NAME:P Pranay Kumar H.NO: 2303A51829 BATCH:26

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING**	
Program Name: B. Tech	**Assignment Type: Lab**		**Academic Year:**2025-2026
Course Coordinator Name	Dr. Rishabh Mittal		
Instructor(s) Name	Mr. S Naresh Kumar Ms. B. Swathi Dr. Sasanko Shekhar Gantayat Mr. Md Sallauddin Dr. Mathivanan Mr. Y Srikanth Ms. N Shilpa Dr. Rishabh Mittal (Coordinator) Dr. R. Prashant Kumar Mr. Ankushavali MD Mr. B Viswanath Ms. Sujitha Reddy Ms. A. Anitha Ms. M.Madhuri Ms. Katherashala Swetha Ms. Velpula sumalatha Mr. Bingi Raju		
Course Code	23CS002PC304	**Course Title**	AI Assisted Coding
Year/Sem	III/I	**Regulation**	R23
Date and Day of Assignment	Week 2 - Wednesday	**Time(s)**	23CSBTB01 To 23CSBTB52
Duration	2 Hours	**Applicable to Batches**	All batches
Assignment Number: 3.3(Present assignment number)/**24**(Total number of assignments)			

Q.No.	Question	*Expected Time to complete*
1	**Lab 3: Application for TGNPDCL – Electricity Bill Generation Using Python & AI Tools** **Lab Objectives** • To design a real-world electricity billing application using Python • To use AI-assisted coding tools for logic generation and optimization • To understand conditional logic and arithmetic operations • To generate structured billing output similar to utility bills **Lab Outcomes (LOs)** After completing this lab, students will be able to:	Week2 - Wednesday

NAME:P Pranay Kumar H.NO: 2303A51829 BATCH:26

- Read and validate user input in Python
- Apply conditional logic for tariff-based billing
- Use AI tools to assist in program development
- Calculate and display electricity bill components
- Build a complete real-time application

Task 1: AI-Generated Logic for Reading Consumer Details

Scenario

An electricity billing system must collect accurate consumer data.

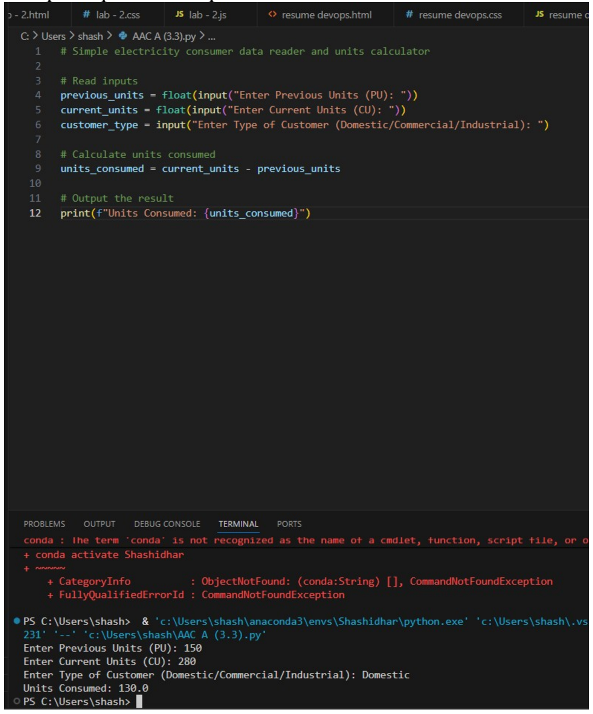
Task Description

Use an AI tool (GitHub Copilot / Gemini) to generate a Python program that:

- Reads:
  - Previous Units (PU)
  - Current Units (CU)
  - Type of Customer
- Calculates units consumed
- Implements logic directly in the main program (no functions)

Expected Output

- Correct input reading
- Units consumed calculation
- Screenshot showing AI-generated code
- Sample input and output



```python
# Simple electricity consumer data reader and units calculator

# Read inputs
previous_units = float(input("Enter Previous Units (PU): "))
current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer (Domestic/Commercial/Industrial): ")

# Calculate units consumed
units_consumed = current_units - previous_units

# Output the result
print(f"Units Consumed: {units_consumed}")
```

```
conda : The term 'conda' is not recognized as the name of a cmdlet, function, script file, or o
+ conda activate Shashidhar
+ ~~~~~
    + CategoryInfo          : ObjectNotFound: (conda:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\shash>  & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe' 'c:\Users\shash\.vs
231' '--' 'c:\Users\shash\AAC A (3.3).py'
Enter Previous Units (PU): 150
Enter Current Units (CU): 280
Enter Type of Customer (Domestic/Commercial/Industrial): Domestic
Units Consumed: 130.0
PS C:\Users\shash>
```

Task 2: Energy Charges Calculation Based on Units Consumed

Scenario

Energy charges depend on the number of units consumed and customer type.

Task Description

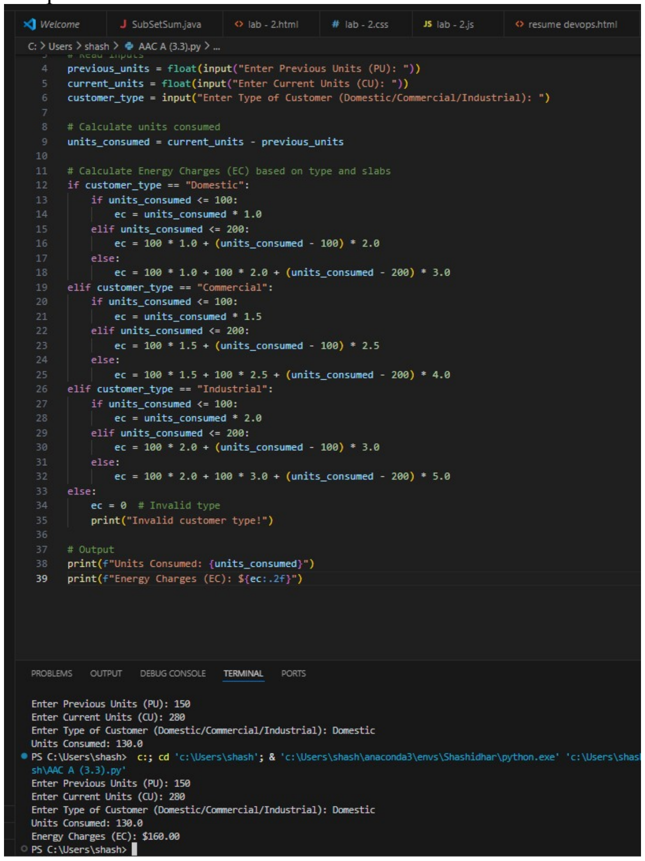
Review the AI-generated code from Task 1 and extend it to:

- Calculate **Energy Charges (EC)**
- Use conditional statements based on:
  - Domestic
  - Commercial
  - Industrial consumers
- Improve readability using AI prompts such as:

- *"Simplify energy charge calculation logic"*
- *"Optimize conditional statements"*

Expected Output

- Correct EC calculation
- Clear conditional logic
- Original and improved versions (optional)
- Sample execution results



