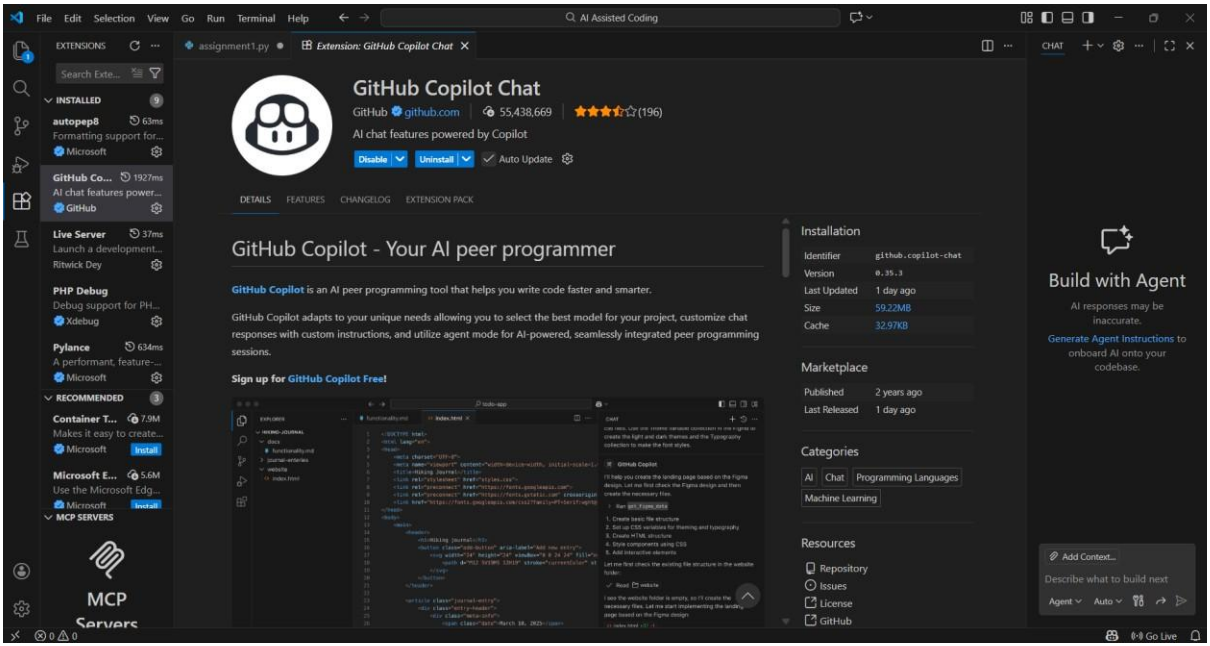
ASSIGNMENT – 1.3

D.Charmi

2303A51314

Batch - 05

Task – 0 : Installation of GitHub Copilot


File Edit Selection View Go Run Terminal Help
AI Assisted Coding
EXTENSIONS
INSTALLED
autopep8
GitHub Co...
Live Server
PHP Debug
Pylance
RECOMMENDED
Container T...
Microsoft E...
MCP SERVERS
MCP Servers
assignment1.py
Extension: GitHub Copilot Chat
Disable
Uninstall
Auto Update
DETAILS FEATURES CHANGELOG EXTENSION PACK
CHAT
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Add Context...
Describe what to build next
Agent Auto

GitHub Copilot Chat

GitHub github.com | 55,438,669 | (196)

AI chat features powered by Copilot

GitHub Copilot - Your AI peer programmer

GitHub Copilot is an AI peer programming tool that helps you write code faster and smarter.

GitHub Copilot adapts to your unique needs allowing you to select the best model for your project, customize chat responses with custom instructions, and utilize agent mode for AI-powered, seamlessly integrated peer programming sessions.

Sign up for GitHub Copilot Free!

Installation

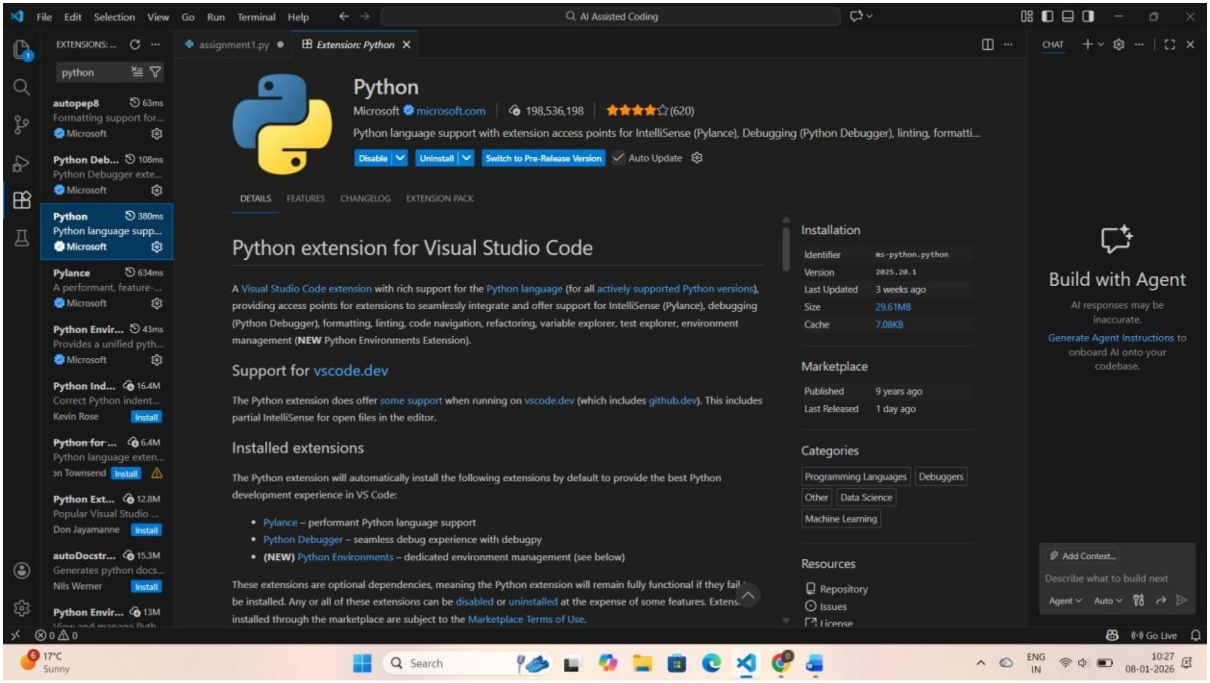
Identifier	github.copilot-chat
Version	0.35.3
Last Updated	1 day ago
Size	59.22MB
Cache	32.97KB

