Lab Assignment – 9.3

Course: AI - Assisted Coding

Name :P.Srikar

Htno :2403A52002

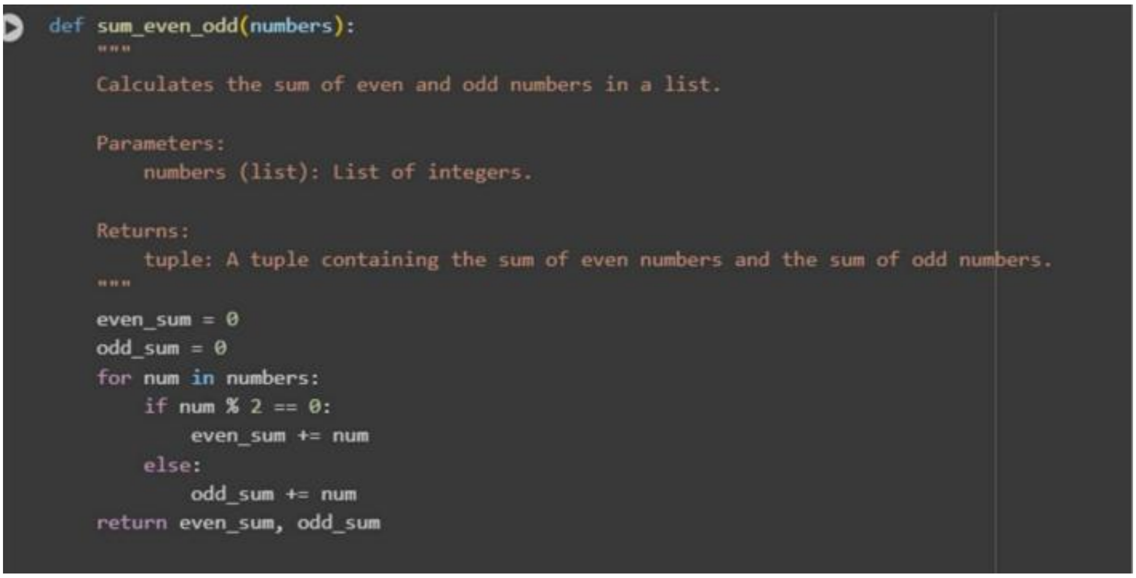
Task-1:

Basic Docstring Generation

- Write python function to return sum of even and odd numbers in the given list.
- Incorporate manual docstring in code with Google Style
- Use an AI-assisted tool (e.g., Copilot, Cursor AI) to generate a docstring describing the function.
- Compare the AI-generated docstring with your manually written one.

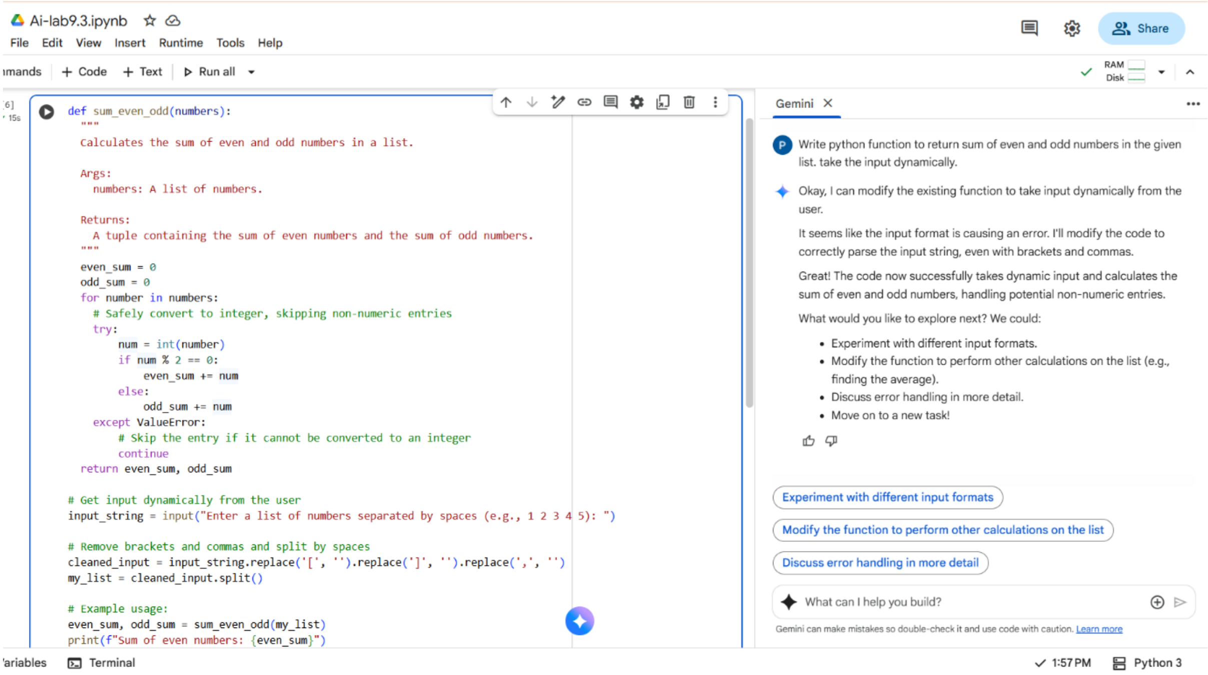
Prompt:Write python function to return sum of even and odd numbers in the given list. take the input dynamically.