```python
previous_units = float(input("Enter Previous Units (PU): "))
current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer (Domestic/Commercial/Industrial): ")

# Calculate units consumed
units_consumed = current_units - previous_units

# Calculate Energy Charges (EC) based on type and slabs
if customer_type == "Domestic":
    if units_consumed <= 100:
        ec = units_consumed * 1.0
    elif units_consumed <= 200:
        ec = 100 * 1.0 + (units_consumed - 100) * 2.0
    else:
        ec = 100 * 1.0 + 100 * 2.0 + (units_consumed - 200) * 3.0
elif customer_type == "Commercial":
    if units_consumed <= 100:
        ec = units_consumed * 1.5
    elif units_consumed <= 200:
        ec = 100 * 1.5 + (units_consumed - 100) * 2.5
    else:
        ec = 100 * 1.5 + 100 * 2.5 + (units_consumed - 200) * 4.0
elif customer_type == "Industrial":
    if units_consumed <= 100:
        ec = units_consumed * 2.0
    elif units_consumed <= 200:
        ec = 100 * 2.0 + (units_consumed - 100) * 3.0
    else:
        ec = 100 * 2.0 + 100 * 3.0 + (units_consumed - 200) * 5.0
else:
    ec = 0  # Invalid type
    print("Invalid customer type!")

# Output
print(f"Units Consumed: {units_consumed}")
print(f"Energy Charges (EC): ${ec:.2f}")
```

```
Enter Previous Units (PU): 150
Enter Current Units (CU): 280
Enter Type of Customer (Domestic/Commercial/Industrial): Domestic
Units Consumed: 130.0
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe' 'c:\Users\shas
sh\AAC A (3.3).py'
Enter Previous Units (PU): 150
Enter Current Units (CU): 280
Enter Type of Customer (Domestic/Commercial/Industrial): Domestic
Units Consumed: 130.0
Energy Charges (EC): $160.00
PS C:\Users\shash>
```

Task 3: Modular Design Using AI Assistance (Using Functions)

Scenario

Billing logic must be reusable for multiple consumers.

Task Description

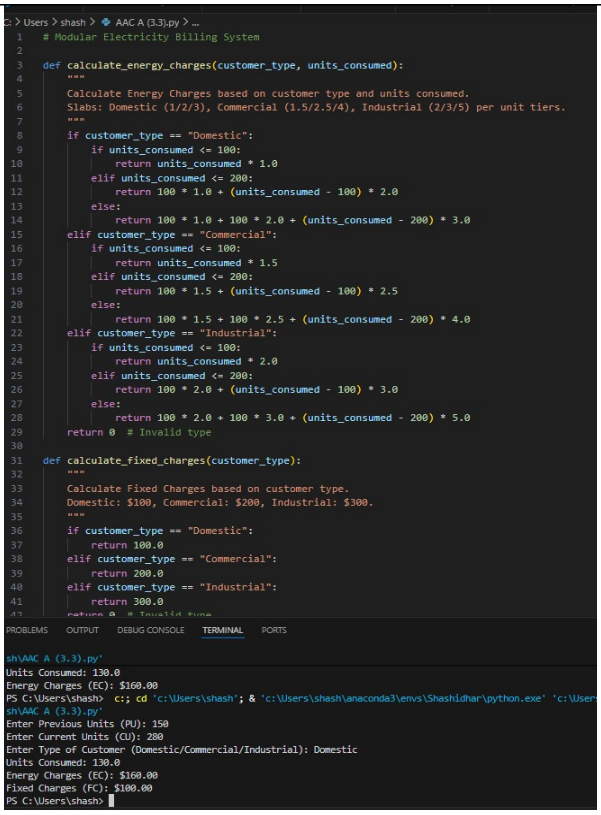
Use AI assistance to generate a Python program that:

- Uses user-defined functions to:
  - Calculate Energy Charges
  - Calculate Fixed Charges
- Returns calculated values
- Includes meaningful comments

Expected Output

- Function-based Python program
- Correct EC and FC values
- Screenshots of AI-assisted function generation
- Test cases with outputs

NAME:P Pranay Kumar H.NO: 2303A51829 BATCH:26



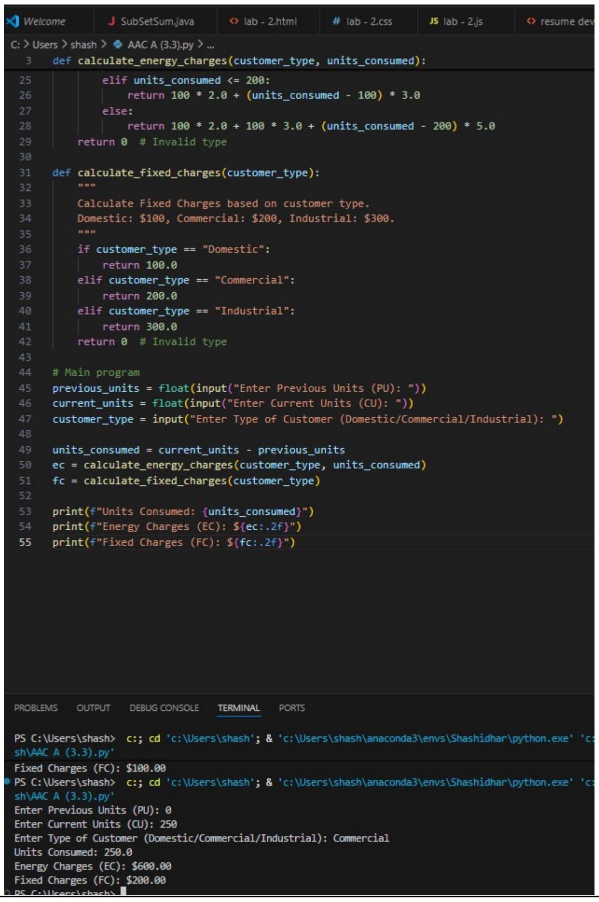
C: > Users > shash > AAC A (3.3).py > ...