Marketplace

Published	2 years ago
Last Released	1 day ago

Categories: AI, Chat, Programming Languages, Machine Learning

Resources: Repository, Issues, License, GitHub



Python

Microsoft microsoft.com | 198,536,198 | (620)

Python language support with extension access points for IntelliSense (Pylance), Debugging (Python Debugger), linting, formatti...

Python extension for Visual Studio Code

A Visual Studio Code extension with rich support for the Python language (for all actively supported Python versions), providing access points for extensions to seamlessly integrate and offer support for IntelliSense (Pylance), debugging (Python Debugger), formatting, linting, code navigation, refactoring, variable explorer, test explorer, environment management (**NEW** Python Environments Extension).

Support for vscode.dev

The Python extension does offer some support when running on vscode.dev (which includes github.dev). This includes partial IntelliSense for open files in the editor.

Installed extensions

The Python extension will automatically install the following extensions by default to provide the best Python development experience in VS Code:

- Pylance – performant Python language support
- Python Debugger – seamless debug experience with debugpy
- **(NEW)** Python Environments – dedicated environment management (see below)

These extensions are optional dependencies, meaning the Python extension will remain fully functional if they fail to be installed. Any or all of these extensions can be disabled or uninstalled at the expense of some features. Extens... installed through the marketplace are subject to the Marketplace Terms of Use.

Installation

Identifier	ms-python.python
Version	2025.20.1
Last Updated	3 weeks ago
Size	29.61MB
Cache	7.08KB

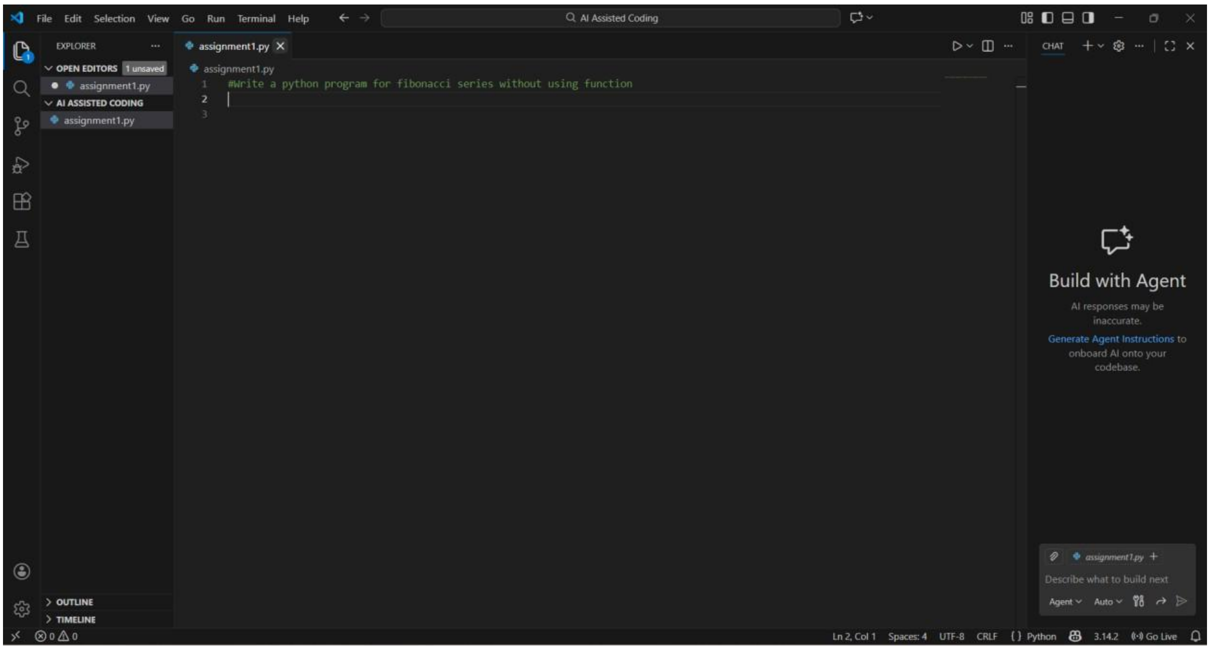
Marketplace

Published	9 years ago
Last Released	1 day ago

Categories: Programming Languages, Debuggers, Other, Data Science, Machine Learning