Manually written code:



```
def sum_even_odd(numbers):
    """
    Calculates the sum of even and odd numbers in a list.

    Parameters:
        numbers (list): List of integers.

    Returns:
        tuple: A tuple containing the sum of even numbers and the sum of odd numbers.
    """
    even_sum = 0
    odd_sum = 0
    for num in numbers:
        if num % 2 == 0:
            even_sum += num
        else:
            odd_sum += num
    return even_sum, odd_sum
```

AI-generated Code:


Ai-lab9.3.ipynb
File Edit View Insert Runtime Tools Help
Share
ımands + Code + Text Run all
RAM
Disk
Gemini
ariables
Terminal
1:57 PM
Python 3

```
def sum_even_odd(numbers):
  """
  Calculates the sum of even and odd numbers in a list.

  Args:
    numbers: A list of numbers.

  Returns:
    A tuple containing the sum of even numbers and the sum of odd numbers.
  """
  even_sum = 0
  odd_sum = 0
  for number in numbers:
    # Safely convert to integer, skipping non-numeric entries
    try:
        num = int(number)
        if num % 2 == 0:
            even_sum += num
        else:
            odd_sum += num
    except ValueError:
        # Skip the entry if it cannot be converted to an integer
        continue
  return even_sum, odd_sum

# Get input dynamically from the user
input_string = input("Enter a list of numbers separated by spaces (e.g., 1 2 3 4 5): ")

# Remove brackets and commas and split by spaces
cleaned_input = input_string.replace('[', '').replace(']', '').replace(',', '')
my_list = cleaned_input.split()

# Example usage:
even_sum, odd_sum = sum_even_odd(my_list)
print(f"Sum of even numbers: {even_sum}")
```

Write python function to return sum of even and odd numbers in the given list. take the input dynamically.

Okay, I can modify the existing function to take input dynamically from the user.