```python
# Modular Electricity Billing System

def calculate_energy_charges(customer_type, units_consumed):
    """
    Calculate Energy Charges based on customer type and units consumed.
    Slabs: Domestic (1/2/3), Commercial (1.5/2.5/4), Industrial (2/3/5) per unit tiers.
    """
    if customer_type == "Domestic":
        if units_consumed <= 100:
            return units_consumed * 1.0
        elif units_consumed <= 200:
            return 100 * 1.0 + (units_consumed - 100) * 2.0
        else:
            return 100 * 1.0 + 100 * 2.0 + (units_consumed - 200) * 3.0
    elif customer_type == "Commercial":
        if units_consumed <= 100:
            return units_consumed * 1.5
        elif units_consumed <= 200:
            return 100 * 1.5 + (units_consumed - 100) * 2.5
        else:
            return 100 * 1.5 + 100 * 2.5 + (units_consumed - 200) * 4.0
    elif customer_type == "Industrial":
        if units_consumed <= 100:
            return units_consumed * 2.0
        elif units_consumed <= 200:
            return 100 * 2.0 + (units_consumed - 100) * 3.0
        else:
            return 100 * 2.0 + 100 * 3.0 + (units_consumed - 200) * 5.0
    return 0  # Invalid type

def calculate_fixed_charges(customer_type):
    """
    Calculate Fixed Charges based on customer type.
    Domestic: $100, Commercial: $200, Industrial: $300.
    """
    if customer_type == "Domestic":
        return 100.0
    elif customer_type == "Commercial":
        return 200.0
    elif customer_type == "Industrial":
        return 300.0
    return 0  # Invalid type
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
sh\AAC A (3.3).py'
Units Consumed: 130.0
Energy Charges (EC): $160.00
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe' 'c:\User
sh\AAC A (3.3).py'
Enter Previous Units (PU): 150
Enter Current Units (CU): 280
Enter Type of Customer (Domestic/Commercial/Industrial): Domestic
Units Consumed: 130.0
Energy Charges (EC): $160.00
Fixed Charges (FC): $100.00
PS C:\Users\shash>
```

•



Welcome | SubSetSum.java | lab - 2.html | lab - 2.css | lab - 2.js | resume dev

C: > Users > shash > AAC A (3.3).py > ...

```python
def calculate_energy_charges(customer_type, units_consumed):
        elif units_consumed <= 200:
            return 100 * 2.0 + (units_consumed - 100) * 3.0
        else:
            return 100 * 2.0 + 100 * 3.0 + (units_consumed - 200) * 5.0
    return 0  # Invalid type

def calculate_fixed_charges(customer_type):
    """
    Calculate Fixed Charges based on customer type.
    Domestic: $100, Commercial: $200, Industrial: $300.
    """
    if customer_type == "Domestic":
        return 100.0
    elif customer_type == "Commercial":
        return 200.0
    elif customer_type == "Industrial":
        return 300.0
    return 0  # Invalid type

# Main program
previous_units = float(input("Enter Previous Units (PU): "))
current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer (Domestic/Commercial/Industrial): ")

units_consumed = current_units - previous_units
ec = calculate_energy_charges(customer_type, units_consumed)
fc = calculate_fixed_charges(customer_type)

print(f"Units Consumed: {units_consumed}")
print(f"Energy Charges (EC): ${ec:.2f}")
print(f"Fixed Charges (FC): ${fc:.2f}")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe' 'c:
sh\AAC A (3.3).py'
Fixed Charges (FC): $100.00
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe' 'c:
sh\AAC A (3.3).py'
Enter Previous Units (PU): 0
Enter Current Units (CU): 250
Enter Type of Customer (Domestic/Commercial/Industrial): Commercial
Units Consumed: 250.0
Energy Charges (EC): $600.00
Fixed Charges (FC): $200.00
PS C:\Users\shash>
```

NAME:P Pranay Kumar H.NO: 2303A51829 BATCH:26

Task 4: Calculation of Additional Charges

Scenario

Electricity bills include multiple additional charges.

Task Description

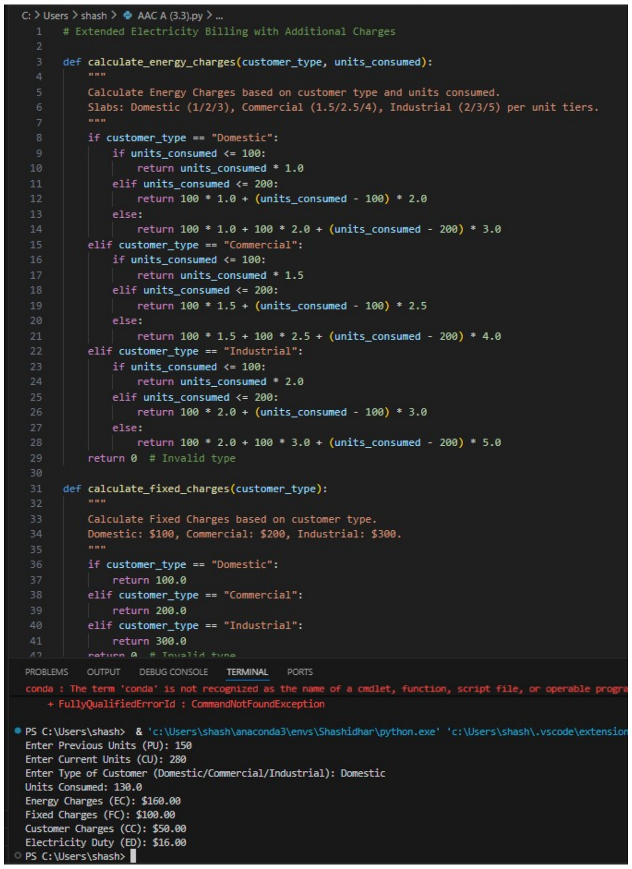
Extend the program to calculate:

- **FC –** Fixed Charges
- **CC –** Customer Charges
- **ED –** Electricity Duty (percentage of EC)

Use AI prompts like:

- *"Add electricity duty calculation"*
- *"Improve billing accuracy"*

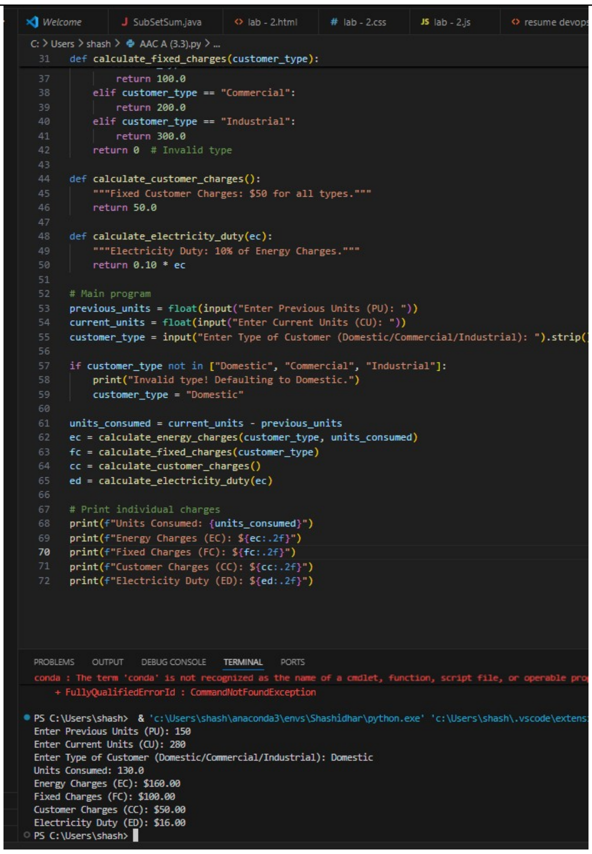
Expected Output

- Individual charge values printed
- Correct duty calculation
- Well-structured output
- Verified intermediate results
- 