Resources: Repository, Issues, License

Task-1: Prompt



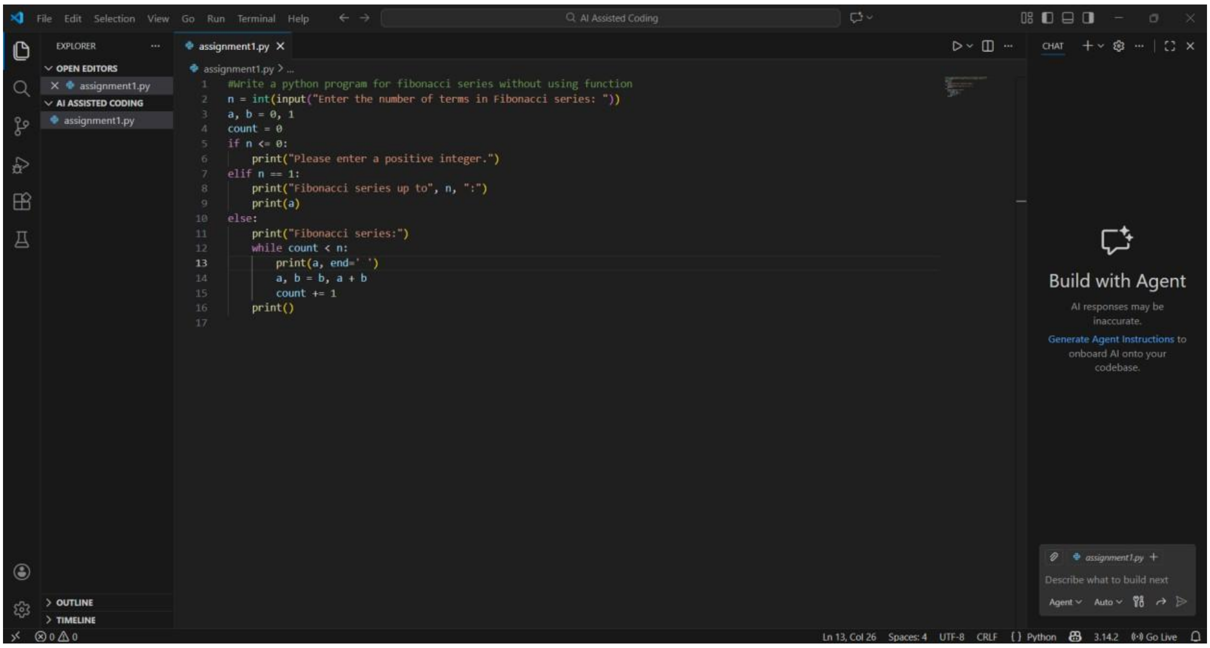
```python
#Write a python program for fibonacci series without using function

```

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

Describe what to build next


File Edit Selection View Go Run Terminal Help
AI Assisted Coding
EXPLORER
OPEN EDITORS
assignment1.py
AI ASSISTED CODING
assignment1.py
OUTLINE
TIMELINE
CHAT
Ln 13, Col 26 Spaces: 4 UTF-8 CRLF {} Python 3.14.2 Go Live

```python
#Write a python program for fibonacci series without using function
n = int(input("Enter the number of terms in Fibonacci series: "))
a, b = 0, 1
count = 0
if n <= 0:
    print("Please enter a positive integer.")
elif n == 1:
    print("Fibonacci series up to", n, ":")
    print(a)
else:
    print("Fibonacci series:")
    while count < n:
        print(a, end=' ')
        a, b = b, a + b
        count += 1
    print()

```

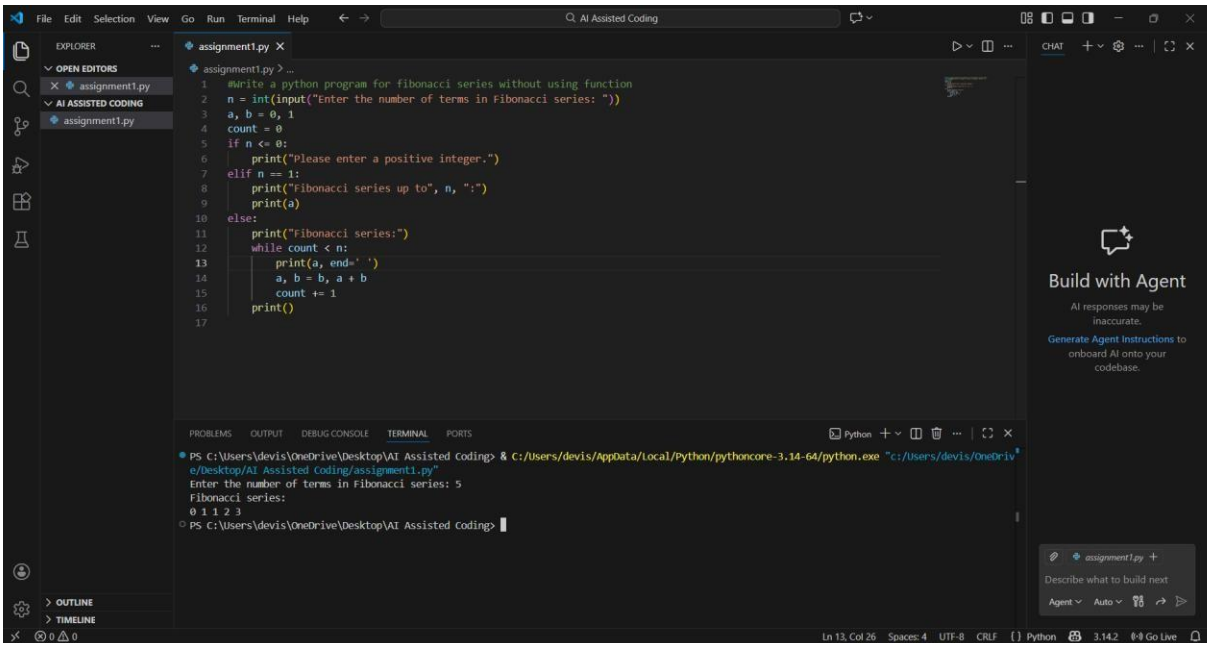
Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

