ASSIGNMENT-9.2

Name: P. Samanvith

Roll Number: 2303A52090

Batch: 40

Task Description -1 (Documentation – Function Summary Generation)

Task:

Use AI to generate concise functional summaries for each Python function in a given script.

Instructions:

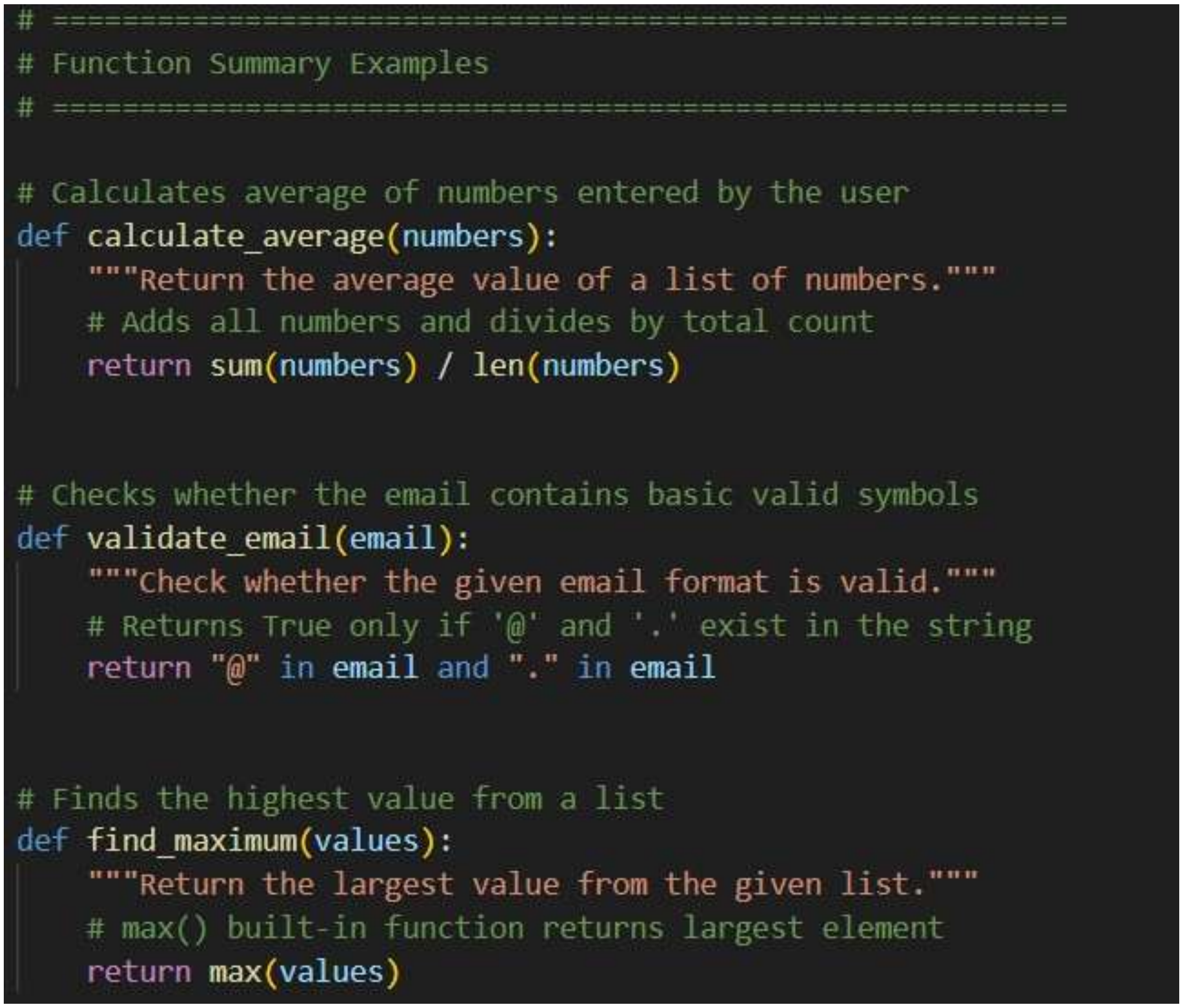
• Provide a Python script to the AI.

• Ask the AI to write a short summary describing the purpose of each function.

• Ensure summaries are brief and technically accurate.

• Do not include code implementation details.

