AI ASSISTED CODING

Assignment-3.1

M.KEERTHANA

2403A51259

BATCH-11

CSE-GEN

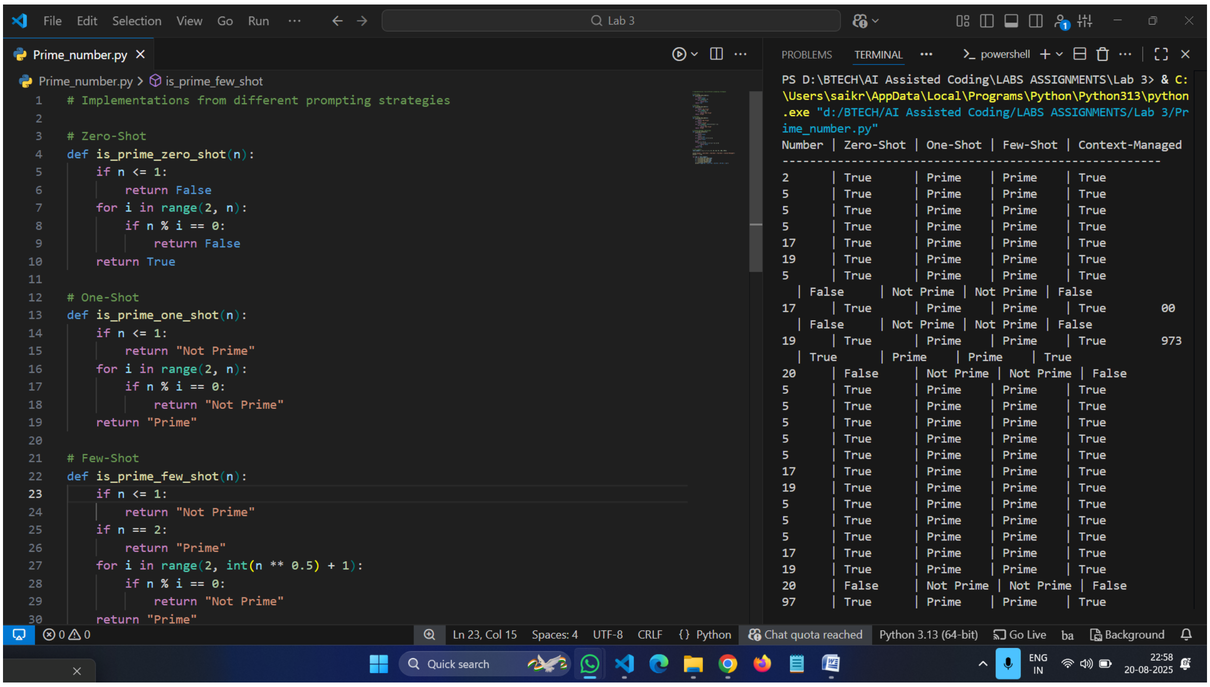
1. Select a simple task: *"Write a Python function to check if a number is prime."*
2. Use different prompting strategies to generate the solution:
    a) Zero-Shot – no examples.
    b) One-Shot – one example provided.
    c) Few-Shot – multiple examples provided.
    d) Context-Managed – detailed prompt with constraints and instructions.
3. Record AI responses and refine prompts to improve code quality.
4. Request AI to optimize the logic for efficiency.
5. Compare results and document improvements.

1. Sample Prompts

- Zero-Shot:

  Write a Python function to check if a number is prime.
- One-Shot:

  Example: Input: 5 → Output: Prime. Now, write a function to check if a number is prime.



```python
# Implementations from different prompting strategies

# Zero-Shot
def is_prime_zero_shot(n):
    if n <= 1:
        return False
    for i in range(2, n):
        if n % i == 0:
            return False
    return True

# One-Shot
def is_prime_one_shot(n):
    if n <= 1:
        return "Not Prime"
    for i in range(2, n):
        if n % i == 0:
            return "Not Prime"
    return "Prime"

# Few-Shot
def is_prime_few_shot(n):
    if n <= 1:
        return "Not Prime"
    if n == 2:
        return "Prime"
    for i in range(2, int(n ** 0.5) + 1):
        if n % i == 0:
            return "Not Prime"
    return "Prime"
```

```
PS D:\BTECH\AI Assisted Coding\LABS ASSIGNMENTS\Lab 3> & C:
\Users\saikr\AppData\Local\Programs\Python\Python313\python
.exe "d:/BTECH/AI Assisted Coding/LABS ASSIGNMENTS/Lab 3/Pr
ime_number.py"
Number | Zero-Shot | One-Shot | Few-Shot | Context-Managed
-------------------------------------------------------
2      | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
17     | True      | Prime    | Prime    | True
19     | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
  | False     | Not Prime | Not Prime | False
17     | True      | Prime    | Prime    | True       00
  | False     | Not Prime | Not Prime | False
19     | True      | Prime    | Prime    | True       973
  | True      | Prime    | Prime    | True
20     | False     | Not Prime | Not Prime | False
5      | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
17     | True      | Prime    | Prime    | True
19     | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
5      | True      | Prime    | Prime    | True
17     | True      | Prime    | Prime    | True
19     | True      | Prime    | Prime    | True
20     | False     | Not Prime | Not Prime | False
97     | True      | Prime    | Prime    | True
```