Describe what to build next

Output:



File Edit Selection View Go Run Terminal Help
AI Assisted Coding
EXPLORER
OPEN EDITORS
assignment1.py
AI ASSISTED CODING
assignment1.py
OUTLINE
TIMELINE
assignment1.py
CHAT
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Describe what to build next
Agent
Auto
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
Python
Ln 13, Col 26 Spaces: 4 UTF-8 CRLF {} Python 3.14.2 Go Live

```python
#Write a python program for fibonacci series without using function
n = int(input("Enter the number of terms in Fibonacci series: "))
a, b = 0, 1
count = 0
if n <= 0:
    print("Please enter a positive integer.")
elif n == 1:
    print("Fibonacci series up to", n, ":")
    print(a)
else:
    print("Fibonacci series:")
    while count < n:
        print(a, end=' ')
        a, b = b, a + b
        count += 1
    print()
```

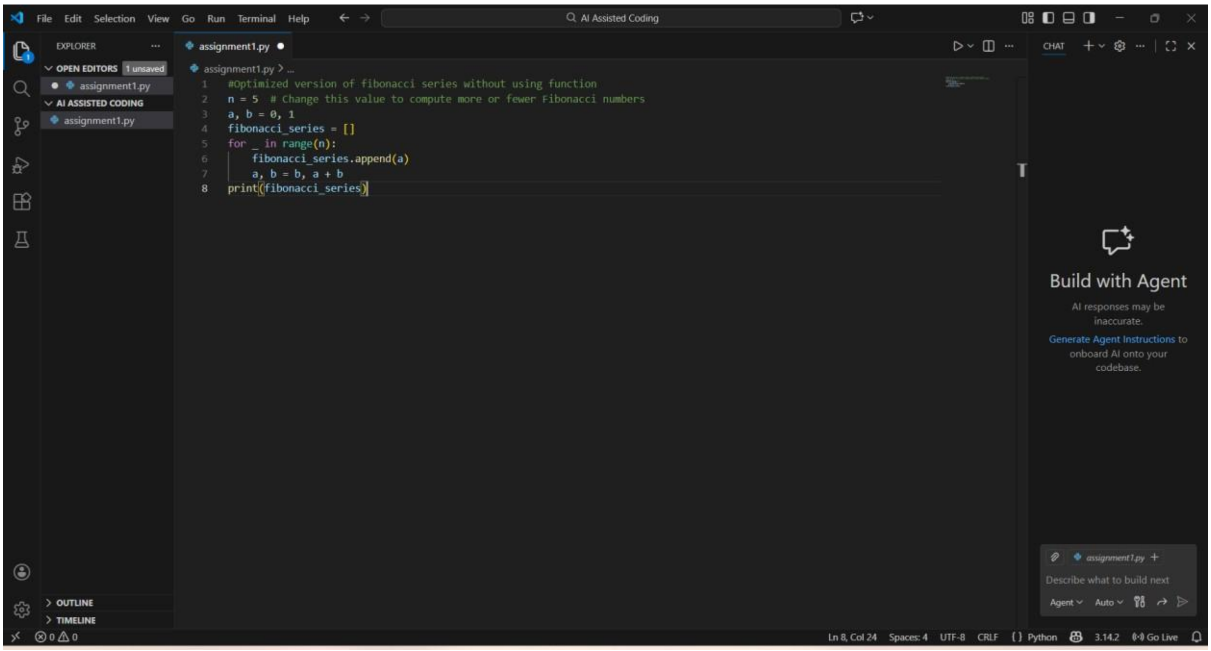
```
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding> & C:/Users/devis/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/Users/devis/OneDriv
e/Desktop/AI Assisted Coding/assignment1.py"
Enter the number of terms in Fibonacci series: 5
Fibonacci series:
0 1 1 2 3
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding>
```

Explanation :

This program generates the Fibonacci series up to 'n' terms without using any functions. It first takes an integer input 'n' from the user, which represents the number of terms to be printed. It initializes two variables 'a' and 'b' to 0 and 1, which are the first two terms of the Fibonacci series. A counter variable 'count' is also initialized to keep track of the number of terms printed. The program checks if 'n' is less than or equal to 0, in which case it prompts the user to enter a positive integer. If 'n' is 1, it prints only the first term of the series (0). For values of 'n' greater than 1, it enters a while loop that continues until 'count' is less than 'n'. Inside the loop, it prints the current value of 'a', then updates 'a' and 'b' to the next two terms in the series. The values of 'a' and 'b' are updated using tuple unpacking: 'a' takes the value of 'b', and 'b' takes the sum of the previous 'a' and 'b'. The counter 'count' is incremented by 1 in each iteration. Finally, it prints the complete Fibonacci series up to 'n' terms.