Code:



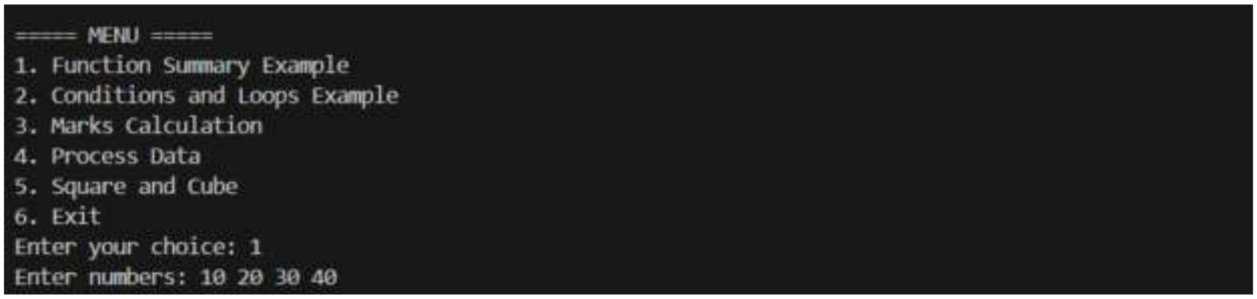
```python
# ============================================================
# Function Summary Examples
# ============================================================

# Calculates average of numbers entered by the user
def calculate_average(numbers):
    """Return the average value of a list of numbers."""
    # Adds all numbers and divides by total count
    return sum(numbers) / len(numbers)


# Checks whether the email contains basic valid symbols
def validate_email(email):
    """Check whether the given email format is valid."""
    # Returns True only if '@' and '.' exist in the string
    return "@" in email and "." in email


# Finds the highest value from a list
def find_maximum(values):
    """Return the largest value from the given list."""
    # max() built-in function returns largest element
    return max(values)
```

Expected Output -1:



```
===== MENU =====
1. Function Summary Example
2. Conditions and Loops Example
3. Marks Calculation
4. Process Data
5. Square and Cube
6. Exit
Enter your choice: 1
Enter numbers: 10 20 30 40
```

Task Description - **–**

Task:

2 (Documentation Logical Explanation for Conditions and Loops)

Use AI to document the logic behind conditional statements and loops in a Python program.

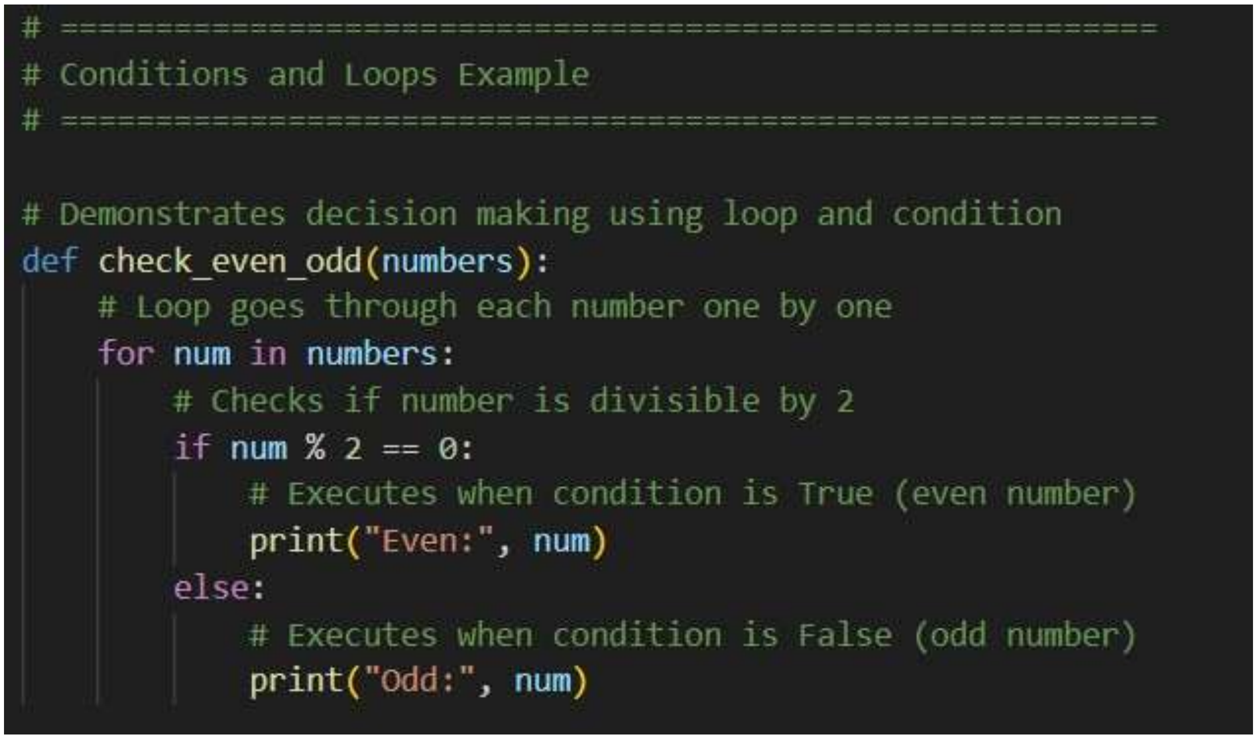
Instructions:

• Provide a Python program without comments.

• Instruct AI to explain only decision-making logic and loop behavior.

• Skip basic syntax explanations.

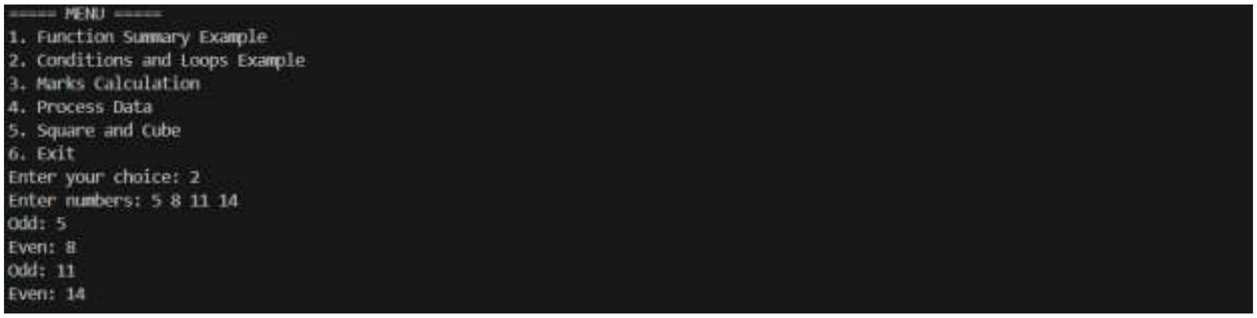
Code:



```python
# ==========================================================
# Conditions and Loops Example
# ==========================================================

# Demonstrates decision making using loop and condition
def check_even_odd(numbers):
    # Loop goes through each number one by one
    for num in numbers:
        # Checks if number is divisible by 2
        if num % 2 == 0:
            # Executes when condition is True (even number)
            print("Even:", num)
        else:
            # Executes when condition is False (odd number)
            print("Odd:", num)
```

Expected Output -2:



```
===== MENU =====
1. Function Summary Example
2. Conditions and Loops Example
3. Marks Calculation
4. Process Data
5. Square and Cube
6. Exit
Enter your choice: 2
Enter numbers: 5 8 11 14
Odd: 5
Even: 8
Odd: 11
Even: 14
```

Task Description - **–**

Task:

3 (Documentation File-Level Overview)

Use AI to generate a high-level overview describing the functionality of an entire Python file.