Task: Mobile Data Usage Billing Application (1.0 Marks)

Objective:

Use Python programming and AI-assisted coding tools to create an application that simulates mobile data billing for a telecom service provider.

Instructions

1. Use GitHub Copilot or Google Gemini to assist in writing the program.
2. Read the following inputs from the user:
   - Data Consumed (in GB)
   - Plan Type (Prepaid / Postpaid)
   - Additional Services Used (e.g., caller tune, OTT subscription, etc.)
3. Implement billing logic to calculate:
   - DC (Data Charges) – charges based on data consumption
   - VC (Value-added Charges) – charges for additional services
   - Tax – applicable tax on the total bill
4. Display an itemized bill showing:
   - Plan Type
   - Data Usage and Charges
   - Value-added Services and Charges

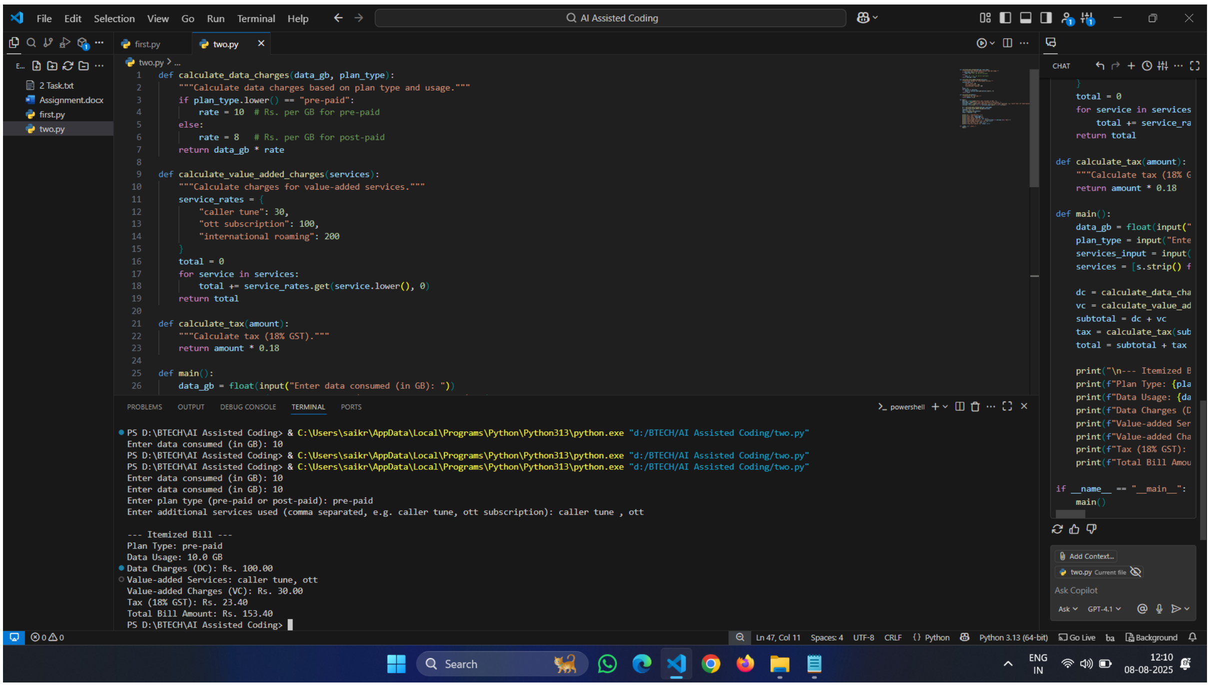
- ○ Tax
- ○ Total Bill Amount

Requirements

- Students must refer to their actual mobile bill for charge structure (data cost, service fees, taxes) to make the program realistic.
- AI assistance (Copilot/Gemini) must be used to generate and refine the initial code.