Task-2

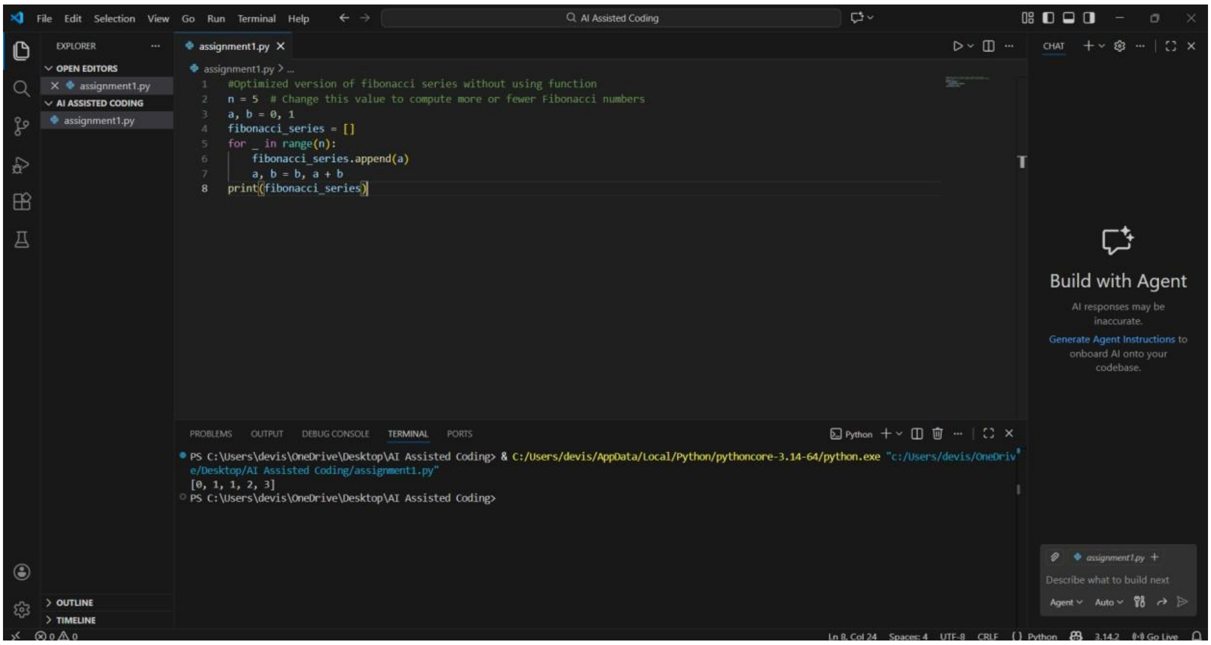
Prompt : #Optimized version of fibonacci series without using function



```python
#Optimized version of fibonacci series without using function
n = 5  # Change this value to compute more or fewer Fibonacci numbers
a, b = 0, 1
fibonacci_series = []
for _ in range(n):
    fibonacci_series.append(a)
    a, b = b, a + b
print(fibonacci_series)
```

File Edit Selection View Go Run Terminal Help
AI Assisted Coding
EXPLORER
OPEN EDITORS 1 unsaved
assignment1.py
AI ASSISTED CODING
OUTLINE
TIMELINE
CHAT
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Describe what to build next
Agent Auto
Ln 8, Col 24 Spaces: 4 UTF-8 CRLF {} Python 3.14.2 Go Live

Output :



```python
#Optimized version of fibonacci series without using function
n = 5  # Change this value to compute more or fewer Fibonacci numbers
a, b = 0, 1
fibonacci_series = []
for _ in range(n):
    fibonacci_series.append(a)
    a, b = b, a + b
print(fibonacci_series)
```

```
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding> & C:/Users/devis/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/Users/devis/OneDriv
e/Desktop/AI Assisted Coding/assignment1.py"
[0, 1, 1, 2, 3]
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding>
```

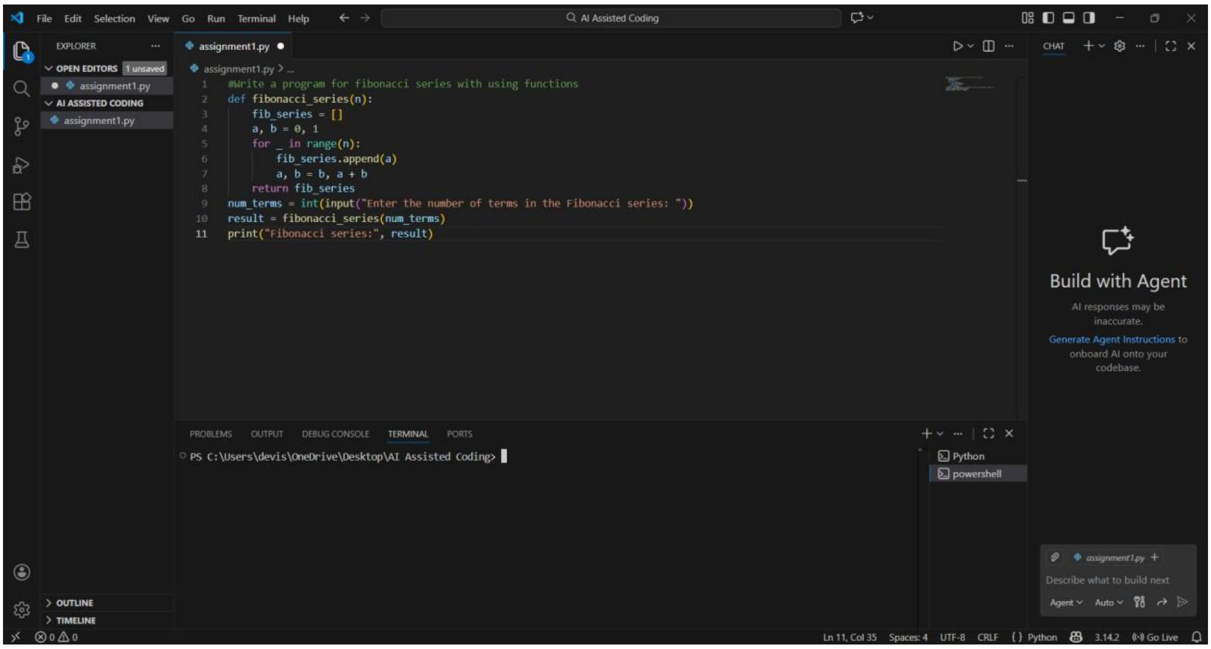
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python
Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Describe what to build next

Explanation : # This code generates the first 'n' numbers in the Fibonacci series using an iterative approach.