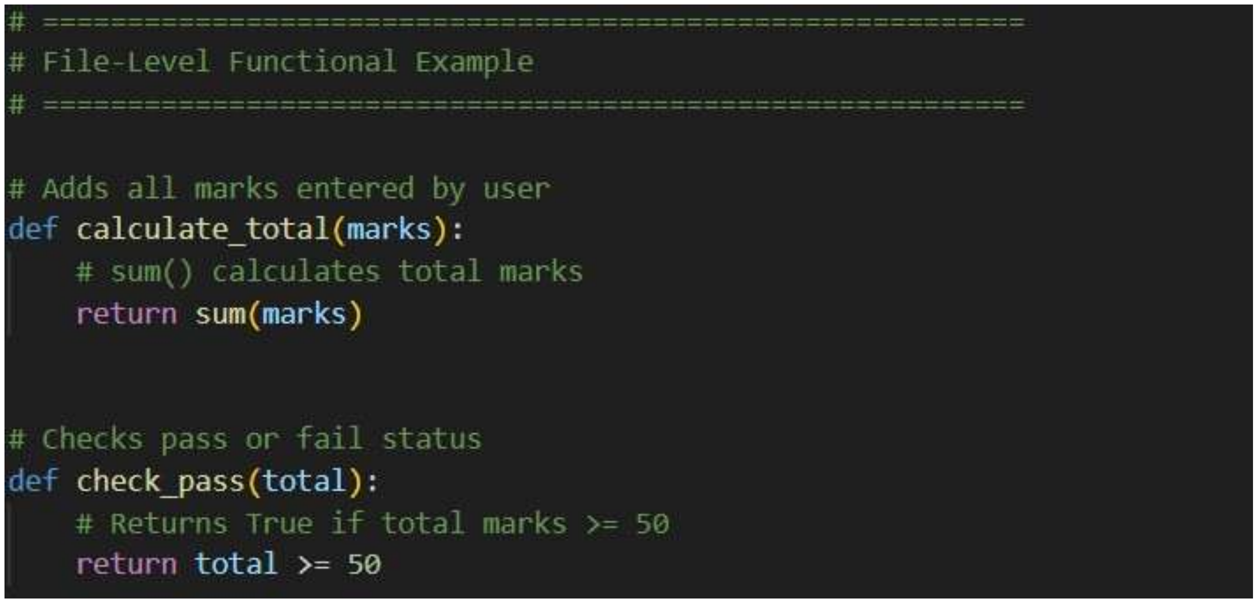
Instructions:

• Provide the complete Python file to AI.

• Ask AI to write a brief overview summarizing the file's purpose and functionality.

• Place the overview at the top of the file.

Code:

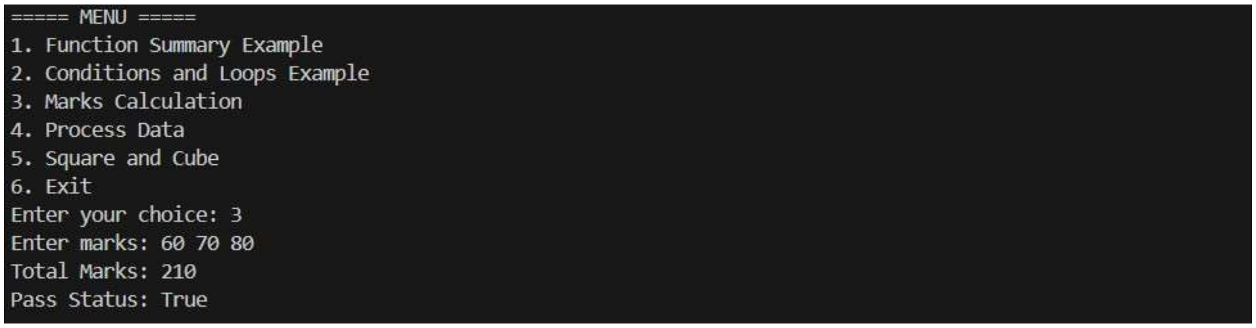


```python
# ==========================================================
# File-Level Functional Example
# ==========================================================

# Adds all marks entered by user
def calculate_total(marks):
    # sum() calculates total marks
    return sum(marks)


# Checks pass or fail status
def check_pass(total):
    # Returns True if total marks >= 50
    return total >= 50
```

Expected Output -3:



```
===== MENU =====
1. Function Summary Example
2. Conditions and Loops Example
3. Marks Calculation
4. Process Data
5. Square and Cube
6. Exit
Enter your choice: 3
Enter marks: 60 70 80
Total Marks: 210
Pass Status: True
```

Task Description -   –

Task:

4 (Documentation   Refine Existing Documentation)

Use AI to improve clarity and consistency of existing documentation in Python code.

