

AI ASSISTED CODING

ASSIGNMENT – 2.4

Name: V. Vyshnavi

Hallticket No. : 2303A51968

Batch-14

Task 1: Use Cursor AI to generate a Python class Book with attributes title, author, and a summary () method.

Prompt : “Generate a Python class named Book with attributes title, author, and a method summary() that returns a formatted string with the title and author.”

Code and output :

The screenshot shows the Microsoft Visual Studio Code interface. In the center, there is a code editor window titled "2.4.1.py" with the following Python code:

```
#Write a Python program to check whether Untitled-2
class Book:
    def __init__(self, title, author):
        self.title = title
        self.author = author

    def summary(self):
        return f'{self.title} is written by {self.author}'
```

Below the code editor, the terminal window shows the following command-line interaction:

```
PS C:\Users\sathw\Documents\AI-Assisted-Coding> & 'c:\Users\sathw\AppData\Local\Microsoft\WindowsApp
s\python3.13.exe' 'c:\Users\sathw\vscode\extensions\ms-python.debug-2025.18.0-win32-x64\bundled\li
bs\debugpy\launcher' 6225 '--' 'c:\Users\sathw\Documents\AI-Assisted-Coding\2.4.1.py'
'To Kill a Mockingbird' is written by Harper Lee.
PS C:\Users\sathw\Documents\AI-Assisted-Coding> <
PS C:\Users\sathw\Documents\AI-Assisted-Coding> <
PS C:\Users\sathw\Documents\AI-Assisted-Coding> <cd 'c:\Users\sathw\Documents\AI-Assisted-Coding'
; & 'c:\Users\sathw\AppData\Local\Microsoft\WindowsApps\python3.13.exe' 'c:\Users\sathw\vscode\exten
sions\ms-python.debug-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' 49681 '--' 'c:\Users\sat
hw\Documents\AI-Assisted-Coding\2.4.2.py'
[{"name": "Bobby", "age": 28}, {"name": "Ashok", "age": 25}, {"name": "mohan", "age": 30}]
PS C:\Users\sathw\Documents\AI-Assisted-Coding> <
PS C:\Users\sathw\Documents\AI-Assisted-Coding> <
```

The status bar at the bottom indicates the file is saved (S), the current line is line 8, column 62, and the file size is 3.13.9 (Microsoft Store). The system tray shows it's 14:46, 19-01-2026, and the weather is 29°C, Sunny.

Justification:

Object-Oriented Programming helps model real-world entities like books using classes and objects

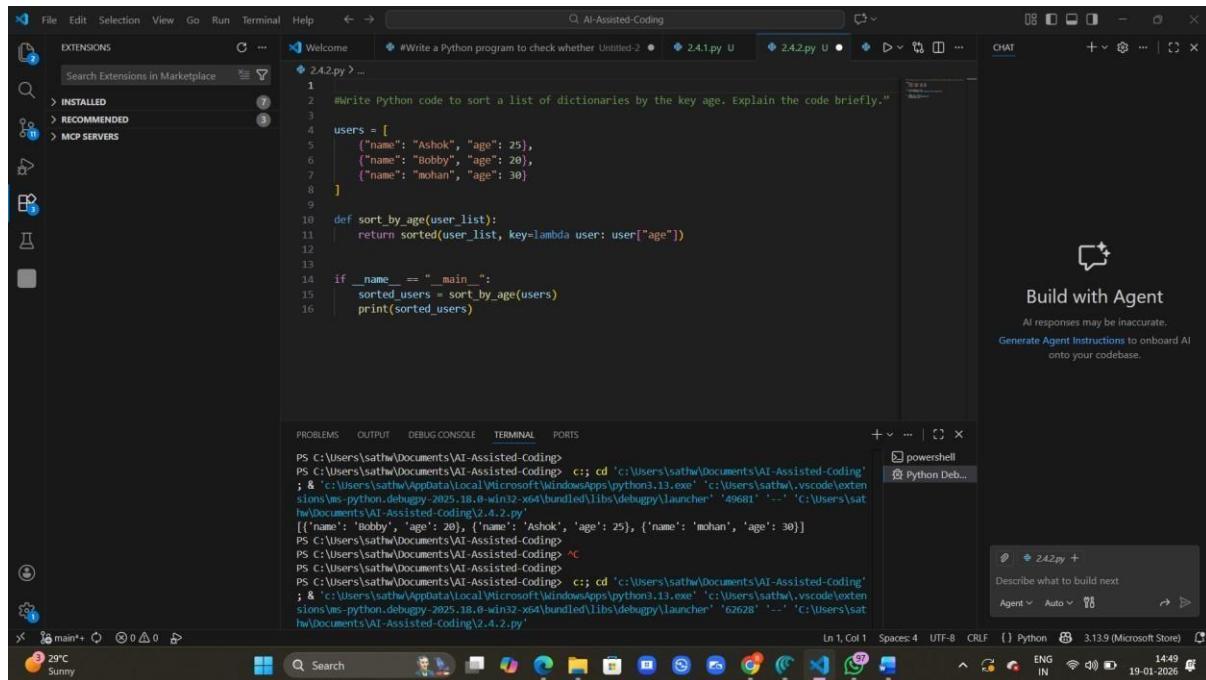
Encapsulation allows data (title, author) and behavior (summary) to be bundled together logically.