It initializes two variables 'a' and 'b' to represent the first two Fibonacci numbers (0 and 1). In each iteration of the loop, it appends the current value of 'a' to the list and then updates 'a' and 'b' to the next two numbers in the series. This method is efficient in terms of both time and space complexity.

Task – 3:

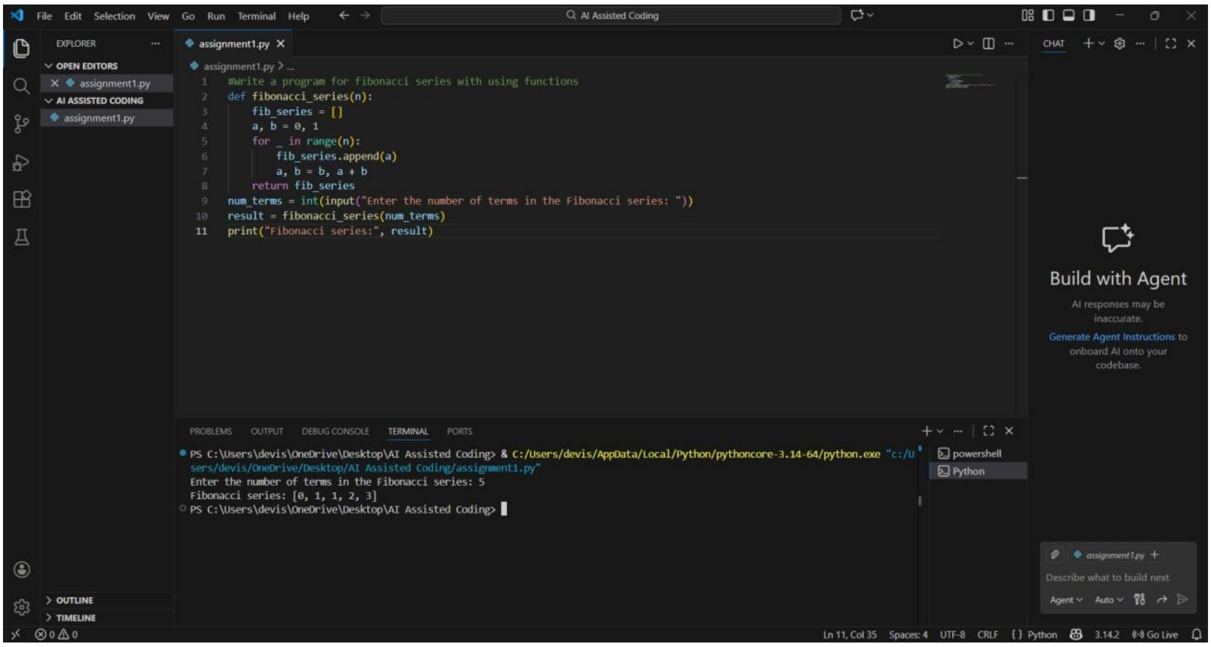
Prompt : #Write a program for fibonacci series with using functions



```python
#Write a program for fibonacci series with using functions
def fibonacci_series(n):
    fib_series = []
    a, b = 0, 1
    for _ in range(n):
        fib_series.append(a)
        a, b = b, a + b
    return fib_series
num_terms = int(input("Enter the number of terms in the Fibonacci series: "))
result = fibonacci_series(num_terms)
print("Fibonacci series:", result)
```

```
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding>
```

Output :



```python
#Write a program for fibonacci series with using functions
def fibonacci_series(n):
    fib_series = []
    a, b = 0, 1
    for _ in range(n):
        fib_series.append(a)
        a, b = b, a + b
    return fib_series
num_terms = int(input("Enter the number of terms in the Fibonacci series: "))
result = fibonacci_series(num_terms)
print("Fibonacci series:", result)
```

