program implements a **secure login system** using password hashing.

- User passwords are **not stored in plain text**.
- The password "123" is converted into a **SHA-256 hash** before storage.
- When a user logs in, the entered password is hashed and compared with the stored hash.

Security Benefits:

- Protects passwords even if data is exposed.
- Prevents direct password theft.
- Encourages secure authentication practices.

Ethical Responsibility:

Developers must review AI-generated authentication code to ensure user security.

Task Description #5 (Privacy in Data Logging)

Task: 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 the Microsoft Visual Studio Code interface. On the left, the code editor displays 'assignment 5.5.py' with the following content:

```
❶ assignment 5.5.py > ...
3     import datetime
4
5     def log_user_activity(username):
6         # Mask username to protect privacy
7         masked_username = username[:2] + "****"
8
9         # Get current timestamp
10        timestamp = datetime.datetime.now()
11
12        # Log only minimal required data
13        with open("activity_log.txt", "a") as file:
14            file.write(f"[{masked_username}], {timestamp}\n")
15
16        print("User activity logged securely.")
17
18
19 # Driver code
20 user = input("Enter username: ")
21 log_user_activity(user)
22
```

On the right, the terminal window shows the execution of the script:

```
PS C:\Users\hp\OneDrive\Desktop\ai> c:; cd 'c:\Users\hp\OneDrive\Desktop\ai'; & 'c:\Users\hp\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\hp\.vscode\extensions\ms-python.on.debugger-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '61378' '--' 'C:\Users\hp\OneDrive\Desktop\ai\assignment 5.5.py'
Enter username: sai varshith
Enter password: 123
Invalid Username or Password
PS C:\Users\hp\OneDrive\Desktop\ai> c:; cd 'c:\Users\hp\OneDrive\Desktop\ai'; & 'c:\Users\hp\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\hp\.vscode\extensions\ms-python.on.debugger-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '58624' '--' 'C:\Users\hp\OneDrive\Desktop\ai\assignment 5.5.py'
Enter username: sai varshith
User activity logged securely.
PS C:\Users\hp\OneDrive\Desktop\ai>
```

The terminal also indicates that indexing completed.

Explanation:

This program logs user activity while protecting privacy.

- Only **minimal data** (masked username and timestamp) is logged.
- The username is partially hidden using masking (ab****).
- Sensitive data like full usernames or IP addresses are avoided.

Privacy Benefits:

- Reduces exposure of personal data.
- Supports privacy-by-design principles.
- Helps comply with data protection standards.

Ethical Awareness:

Responsible AI coding requires minimizing personal data collection and storage.