```
C: > Users > shash > AAC A (3.3).py > ...
# Extended Electricity Billing with Additional Charges

def calculate_energy_charges(customer_type, units_consumed):
    """
    Calculate Energy Charges based on customer type and units consumed.
    Slabs: Domestic (1/2/3), Commercial (1.5/2.5/4), Industrial (2/3/5) per unit tiers.
    """
    if customer_type == "Domestic":
        if units_consumed <= 100:
            return units_consumed * 1.0
        elif units_consumed <= 200:
            return 100 * 1.0 + (units_consumed - 100) * 2.0
        else:
            return 100 * 1.0 + 100 * 2.0 + (units_consumed - 200) * 3.0
    elif customer_type == "Commercial":
        if units_consumed <= 100:
            return units_consumed * 1.5
        elif units_consumed <= 200:
            return 100 * 1.5 + (units_consumed - 100) * 2.5
        else:
            return 100 * 1.5 + 100 * 2.5 + (units_consumed - 200) * 4.0
    elif customer_type == "Industrial":
        if units_consumed <= 100:
            return units_consumed * 2.0
        elif units_consumed <= 200:
            return 100 * 2.0 + (units_consumed - 100) * 3.0
        else:
            return 100 * 2.0 + 100 * 3.0 + (units_consumed - 200) * 5.0
    return 0  # Invalid type

def calculate_fixed_charges(customer_type):
    """
    Calculate Fixed Charges based on customer type.
    Domestic: $100, Commercial: $200, Industrial: $300.
    """
    if customer_type == "Domestic":
        return 100.0
    elif customer_type == "Commercial":
        return 200.0
    elif customer_type == "Industrial":
        return 300.0
    return 0  # Invalid type
```

```
conda : The term 'conda' is not recognized as the name of a cmdlet, function, script file, or operable progra
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\shash>  & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe' 'c:\Users\shash\.vscode\extension
Enter Previous Units (PU): 150
Enter Current Units (CU): 280
Enter Type of Customer (Domestic/Commercial/Industrial): Domestic
Units Consumed: 130.0
Energy Charges (EC): $160.00
Fixed Charges (FC): $100.00
Customer Charges (CC): $50.00
Electricity Duty (ED): $16.00
PS C:\Users\shash>
```

NAME:P Pranay Kumar H.NO: 2303A51829 BATCH:26



```python
def calculate_fixed_charges(customer_type):
            return 100.0
        elif customer_type == "Commercial":
            return 200.0
        elif customer_type == "Industrial":
            return 300.0
        return 0  # Invalid type

def calculate_customer_charges():
    """Fixed Customer Charges: $50 for all types."""
    return 50.0

def calculate_electricity_duty(ec):
    """Electricity Duty: 10% of Energy Charges."""
    return 0.10 * ec

# Main program
previous_units = float(input("Enter Previous Units (PU): "))
current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer (Domestic/Commercial/Industrial): ").strip(

if customer_type not in ["Domestic", "Commercial", "Industrial"]:
    print("Invalid type! Defaulting to Domestic.")
    customer_type = "Domestic"

units_consumed = current_units - previous_units
ec = calculate_energy_charges(customer_type, units_consumed)
fc = calculate_fixed_charges(customer_type)
cc = calculate_customer_charges()
ed = calculate_electricity_duty(ec)

# Print individual charges
print(f"Units Consumed: {units_consumed}")
print(f"Energy Charges (EC): ${ec:.2f}")
print(f"Fixed Charges (FC): ${fc:.2f}")
print(f"Customer Charges (CC): ${cc:.2f}")
print(f"Electricity Duty (ED): ${ed:.2f}")
```