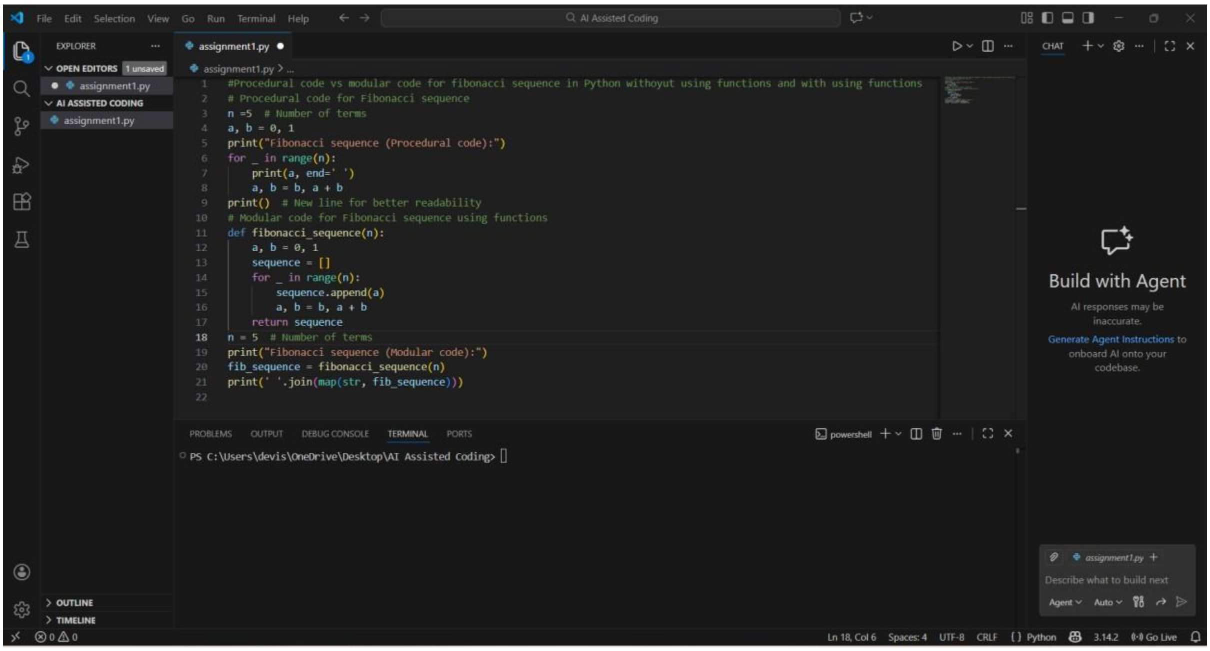
```
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding> & C:/Users/devis/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/U
sers/devis/OneDrive/Desktop/AI Assisted Coding/assignment1.py"
Enter the number of terms in the Fibonacci series: 5
Fibonacci series: [0, 1, 1, 2, 3]
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding>
```

Explanation :

This program defines a function called fibonacci_series that takes an integer n as input and returns a list containing the first n terms of the Fibonacci series. Inside the function, we initialize an empty list fib_series to store the Fibonacci numbers. We also initialize two variables a and b to represent the first two numbers in the series (0 and 1). We then use a for loop to iterate n times. In each iteration, we append the current value of a to the fib_series list, and then update a and b to the next two numbers in the series using tuple unpacking. After defining the function, we prompt the user to input the number of terms they want in the Fibonacci series. We convert this input to an integer and store it in the variable num_terms. We then call the fibonacci_series function with num_terms as the argument and store the result in the variable result. Finally, we print the resulting Fibonacci series

Task – 4 :

Prompt : #Procedural code vs modular code for fibonacci sequence in Python without using functions and with using functions



```python
#Procedural code vs modular code for fibonacci sequence in Python withoyut using functions and with using functions
# Procedural code for Fibonacci sequence
n =5  # Number of terms
a, b = 0, 1
print("Fibonacci sequence (Procedural code):")
for _ in range(n):
    print(a, end=' ')
    a, b = b, a + b
print()  # New line for better readability
# Modular code for Fibonacci sequence using functions
def fibonacci_sequence(n):
    a, b = 0, 1
    sequence = []
    for _ in range(n):
        sequence.append(a)
        a, b = b, a + b
    return sequence
n = 5  # Number of terms
print("Fibonacci sequence (Modular code):")
fib_sequence = fibonacci_sequence(n)
print(' '.join(map(str, fib_sequence)))

```

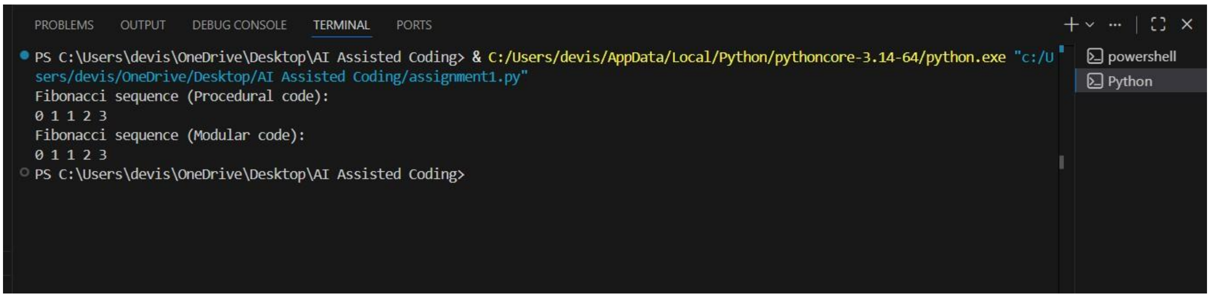
```
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding> 
```

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

Describe what to build next

Output :



```
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding> & C:/Users/devis/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/U
sers/devis/OneDrive/Desktop/AI Assisted Coding/assignment1.py"
Fibonacci sequence (Procedural code):
0 1 1 2 3
Fibonacci sequence (Modular code):
0 1 1 2 3
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding>
```

Explanation :

1. Code Clarity:

- Procedural Code: The logic is straightforward but can become cluttered as the program grows.

- Modular Code: The use of functions makes the code more organized and easier to read. Each function has a specific purpose.

2. Reusability:

- Procedural Code: The code is not reusable. If you need to generate the Fibonacci sequence in another part of the program, you would have to duplicate the code.

- Modular Code: The function can be reused anywhere in the program or in other programs without rewriting the logic.

3. Debugging:

- Procedural Code: Debugging can be more challenging as the entire logic is in one place. Identifying issues may require going through the whole code.

- Modular Code: Debugging is easier since you can isolate issues within specific functions. You can test functions independently.

4. Maintainability:

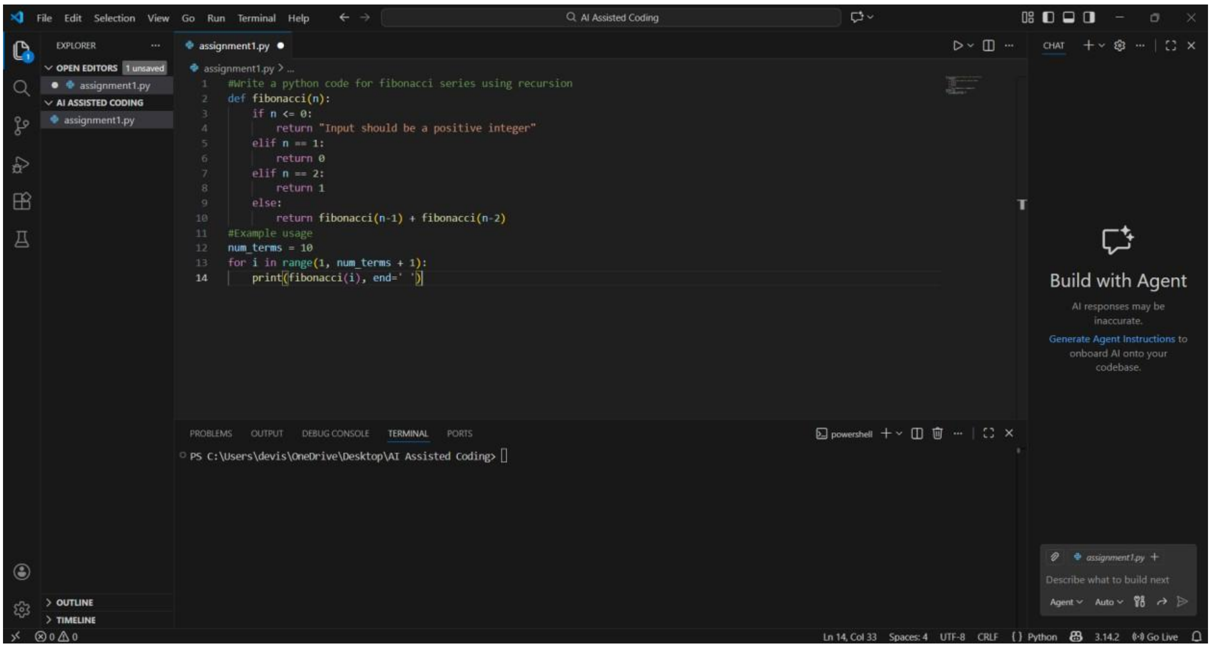
- Procedural Code: Maintaining the code can be difficult as changes may require modifications in multiple places.

- Modular Code: The code is easier to maintain. Changes can be made within functions without affecting the overall structure of the program.

Overall, modular code is generally preferred for larger and more complex programs due to its advantages in clarity, reusability, debugging, and maintainability.

Task – 5:

Prompt : #Write a python code for fibonacci series using recursion

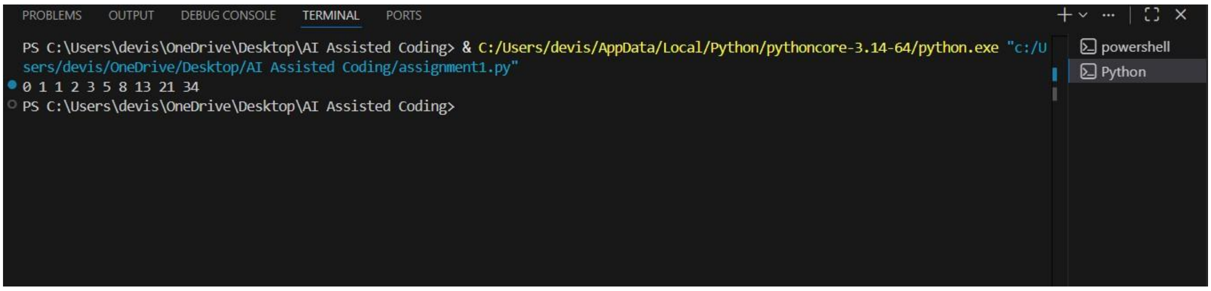


```python
#Write a python code for fibonacci series using recursion
def fibonacci(n):
    if n <= 0:
        return "Input should be a positive integer"
    elif n == 1:
        return 0
    elif n == 2:
        return 1
    else:
        return fibonacci(n-1) + fibonacci(n-2)
#Example usage
num_terms = 10
for i in range(1, num_terms + 1):
    print(fibonacci(i), end=' ')
```

```
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding>
```

Build with Agent
AI responses may be inaccurate.
Generate Agent Instructions to onboard AI onto your codebase.
Describe what to build next
Ln 14, Col 33 Spaces: 4 UTF-8 CRLF Python 3.14.2 Go Live

Output :



```
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding> & C:/Users/devis/AppData/Local/Python/pythoncore-3.14-64/python.exe "c:/U
sers/devis/OneDrive/Desktop/AI Assisted Coding/assignment1.py"
0 1 1 2 3 5 8 13 21 34
PS C:\Users\devis\OneDrive\Desktop\AI Assisted Coding>
```

Explanation :

The fibonacci function takes a positive integer n as input and returns the nth term in the Fibonacci series. The base cases are defined for n = 1 and n = 2, returning 0 and 1 respectively. For any n greater than 2, the function calls itself recursively to calculate the sum of the two preceding terms in the series (fibonacci(n-1) + fibonacci(n-2)).The example usage demonstrates how to print the first 10 terms of the Fibonacci series by calling the fibonacci function in a loop.