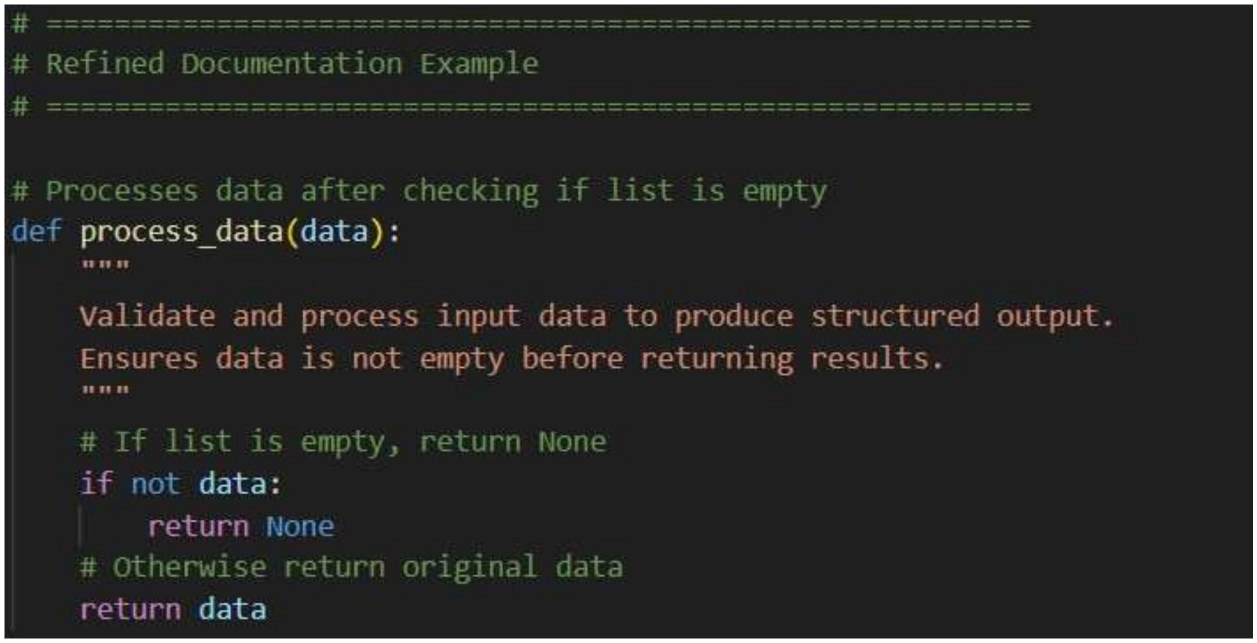
Instructions:

• Provide Python code containing basic or unclear comments.

• Ask AI to rewrite the documentation to improve clarity and consistency.

• Ensure technical meaning remains unchanged.

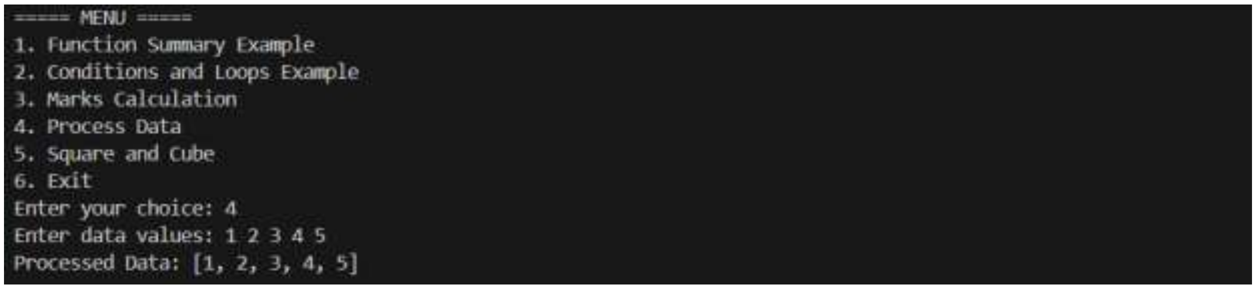
Code:



```python
# ============================================================
# Refined Documentation Example
# ============================================================

# Processes data after checking if list is empty
def process_data(data):
    """
    Validate and process input data to produce structured output.
    Ensures data is not empty before returning results.
    """
    # If list is empty, return None
    if not data:
        return None
    # Otherwise return original data
    return data
```

Expected Output -4:



```
===== MENU =====
1. Function Summary Example
2. Conditions and Loops Example
3. Marks Calculation
4. Process Data
5. Square and Cube
6. Exit
Enter your choice: 4
Enter data values: 1 2 3 4 5
Processed Data: [1, 2, 3, 4, 5]
```

Task Description -5 (Documentation – Prompt Detail Impact Study)

Task:

Study the impact of prompt detail on AI-generated documentation quality.