```
conda : The term 'conda' is not recognized as the name of a cmdlet, function, script file, or operable pro
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\shash>  & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe' 'c:\Users\shash\.vscode\extens
Enter Previous Units (PU): 150
Enter Current Units (CU): 280
Enter Type of Customer (Domestic/Commercial/Industrial): Domestic
Units Consumed: 130.0
Energy Charges (EC): $160.00
Fixed Charges (FC): $100.00
Customer Charges (CC): $50.00
Electricity Duty (ED): $16.00
PS C:\Users\shash>
```

Task 5: Final Bill Generation and Output Analysis
Scenario

The final electricity bill must present all values clearly.

Task Description

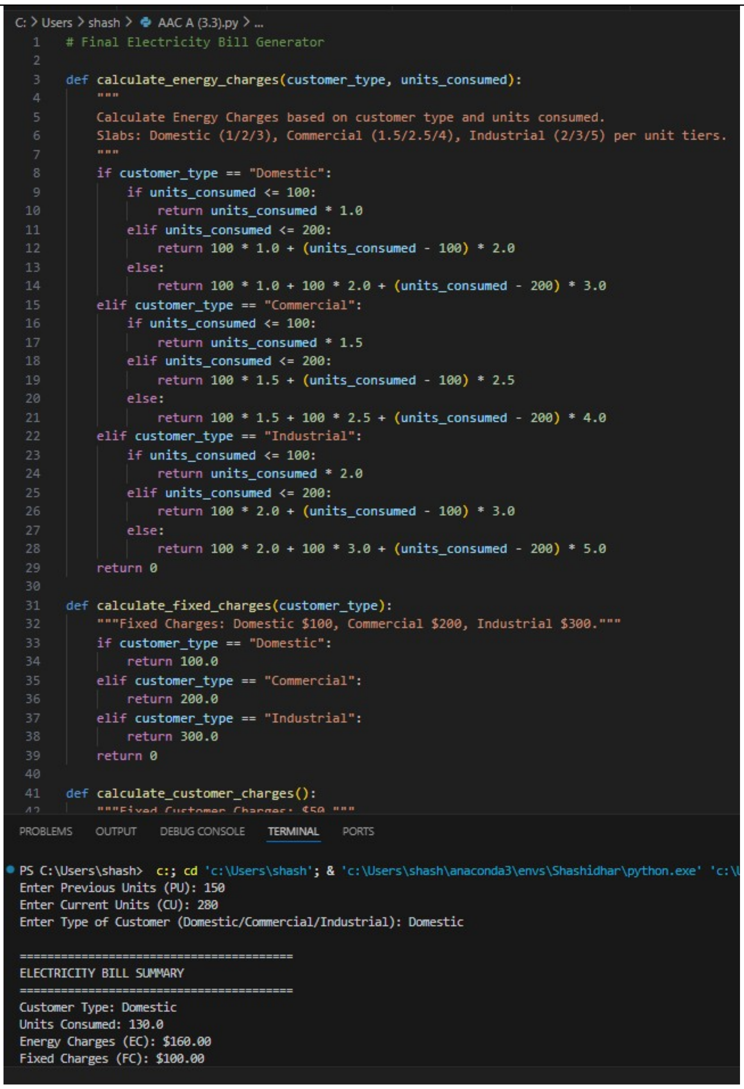
Develop the final Python application to:

- Calculate total bill:
- Total Bill = EC + FC + CC + ED
- Display:
  - Energy Charges (EC)
  - Fixed Charges (FC)
  - Customer Charges (CC)
  - Electricity Duty (ED)
  - Total Bill Amount
- Analyze the program based on:
  - Accuracy
  - Readability
  - Real-world applicability

Expected Output

- Complete electricity bill output
- Neatly formatted display
- Sample input/output
- Short analysis paragraph

NAME:P Pranay Kumar H.NO: 2303A51829 BATCH:26

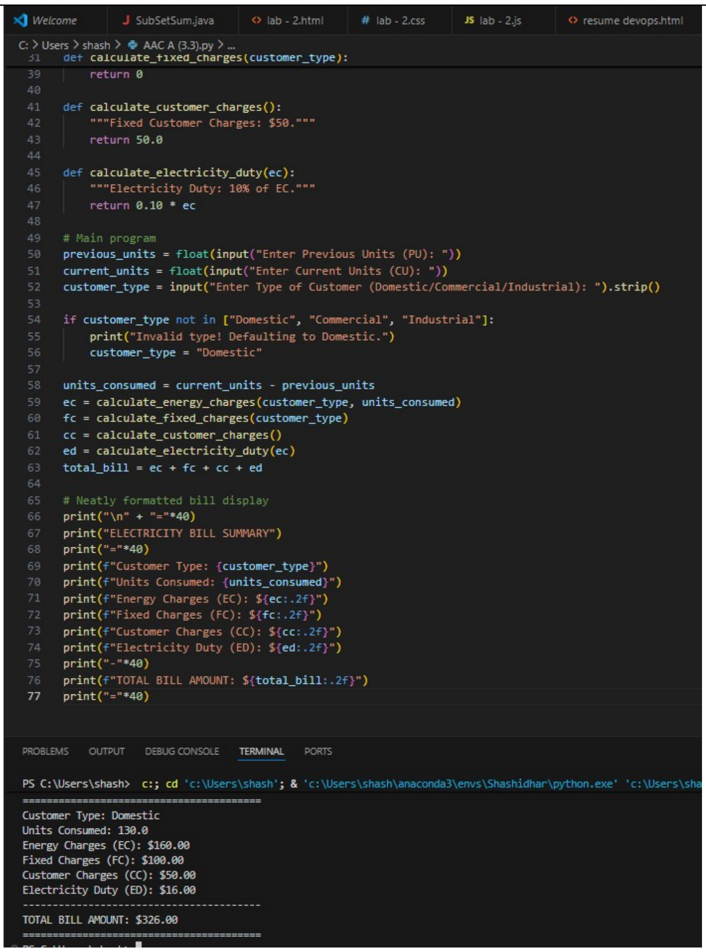
- 

C: > Users > shash > AAC A (3.3).py > ...

```python
# Final Electricity Bill Generator

def calculate_energy_charges(customer_type, units_consumed):
    """
    Calculate Energy Charges based on customer type and units consumed.
    Slabs: Domestic (1/2/3), Commercial (1.5/2.5/4), Industrial (2/3/5) per unit tiers.
    """
    if customer_type == "Domestic":
        if units_consumed <= 100:
            return units_consumed * 1.0
        elif units_consumed <= 200:
            return 100 * 1.0 + (units_consumed - 100) * 2.0
        else:
            return 100 * 1.0 + 100 * 2.0 + (units_consumed - 200) * 3.0
    elif customer_type == "Commercial":
        if units_consumed <= 100:
            return units_consumed * 1.5
        elif units_consumed <= 200:
            return 100 * 1.5 + (units_consumed - 100) * 2.5
        else:
            return 100 * 1.5 + 100 * 2.5 + (units_consumed - 200) * 4.0
    elif customer_type == "Industrial":
        if units_consumed <= 100:
            return units_consumed * 2.0
        elif units_consumed <= 200:
            return 100 * 2.0 + (units_consumed - 100) * 3.0
        else:
            return 100 * 2.0 + 100 * 3.0 + (units_consumed - 200) * 5.0
    return 0

def calculate_fixed_charges(customer_type):
    """Fixed Charges: Domestic $100, Commercial $200, Industrial $300."""
    if customer_type == "Domestic":
        return 100.0
    elif customer_type == "Commercial":
        return 200.0
    elif customer_type == "Industrial":
        return 300.0
    return 0

def calculate_customer_charges():
    """Fixed Customer Charges: $50."""
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe' 'c:\
Enter Previous Units (PU): 150
Enter Current Units (CU): 280
Enter Type of Customer (Domestic/Commercial/Industrial): Domestic

========================================
ELECTRICITY BILL SUMMARY
========================================
Customer Type: Domestic
Units Consumed: 130.0
Energy Charges (EC): $160.00
Fixed Charges (FC): $100.00
```



```python
def calculate_fixed_charges(customer_type):
    return 0

def calculate_customer_charges():
    """Fixed Customer Charges: $50."""
    return 50.0

def calculate_electricity_duty(ec):
    """Electricity Duty: 10% of EC."""
    return 0.10 * ec

# Main program
previous_units = float(input("Enter Previous Units (PU): "))
current_units = float(input("Enter Current Units (CU): "))
customer_type = input("Enter Type of Customer (Domestic/Commercial/Industrial): ").strip()

if customer_type not in ["Domestic", "Commercial", "Industrial"]:
    print("Invalid type! Defaulting to Domestic.")
    customer_type = "Domestic"

units_consumed = current_units - previous_units
ec = calculate_energy_charges(customer_type, units_consumed)
fc = calculate_fixed_charges(customer_type)
cc = calculate_customer_charges()
ed = calculate_electricity_duty(ec)
total_bill = ec + fc + cc + ed