Deliverables

- AI prompts used for code generation.
- AI-generated Python code and any optimized version.



```python
def calculate_data_charges(data_gb, plan_type):
    """Calculate data charges based on plan type and usage."""
    if plan_type.lower() == "pre-paid":
        rate = 10  # Rs. per GB for pre-paid
    else:
        rate = 8   # Rs. per GB for post-paid
    return data_gb * rate

def calculate_value_added_charges(services):
    """Calculate charges for value-added services."""
    service_rates = {
        "caller tune": 30,
        "ott subscription": 100,
        "international roaming": 200
    }
    total = 0
    for service in services:
        total += service_rates.get(service.lower(), 0)
    return total

def calculate_tax(amount):
    """Calculate tax (18% GST)."""
    return amount * 0.18

def main():
    data_gb = float(input("Enter data consumed (in GB): "))
```

```
PS D:\BTECH\AI Assisted Coding> & C:\Users\saikr\AppData\Local\Programs\Python\Python313\python.exe "d:/BTECH/AI Assisted Coding/two.py"
Enter data consumed (in GB): 10
PS D:\BTECH\AI Assisted Coding> & C:\Users\saikr\AppData\Local\Programs\Python\Python313\python.exe "d:/BTECH/AI Assisted Coding/two.py"
PS D:\BTECH\AI Assisted Coding> & C:\Users\saikr\AppData\Local\Programs\Python\Python313\python.exe "d:/BTECH/AI Assisted Coding/two.py"
Enter data consumed (in GB): 10
Enter data consumed (in GB): 10
Enter plan type (pre-paid or post-paid): pre-paid
Enter additional services used (comma separated, e.g. caller tune, ott subscription): caller tune , ott

--- Itemized Bill ---
Plan Type: pre-paid
Data Usage: 10.0 GB
Data Charges (DC): Rs. 100.00
Value-added Services: caller tune, ott
Value-added Charges (VC): Rs. 30.00
Tax (18% GST): Rs. 23.40
Total Bill Amount: Rs. 153.40
PS D:\BTECH\AI Assisted Coding>
```

```python
    }
    total = 0
    for service in services
        total += service_ra
    return total

def calculate_tax(amount):
    """Calculate tax (18% G
    return amount * 0.18

def main():
    data_gb = float(input("
    plan_type = input("Ente
    services_input = input(
    services = [s.strip() f

    dc = calculate_data_cha
    vc = calculate_value_ad
    subtotal = dc + vc
    tax = calculate_tax(sub
    total = subtotal + tax

    print("\n--- Itemized B
    print(f"Plan Type: {pla
    print(f"Data Usage: {da
    print(f"Data Charges (D
    print(f"Value-added Ser
    print(f"Value-added Cha
    print(f"Tax (18% GST):
    print(f"Total Bill Amou

if __name__ == "__main__":
    main()
```

Task: Develop an LPG Billing System (1.0 Marks)

Objective

Apply your Python programming skills and utilize AI-assisted coding tools to build an application that calculates the LPG bill based on specified customer inputs and billing parameters.