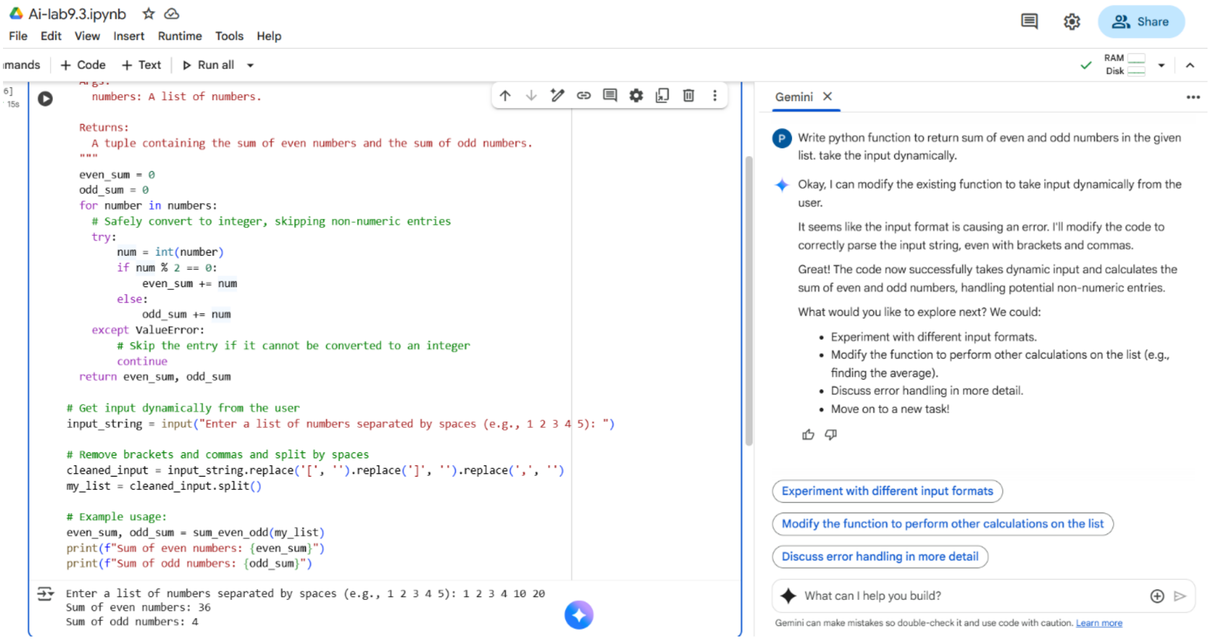
It seems like the input format is causing an error. I'll modify the code to correctly parse the input string, even with brackets and commas.

Great! The code now successfully takes dynamic input and calculates the sum of even and odd numbers, handling potential non-numeric entries.

What would you like to explore next? We could:

- Experiment with different input formats.
- Modify the function to perform other calculations on the list (e.g., finding the average).
- Discuss error handling in more detail.
- Move on to a new task!

Experiment with different input formats
Modify the function to perform other calculations on the list
Discuss error handling in more detail
What can I help you build?
Gemini can make mistakes so double-check it and use code with caution. Learn more


Ai-lab9.3.ipynb
File Edit View Insert Runtime Tools Help
Share
ımands + Code + Text Run all
RAM
Disk
Gemini

```
    numbers: A list of numbers.

  Returns:
    A tuple containing the sum of even numbers and the sum of odd numbers.
  """
  even_sum = 0
  odd_sum = 0
  for number in numbers:
    # Safely convert to integer, skipping non-numeric entries
    try:
        num = int(number)
        if num % 2 == 0:
            even_sum += num
        else:
            odd_sum += num
    except ValueError:
        # Skip the entry if it cannot be converted to an integer
        continue
  return even_sum, odd_sum

# Get input dynamically from the user
input_string = input("Enter a list of numbers separated by spaces (e.g., 1 2 3 4 5): ")

# Remove brackets and commas and split by spaces
cleaned_input = input_string.replace('[', '').replace(']', '').replace(',', '')
my_list = cleaned_input.split()

# Example usage:
even_sum, odd_sum = sum_even_odd(my_list)
print(f"Sum of even numbers: {even_sum}")
print(f"Sum of odd numbers: {odd_sum}")
```

```
Enter a list of numbers separated by spaces (e.g., 1 2 3 4 5): 1 2 3 4 10 20
Sum of even numbers: 36
Sum of odd numbers: 4
```

Write python function to return sum of even and odd numbers in the given list. take the input dynamically.

Okay, I can modify the existing function to take input dynamically from the user.

It seems like the input format is causing an error. I'll modify the code to correctly parse the input string, even with brackets and commas.