# Neatly formatted bill display
print("\n" + "="*40)
print("ELECTRICITY BILL SUMMARY")
print("="*40)
print(f"Customer Type: {customer_type}")
print(f"Units Consumed: {units_consumed}")
print(f"Energy Charges (EC): ${ec:.2f}")
print(f"Fixed Charges (FC): ${fc:.2f}")
print(f"Customer Charges (CC): ${cc:.2f}")
print(f"Electricity Duty (ED): ${ed:.2f}")
print("-"*40)
print(f"TOTAL BILL AMOUNT: ${total_bill:.2f}")
print("="*40)
```

```
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe' 'c:\Users\sha
========================================
Customer Type: Domestic
Units Consumed: 130.0
Energy Charges (EC): $160.00
Fixed Charges (FC): $100.00
Customer Charges (CC): $50.00
Electricity Duty (ED): $16.00
----------------------------------------
TOTAL BILL AMOUNT: $326.00
========================================
```

Note: Report should be submitted as a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots.

NAME:P Pranay Kumar H.NO:2303A51829 BATCH:26

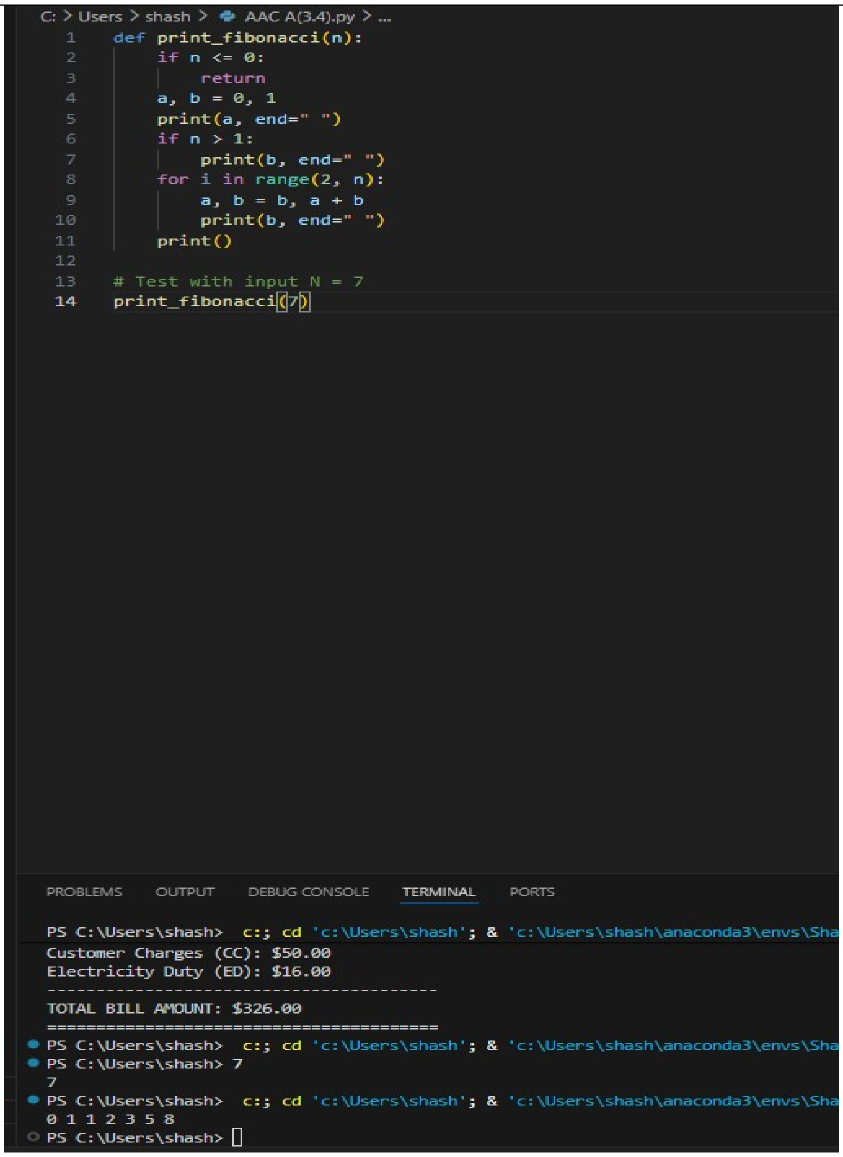
SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE	**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING**

Program Name: B. Tech	**Assignment Type: Lab**	**Academic Year:**2025-2026

Course Coordinator Name	Dr. Rishabh Mittal		
Instructor(s) Name	Mr. S Naresh Kumar Ms. B. Swathi Dr. Sasanko Shekhar Gantayat Mr. Md Sallauddin Dr. Mathivanan Mr. Y Srikanth Ms. N Shilpa Dr. Rishabh Mittal (Coordinator) Dr. R. Prashant Kumar Mr. Ankushavali MD Mr. B Viswanath Ms. Sujitha Reddy Ms. A. Anitha Ms. M.Madhuri Ms. Katherashala Swetha Ms. Velpula sumalatha Mr. Bingi Raju		
CourseCode	23CS002PC304	**Course Title**	AI Assisted Coding
Year/Sem	III/II	**Regulation**	R23
Date and Day of Assignment	**Week2**	**Time(s)**	23CSBTB01 To 23CSBTB52
Duration	2 Hours	**Applicable to Batches**	All batches
Assignment Number: 3.4 (Present assignment number)/**24**(Total number of assignments)			

Q.No.	**Question**	***Expected Time to complete***
1	Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques	Week2

	Task 1: Zero-shot Prompt – Fibonacci Series Generator **Task Description #1** • Without giving an example, write a single comment prompt asking GitHub Copilot to generate a Python function to print the first N Fibonacci numbers. **Expected Output #1** • A complete Python function generated by Copilot without any example provided. • Correct output for sample input N = 7 → 0 1 1 2 3 5 8 • Observation on how Copilot understood the instruction with zero context.	



```
C: > Users > shash > AAC A(3.4).py > ...
def print_fibonacci(n):
    if n <= 0:
        return
    a, b = 0, 1
    print(a, end=" ")
    if n > 1:
        print(b, end=" ")
    for i in range(2, n):
        a, b = b, a + b
        print(b, end=" ")
    print()