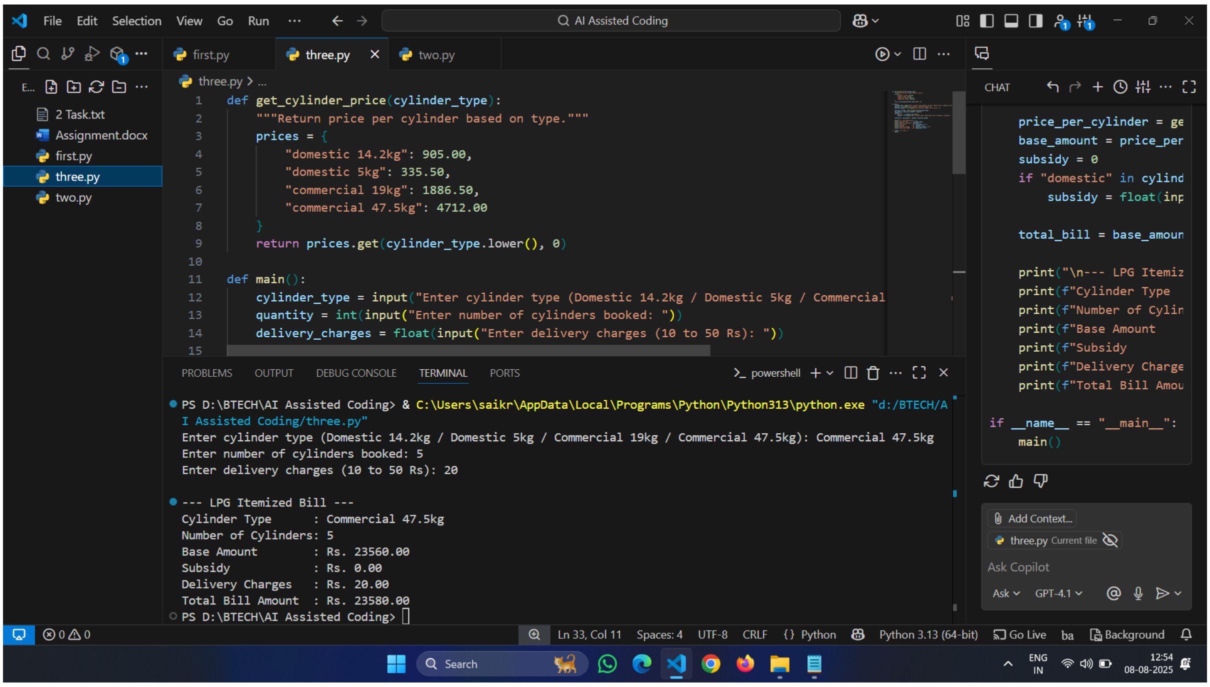
Instructions

1. Use GitHub Copilot or Google Gemini to assist in writing and refining the program.
2. Read the following user inputs:
    - Cylinder Type (Domestic 14.2 kg / Domestic 5 kg / Commercial 19 kg / Commercial 47.5 kg)
    - Number of Cylinders Booked
    - Subsidy Amount (applicable only for domestic cylinders)
3. Refer to the given LPG Price List to determine the price per cylinder:
    - Domestic LPG (14.2 kg) → ₹905.00
    - Domestic LPG (5 kg) → ₹335.50
    - Commercial LPG (19 kg) → ₹1,886.50
    - Commercial LPG (47.5 kg) → ₹4,712.00
    - Delivery Charges (₹10 to ₹50)
4. Implement the billing formula:

Bill Amount = (Price per Cylinder × Quantity) - Subsidy (if applicable) + Delivery Charges

5. Calculate and display an itemized bill including:

- Cylinder Type
- Number of Cylinders
- Base Amount
- Subsidy
- Delivery Charges
- Total Bill Amount



```python
def get_cylinder_price(cylinder_type):
    """Return price per cylinder based on type."""
    prices = {
        "domestic 14.2kg": 905.00,
        "domestic 5kg": 335.50,
        "commercial 19kg": 1886.50,
        "commercial 47.5kg": 4712.00
    }
    return prices.get(cylinder_type.lower(), 0)

def main():
    cylinder_type = input("Enter cylinder type (Domestic 14.2kg / Domestic 5kg / Commercial
    quantity = int(input("Enter number of cylinders booked: "))
    delivery_charges = float(input("Enter delivery charges (10 to 50 Rs): "))
```

```
PS D:\BTECH\AI Assisted Coding> & C:\Users\saikr\AppData\Local\Programs\Python\Python313\python.exe "d:/BTECH/A
I Assisted Coding/three.py"
Enter cylinder type (Domestic 14.2kg / Domestic 5kg / Commercial 19kg / Commercial 47.5kg): Commercial 47.5kg
Enter number of cylinders booked: 5
Enter delivery charges (10 to 50 Rs): 20

--- LPG Itemized Bill ---
Cylinder Type      : Commercial 47.5kg
Number of Cylinders: 5
Base Amount        : Rs. 23560.00
Subsidy            : Rs. 0.00
Delivery Charges   : Rs. 20.00
Total Bill Amount  : Rs. 23580.00
PS D:\BTECH\AI Assisted Coding>
```

```python
    price_per_cylinder = ge
    base_amount = price_per
    subsidy = 0
    if "domestic" in cylind
        subsidy = float(inp

    total_bill = base_amoun

    print("\n--- LPG Itemiz
    print(f"Cylinder Type
    print(f"Number of Cylin
    print(f"Base Amount
    print(f"Subsidy
    print(f"Delivery Charge
    print(f"Total Bill Amou

if __name__ == "__main__":
    main()
```