This approach improves code organization, scalability, and ease of maintenance in software systems.

Task 2: Use Gemini and Cursor AI to generate code that sorts a list of dictionaries by a key.

Prompt: Write Python code to sort a list of dictionaries by the key age. Explain the code briefly.

Code and Output :



The screenshot shows the Microsoft Visual Studio Code interface with the "AI-Assisted Coding" extension open. The code editor displays a Python script named 24.2.py. The code defines a function to sort a list of dictionaries based on the 'age' key. The terminal below shows the command to run the script and its output, which is a sorted list of users. A sidebar on the right provides AI assistance, including a "Build with Agent" section and a status bar at the bottom indicating the file is 3.13.9 (Microsoft Store).

```
#Write Python code to sort a list of dictionaries by the key age. Explain the code briefly.
users = [
    {"name": "Ashok", "age": 25},
    {"name": "Bobby", "age": 20},
    {"name": "mohan", "age": 30}
]
def sort_by_age(user_list):
    return sorted(user_list, key=lambda user: user["age"])
if __name__ == "__main__":
    sorted_users = sort_by_age(users)
    print(sorted_users)
```

```
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
PS C:\Users\sathw\Documents\AI-Assisted-Coding> cd "c:\Users\sathw\Documents\AI-Assisted-Coding"
; & "c:\Users\sathw\AppData\Local\Microsoft\WindowsApps\python3.13.exe" "c:\Users\sathw\.vscode\extensions\ms-python.python.debug-2025.10.0-win32-x64\bundled\libs\debug\launcher" "49681" "-d" "C:\Users\sathw\Documents\AI-Assisted-Coding\2.4.2.py"
[{"name": "Bobby", "age": 20}, {"name": "Ashok", "age": 25}, {"name": "mohan", "age": 30}]
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
PS C:\Users\sathw\Documents\AI-Assisted-Coding> cd "c:\Users\sathw\Documents\AI-Assisted-Coding"
; & "c:\Users\sathw\AppData\Local\Microsoft\WindowsApps\python3.13.exe" "c:\Users\sathw\.vscode\extensions\ms-python.python.debug-2025.10.0-win32-x64\bundled\libs\debug\launcher" "62628" "-d" "C:\Users\sathw\Documents\AI-Assisted-Coding\2.4.2.py"
```

Justification:

Gemini's code is more concise and easy to understand for quick tasks.

Cursor's approach is slightly more structured and reusable due to function encapsulation.

Task 3: Ask Gemini to generate a calculator using functions and explain how it works.

Prompt: Write a Python calculator program using separate functions for add, subtract, multiply, and divide. Then explain how the program works step by step.