# Test with input N = 7
print_fibonacci(7)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Sha
Customer Charges (CC): $50.00
Electricity Duty (ED): $16.00
----------------------------------------
TOTAL BILL AMOUNT: $326.00
========================================
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Sha
PS C:\Users\shash> 7
7
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Sha
0 1 1 2 3 5 8
PS C:\Users\shash> 
```

Task 2: One-shot Prompt – List Reversal Function

Task Description #2

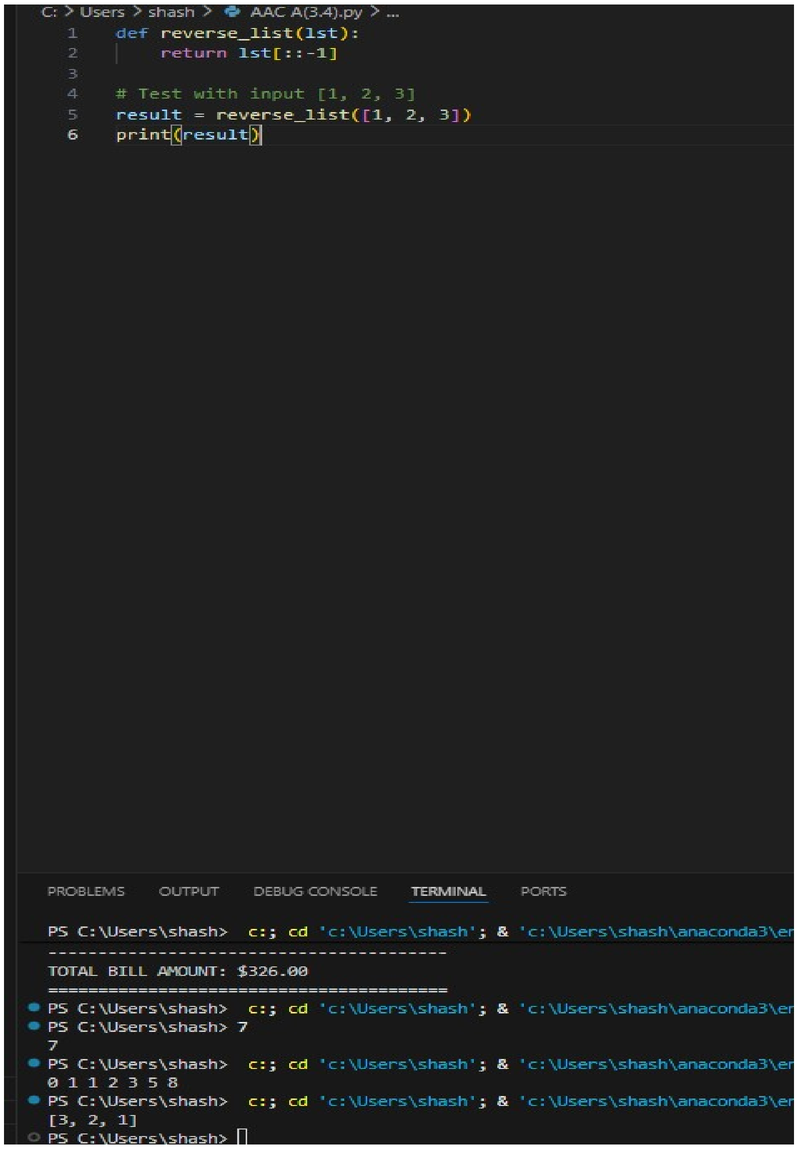
• Write a comment prompt to reverse a list and provide one example below the comment to guide Copilot.

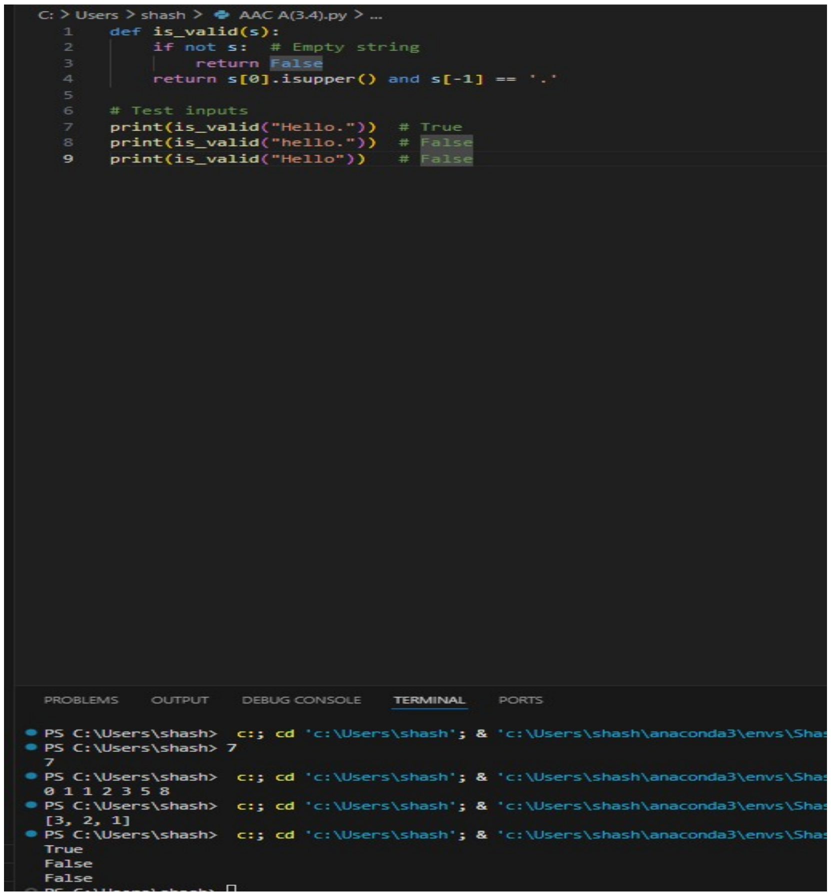
Expected Output #2

• Copilot-generated function to reverse a list using slicing or loop.

• Output: [3, 2, 1] for input [1, 2, 3]

• Observation on how adding a single example improved Copilot's

	accuracy.
	
	C: > Users > shash > AAC A(3.4).py > ...
	`def reverse_list(lst):`
	`    return lst[::-1]`
	
	`# Test with input [1, 2, 3]`
	`result = reverse_list([1, 2, 3])`
	`print(result)`
	
	PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
	`PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\er`
	`----------------------------------------`
	`TOTAL BILL AMOUNT: $326.00`
	`========================================`
	`PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\er`
	`PS C:\Users\shash> 7`
	`7`
	`PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\er`
	`0 1 1 2 3 5 8`
	`PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\er`
	`[3, 2, 1]`
	`PS C:\Users\shash>`
	Task 3: Few-shot Prompt – String Pattern Matching
	Task Description #3
	• Write a comment with 2–3 examples to help Copilot understand how to check if a string starts with a capital letter and ends with a period.
	Expected Output #3
	• A function is_valid() that checks the pattern.
	• Output: True or False based on input.	

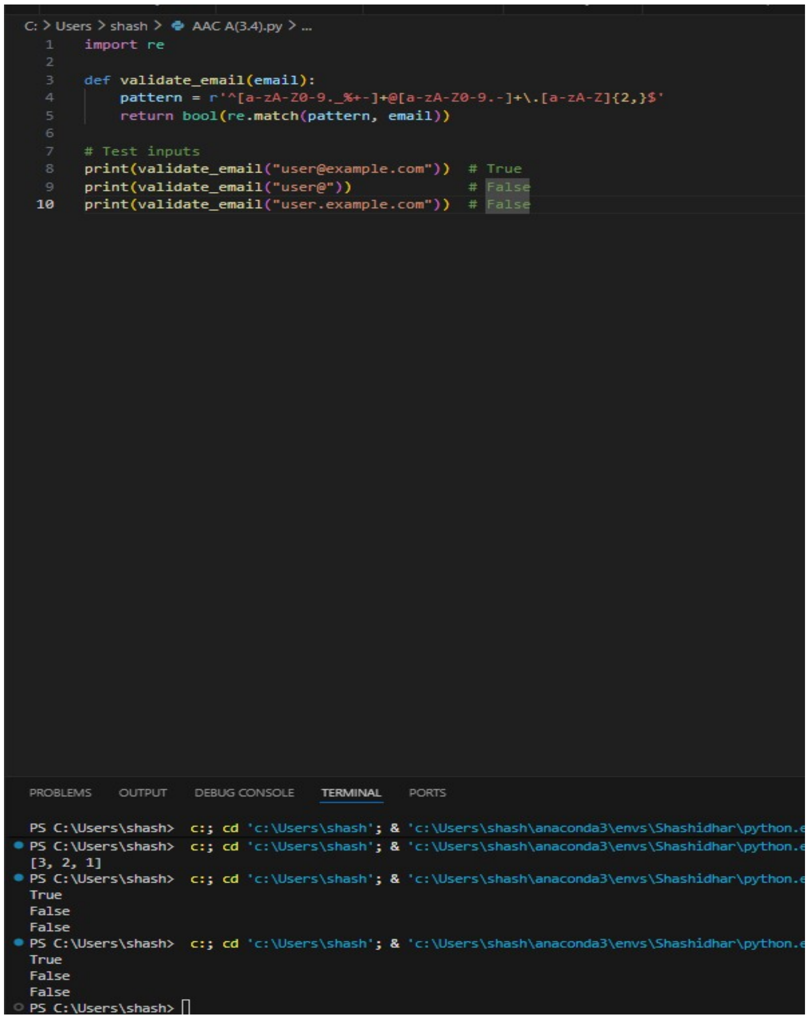
	• Students reflect on how multiple examples guide Copilot to generate more accurate code.  Task 4: Zero-shot vs Few-shot – Email Validator Task Description #4 • First, prompt Copilot to write an email validation function using zero-shot (just the task in comment). • Then, rewrite the prompt using few-shot examples. **Expected Output #4** • Compare both outputs: Zero-shot may result in basic or generic validation. Few-shot gives detailed and specific logic (e.g., @ and domain checking). • Submit both code versions and note how few-shot improves	