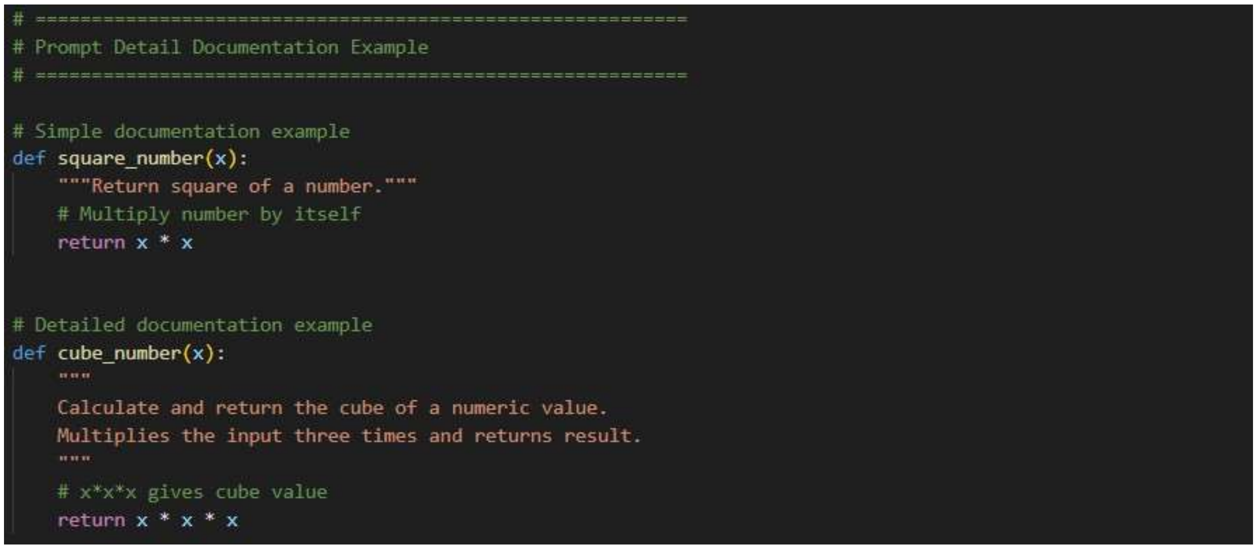
Instructions:

Create two prompts: one brief and one detailed.

• Use both prompts to document the same Python function.

• Compare the generated outputs.

Code:

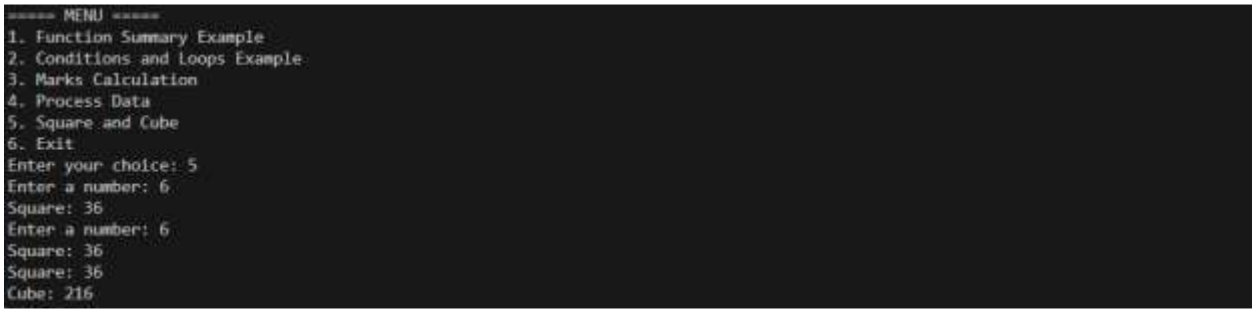


```python
# ==========================================================
# Prompt Detail Documentation Example
# ==========================================================

# Simple documentation example
def square_number(x):
    """Return square of a number."""
    # Multiply number by itself
    return x * x


# Detailed documentation example
def cube_number(x):
    """
    Calculate and return the cube of a numeric value.
    Multiplies the input three times and returns result.
    """
    # x*x*x gives cube value
    return x * x * x
```

Expected Output -5:



```
===== MENU =====
1. Function Summary Example
2. Conditions and Loops Example
3. Marks Calculation
4. Process Data
5. Square and Cube
6. Exit
Enter your choice: 5
Enter a number: 6
Square: 36
Enter a number: 6
Square: 36
Square: 36
Cube: 216
```