Code and Output:

```

File Edit Selection View Go Run Terminal Help < > AI-Assisted-Coding
EXTENSIONS Search Extensions in Marketplace
INSTALLED RECOMMENDED MCP SERVERS
2.4.3.py > ...
1 #Write a Python program to check whether Untitled-2 • 2.4.1.py U 2.4.2.py U 2.4.3.py U ...
2 |
3 def add(a, b):
4     return a + b
5
6 def subtract(a, b):
7     return a - b
8
9 def multiply(a, b):
10    return a * b
11
12 def divide(a, b):
13     if b == 0:
14         return "Error: Division by zero"
15     return a / b
16
17
18 if __name__ == "__main__":
19     a = float(input("Enter first number: "))
20     b = float(input("Enter second number: "))
21
22     print("Addition:", add(a, b))
23     print("Subtraction:", subtract(a, b))

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS C:\Users\sathw\Documents\AI-Assisted-Coding> ^
PS C:\Users\sathw\Documents\AI-Assisted-Coding>
PS C:\Users\sathw\Documents\AI-Assisted-Coding> c;; cd 'c:\Users\sathw\Documents\AI-Assisted-Coding'
2 & cd 'c:\Users\sathw\appdata\local\Microsoft\WindowsApps\python3.11.exe' 'c:\Users\sathw\vscode\exten
sions\pythonda\debug-2025.18.0-win32-x64\bundled\11bs\debug\launcher' '58832' '--' 'c:\Users\sat
hw\Documents\AI-Assisted-Coding\2.4.3.py'
Enter first number: 12
Enter second number: 3
Addition: 15.0
Subtraction: 9.0
Multiplication: 36.0
Division: 4.0
PS C:\Users\sathw\Documents\AI-Assisted-Coding> def is_armstrong(num):

```

Ln 2, Col 1 Spaces: 4 UTF-8 CRLF ENG IN 1452 19-01-2026

Justification:

Each arithmetic operation is written as a separate function, improving modularity.

Functions take two inputs, perform the operation, and return the result.

Division includes a safety check to prevent division by zero errors.

Task 4: Generate an Armstrong number program using Gemini, then improve it using Cursor AI.

Prompt: Write a Python program to check whether a given number is an Armstrong number. Use basic Python constructs and explain briefly.

Code and Input:

```
#write a Python program to check whether a given number is an Armstrong number. Use basic Python functions.
def is_armstrong(num):
    digits = str(num)
    power = len(digits)
    total = sum(int(digit) ** power for digit in digits)
    return total == num

if __name__ == "__main__":
    number = int(input("Enter a number: "))

    if is_armstrong(number):
        print("Armstrong Number")
    else:
        print("Not an Armstrong Number")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Enter a number: 153
Not an Armstrong Number

PS C:\Users\sathw\Documents\AI-Assisted-Coding> ^C

PS C:\Users\sathw\Documents\AI-Assisted-Coding> PS C:\Users\sathw\Documents\AI-Assisted-Coding> c; cd 'c:\Users\sathw\Documents\AI-Assisted-Coding'; & 'c:\Users\sathw\AppData\Local\Microsoft\WindowsApps\python3.11.exe' 'c:\Users\sathw\.vscode\extensions\ms-python.python.debug-2025.18.0.win32-x64\bundled\libs\debugpy\launcher' '50099' '--' 'c:\Users\sathw\Documents\AI-Assisted-Coding\2.4.4.py'

Enter a number: 153
Armstrong Number

PS C:\Users\sathw\Documents\AI-Assisted-Coding> ^C

PS C:\Users\sathw\Documents\AI-Assisted-Coding> PS C:\Users\sathw\Documents\AI-Assisted-Coding> c; cd 'c:\Users\sathw\Documents\AI-Assisted-Coding'

Justification:

The optimized version avoids string conversion, reducing unnecessary overhead.

It uses arithmetic operations, which are more efficient and closer to low-level computation.

The logic is clearer for understanding numeric processing and improves overall performance.