```python
def is_valid(s):
    if not s:  # Empty string
        return False
    return s[0].isupper() and s[-1] == '.'

# Test inputs
print(is_valid("Hello."))  # True
print(is_valid("hello."))  # False
print(is_valid("Hello"))   # False
```

```
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shas
PS C:\Users\shash> 7
7
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shas
0 1 1 2 3 5 8
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shas
[3, 2, 1]
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shas
True
False
False
```

Task 4: Zero-shot vs Few-shot – Email Validator

reliability.



```python
import re

def validate_email(email):
    pattern = r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$'
    return bool(re.match(pattern, email))

# Test inputs
print(validate_email("user@example.com"))  # True
print(validate_email("user@"))             # False
print(validate_email("user.example.com"))  # False
```

```
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.e
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.e
[3, 2, 1]
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.e
True
False
False
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.e
True
False
False
PS C:\Users\shash> 
```

Task 5: Prompt Tuning – Summing Digits of a Number

Task Description #5

• Experiment with 2 different prompt styles to generate a function that returns the sum of digits of a number.

Style 1: Generic task prompt

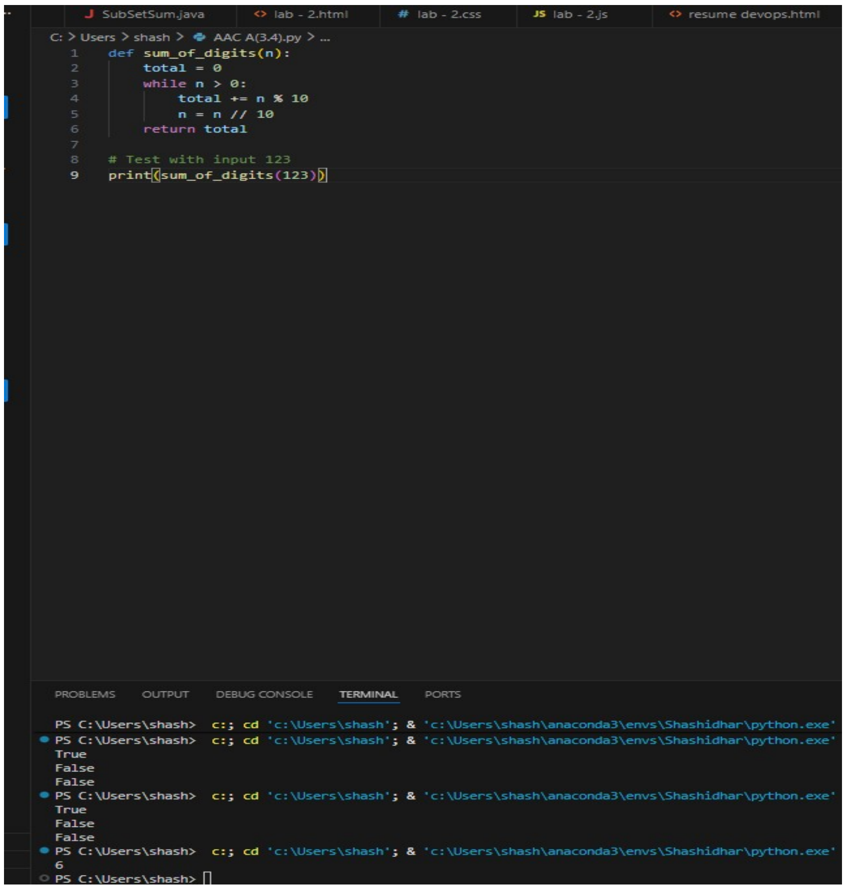
Style 2: Task + Input/Output example

Expected Output #5

• Two versions of the sum_of_digits() function.

• Example Output: sum_of_digits(123) → 6

• Short analysis: which prompt produced cleaner or more optimized code and why?



```python
def sum_of_digits(n):
    total = 0
    while n > 0:
        total += n % 10
        n = n // 10
    return total

# Test with input 123
print(sum_of_digits(123))
```

```
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe'
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe'
True
False
False
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe'
True
False
False
PS C:\Users\shash>  c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe'
6
PS C:\Users\shash> 
```

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots