Assignment

Tasks to be completed are as follows:

1. Setup AI Coding Tool:

- Install and configure GitHub Copilot or Kite with VS Code or JetBrains IDE.
- Enable real-time code suggestions.

2. Class Design Using AI Assistance:

- Begin defining a `Product` class with attributes: **name, price, quantity**.
- Use the AI suggestion feature to automatically complete the `__init__()` method.
- Add a method `calculate_value()` to return **price × quantity**.

3. Create Another Class:

- Define a `Warehouse` class with a list of `Product` objects.
- Use code suggestions to help implement:
  - A method to add a product.
  - A method to display the total value of all products.

4. Reflection:

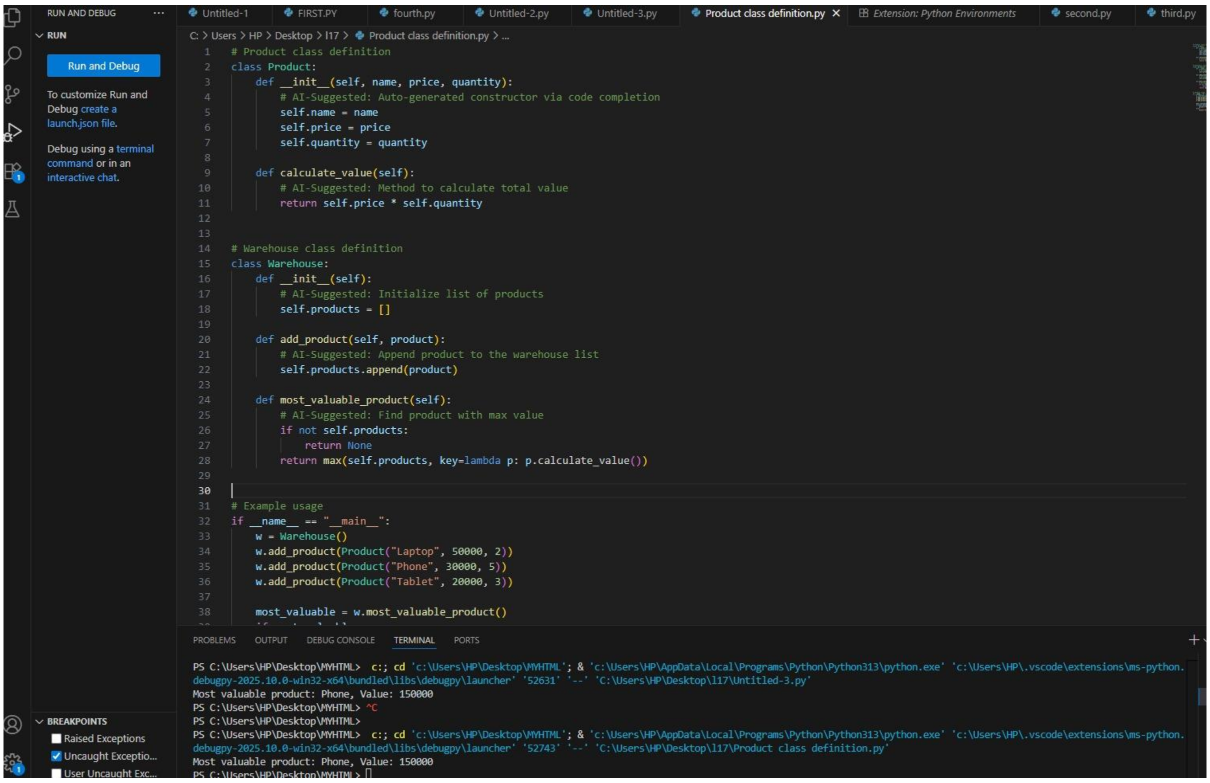
- Identify how much of the code was completed by AI and what manual edits were needed.
- Comment on the relevance and accuracy of AI suggestions.

5. Presentation:

- VS Code with GitHub Copilot or Cursor API and/or Google Colab with Gemini.

Deliverables:

1. Python script with both classes and comments on AI-generated suggestions.
2. Short report (1 page) summarizing your experience with AI code completion.


RUN AND DEBUG
RUN
Run and Debug
To customize Run and Debug create a launch.json file.
Debug using a terminal command or in an interactive chat.
BREAKPOINTS
Raised Exceptions
Uncaught Exceptio...
User Uncaught Exc...
Untitled-1
FIRST.PY
fourth.py
Untitled-2.py
Untitled-3.py
Product class definition.py
Extension: Python Environments
second.py
third.py
C: > Users > HP > Desktop > l17 > Product class definition.py > ...
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS

```python
# Product class definition
class Product:
    def __init__(self, name, price, quantity):
        # AI-Suggested: Auto-generated constructor via code completion
        self.name = name
        self.price = price
        self.quantity = quantity

    def calculate_value(self):
        # AI-Suggested: Method to calculate total value
        return self.price * self.quantity


# Warehouse class definition
class Warehouse:
    def __init__(self):
        # AI-Suggested: Initialize list of products
        self.products = []

    def add_product(self, product):
        # AI-Suggested: Append product to the warehouse list
        self.products.append(product)

    def most_valuable_product(self):
        # AI-Suggested: Find product with max value
        if not self.products:
            return None
        return max(self.products, key=lambda p: p.calculate_value())


# Example usage
if __name__ == "__main__":
    w = Warehouse()
    w.add_product(Product("Laptop", 50000, 2))
    w.add_product(Product("Phone", 30000, 5))
    w.add_product(Product("Tablet", 20000, 3))

    most_valuable = w.most_valuable_product()
```

```
PS C:\Users\HP\Desktop\MYHTML>  c:; cd 'c:\Users\HP\Desktop\MYHTML'; & 'c:\Users\HP\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\HP\.vscode\extensions\ms-python.
debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '52631' '--' 'C:\Users\HP\Desktop\l17\Untitled-3.py'
Most valuable product: Phone, Value: 150000
PS C:\Users\HP\Desktop\MYHTML> ^C
PS C:\Users\HP\Desktop\MYHTML>
PS C:\Users\HP\Desktop\MYHTML>  c:; cd 'c:\Users\HP\Desktop\MYHTML'; & 'c:\Users\HP\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\HP\.vscode\extensions\ms-python.
debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '52743' '--' 'C:\Users\HP\Desktop\l17\Product class definition.py'
Most valuable product: Phone, Value: 150000
PS C:\Users\HP\Desktop\MYHTML> 
```