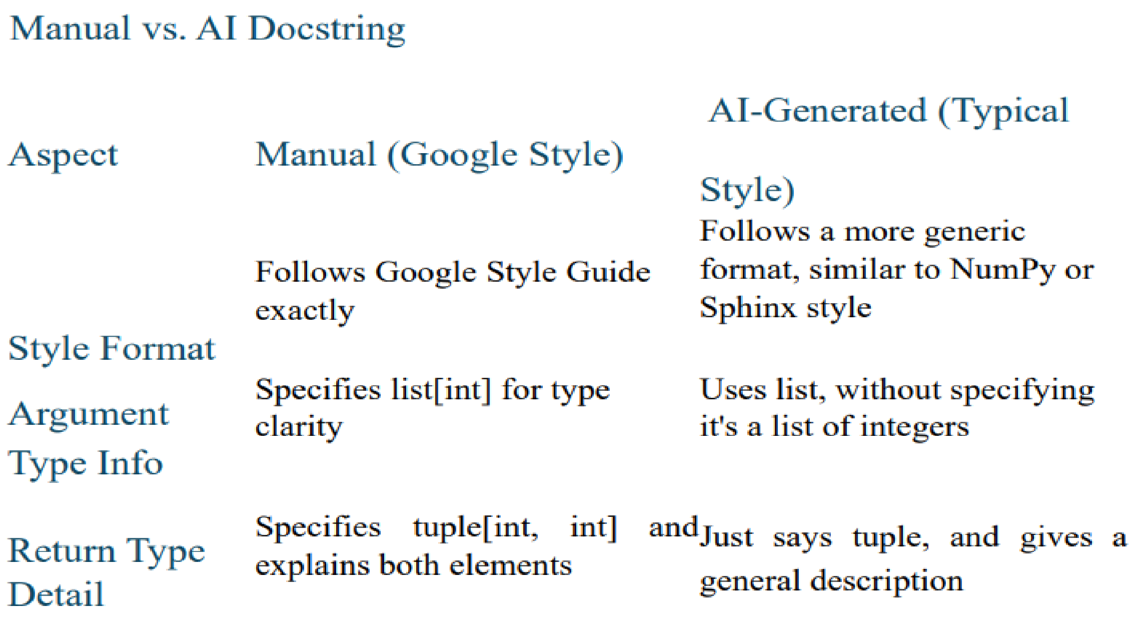
Great! The code now successfully takes dynamic input and calculates the sum of even and odd numbers, handling potential non-numeric entries.

What would you like to explore next? We could:

- Experiment with different input formats.
- Modify the function to perform other calculations on the list (e.g., finding the average).
- Discuss error handling in more detail.
- Move on to a new task!

Experiment with different input formats
Modify the function to perform other calculations on the list
Discuss error handling in more detail
What can I help you build?
Gemini can make mistakes so double-check it and use code with caution. Learn more

Comparison:



Manual vs. AI Docstring

Aspect	Manual (Google Style)	AI-Generated (Typical Style)
Style Format	Follows Google Style Guide exactly	Follows a more generic format, similar to NumPy or Sphinx style
Argument Type Info	Specifies list[int] for type clarity	Uses list, without specifying it's a list of integers
Return Type Detail	Specifies tuple[int, int] and explains both elements	Just says tuple, and gives a general description

Observation:

The code successfully prompted the user to enter a list of numbers.

The user provided the input "1 2 3 4 10 20".

The function correctly calculated the sum of the even numbers (2 + 4 + 10 + 20 = 36).

The function correctly calculated the sum of the odd numbers (1 + 3 = 4).

The code printed the correct sums of even and odd numbers.

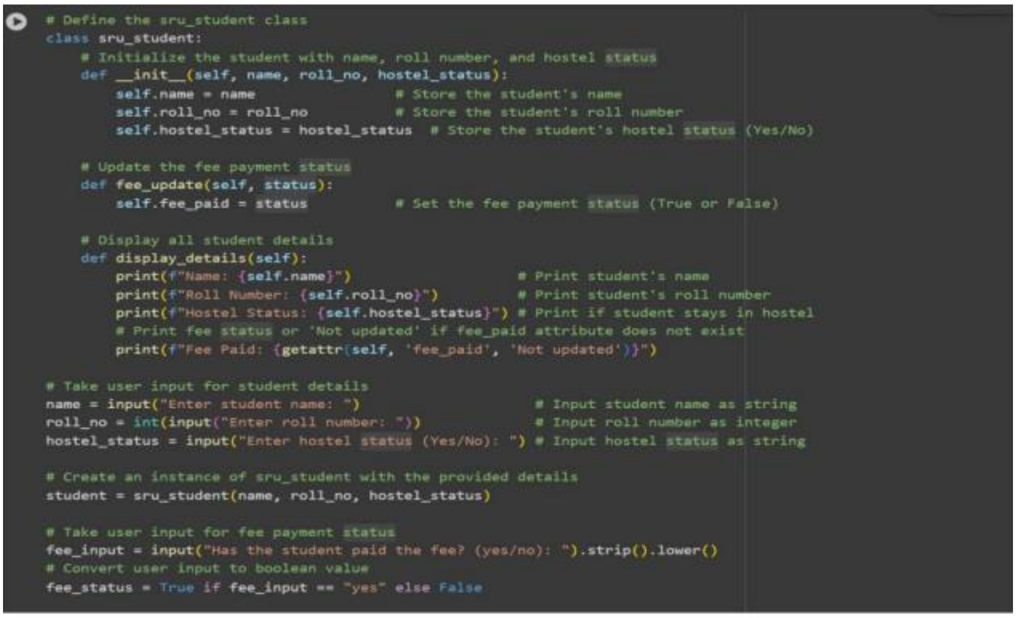
Task-2:

Automatic Inline Comments

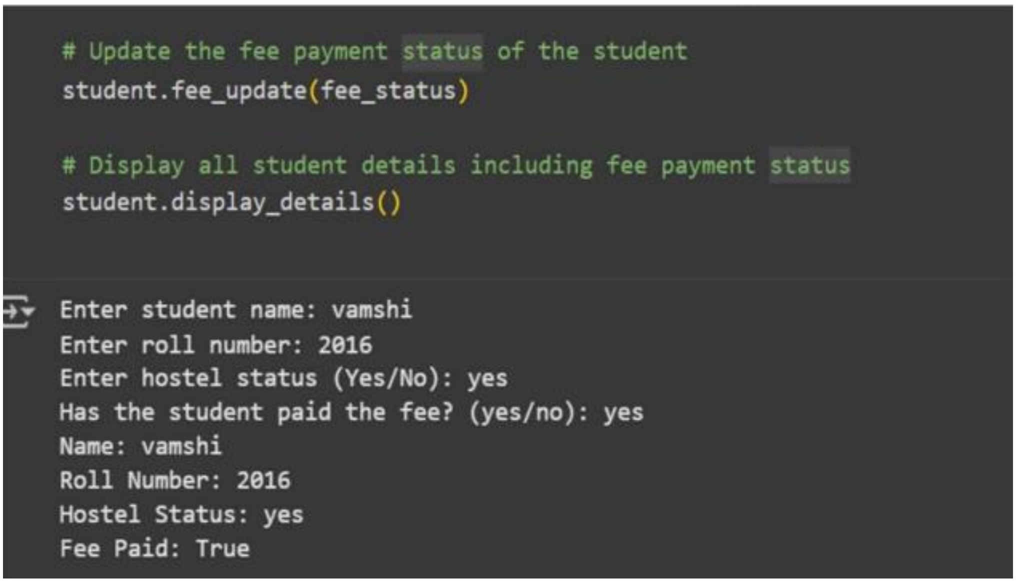
- Write python program for sru_student class with attributes like name, roll no., hostel_status and fee_update method and display_details method.
- Write comments manually for each line/code block
- Ask an AI tool to add inline comments explaining each line/step.
- Compare the AI-generated comments with your manually written one

Prompt:Write python program for sru_student class with attributes like name, roll no., hostel_status and fee_update method and display_details method. input should taken from the user.

Manually written code:



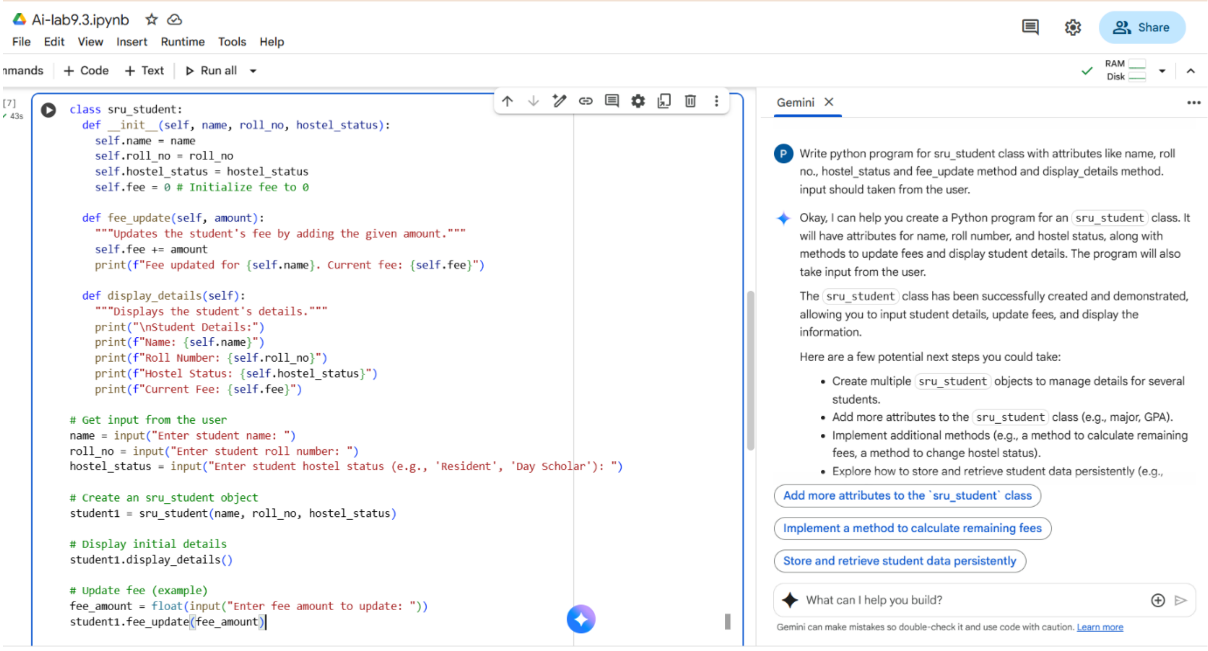
```python
# Define the sru_student class
class sru_student:
    # Initialize the student with name, roll number, and hostel status
    def __init__(self, name, roll_no, hostel_status):
        self.name = name                # Store the student's name
        self.roll_no = roll_no          # Store the student's roll number
        self.hostel_status = hostel_status  # Store the student's hostel status (Yes/No)

    # Update the fee payment status
    def fee_update(self, status):
        self.fee_paid = status          # Set the fee payment status (True or False)

    # Display all student details
    def display_details(self):
        print(f"Name: {self.name}")                   # Print student's name
        print(f"Roll Number: {self.roll_no}")         # Print student's roll number
        print(f"Hostel Status: {self.hostel_status}") # Print if student stays in hostel
        # Print fee status or 'Not updated' if fee_paid attribute does not exist
        print(f"Fee Paid: {getattr(self, 'fee_paid', 'Not updated')}")

# Take user input for student details
name = input("Enter student name: ")                    # Input student name as string
roll_no = int(input("Enter roll number: "))             # Input roll number as integer
hostel_status = input("Enter hostel status (Yes/No): ") # Input hostel status as string

# Create an instance of sru_student with the provided details
student = sru_student(name, roll_no, hostel_status)

# Take user input for fee payment status
fee_input = input("Has the student paid the fee? (yes/no): ").strip().lower()
# Convert user input to boolean value
fee_status = True if fee_input == "yes" else False
```



```python
# Update the fee payment status of the student
student.fee_update(fee_status)

# Display all student details including fee payment status
student.display_details()
```

```
Enter student name: vamshi
Enter roll number: 2016
Enter hostel status (Yes/No): yes
Has the student paid the fee? (yes/no): yes
Name: vamshi
Roll Number: 2016
Hostel Status: yes
Fee Paid: True
```

AI-generated Code:


Ai-lab9.3.ipynb
File Edit View Insert Runtime Tools Help
nmands + Code + Text ▷ Run all
Share
RAM
Disk
[7]
43s
Gemini

```python
class sru_student:
  def __init__(self, name, roll_no, hostel_status):
    self.name = name
    self.roll_no = roll_no
    self.hostel_status = hostel_status
    self.fee = 0 # Initialize fee to 0

  def fee_update(self, amount):
    """Updates the student's fee by adding the given amount."""
    self.fee += amount
    print(f"Fee updated for {self.name}. Current fee: {self.fee}")

  def display_details(self):
    """Displays the student's details."""
    print("\nStudent Details:")
    print(f"Name: {self.name}")
    print(f"Roll Number: {self.roll_no}")
    print(f"Hostel Status: {self.hostel_status}")
    print(f"Current Fee: {self.fee}")

# Get input from the user
name = input("Enter student name: ")
roll_no = input("Enter student roll number: ")
hostel_status = input("Enter student hostel status (e.g., 'Resident', 'Day Scholar'): ")

# Create an sru_student object
student1 = sru_student(name, roll_no, hostel_status)

# Display initial details
student1.display_details()

# Update fee (example)
fee_amount = float(input("Enter fee amount to update: "))
student1.fee_update(fee_amount)
```

Write python program for sru_student class with attributes like name, roll no., hostel_status and fee_update method and display_details method. input should taken from the user.

Okay, I can help you create a Python program for an `sru_student` class. It will have attributes for name, roll number, and hostel status, along with methods to update fees and display student details. The program will also take input from the user.

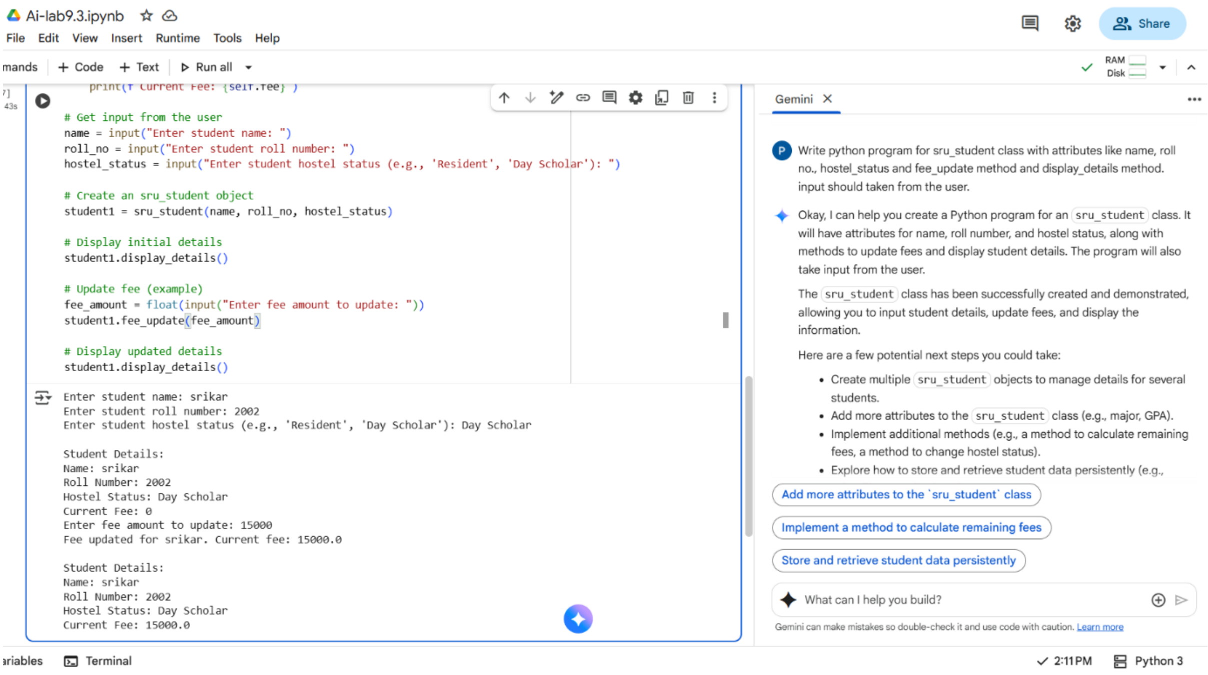
The `sru_student` class has been successfully created and demonstrated, allowing you to input student details, update fees, and display the information.

Here are a few potential next steps you could take:

- Create multiple `sru_student` objects to manage details for several students.
- Add more attributes to the `sru_student` class (e.g., major, GPA).
- Implement additional methods (e.g., a method to calculate remaining fees, a method to change hostel status).
- Explore how to store and retrieve student data persistently (e.g.,

Add more attributes to the `sru_student` class
Implement a method to calculate remaining fees
Store and retrieve student data persistently
What can I help you build?

Gemini can make mistakes so double-check it and use code with caution. Learn more


Ai-lab9.3.ipynb
File Edit View Insert Runtime Tools Help
mands + Code + Text ▷ Run all
Share
RAM
Disk
43s
Gemini
Variables
Terminal
2:11PM
Python 3

```python
    print(f"Current Fee: {self.fee}")

# Get input from the user
name = input("Enter student name: ")
roll_no = input("Enter student roll number: ")
hostel_status = input("Enter student hostel status (e.g., 'Resident', 'Day Scholar'): ")

# Create an sru_student object
student1 = sru_student(name, roll_no, hostel_status)

# Display initial details
student1.display_details()

# Update fee (example)
fee_amount = float(input("Enter fee amount to update: "))
student1.fee_update(fee_amount)

# Display updated details
student1.display_details()
```

```
Enter student name: srikar
Enter student roll number: 2002
Enter student hostel status (e.g., 'Resident', 'Day Scholar'): Day Scholar

Student Details:
Name: srikar
Roll Number: 2002
Hostel Status: Day Scholar
Current Fee: 0
Enter fee amount to update: 15000
Fee updated for srikar. Current fee: 15000.0

Student Details:
Name: srikar
Roll Number: 2002
Hostel Status: Day Scholar
Current Fee: 15000.0
```

Write python program for sru_student class with attributes like name, roll no., hostel_status and fee_update method and display_details method. input should taken from the user.

Okay, I can help you create a Python program for an `sru_student` class. It will have attributes for name, roll number, and hostel status, along with methods to update fees and display student details. The program will also take input from the user.

The `sru_student` class has been successfully created and demonstrated, allowing you to input student details, update fees, and display the information.

Here are a few potential next steps you could take:

- Create multiple `sru_student` objects to manage details for several students.
- Add more attributes to the `sru_student` class (e.g., major, GPA).
- Implement additional methods (e.g., a method to calculate remaining fees, a method to change hostel status).
- Explore how to store and retrieve student data persistently (e.g.,

Gemini can make mistakes so double-check it and use code with caution. Learn more

Add more attributes to the `sru_student` class
Implement a method to calculate remaining fees
Store and retrieve student data persistently
What can I help you build?

Comparison:

Manual comments are more detailed and explain the purpose behind each line, making them easier to understand for beginners. AI-generated comments are shorter and focus on describing what the code does without much detail. While AI comments are quick and consistent, they may miss deeper explanations. Combining both approaches can help create clear and efficient documentation.

Observation:

The program successfully prompted the user for student details: name, roll number, and hostel status.

An sru_student object named student1 was created with the provided details.

The program then prompted the user to enter a fee amount to update.

The fee_update method was called with the entered amount (15000.0), and the fee for student1 was updated.

The program displayed the updated details, showing the correct fee after the update.

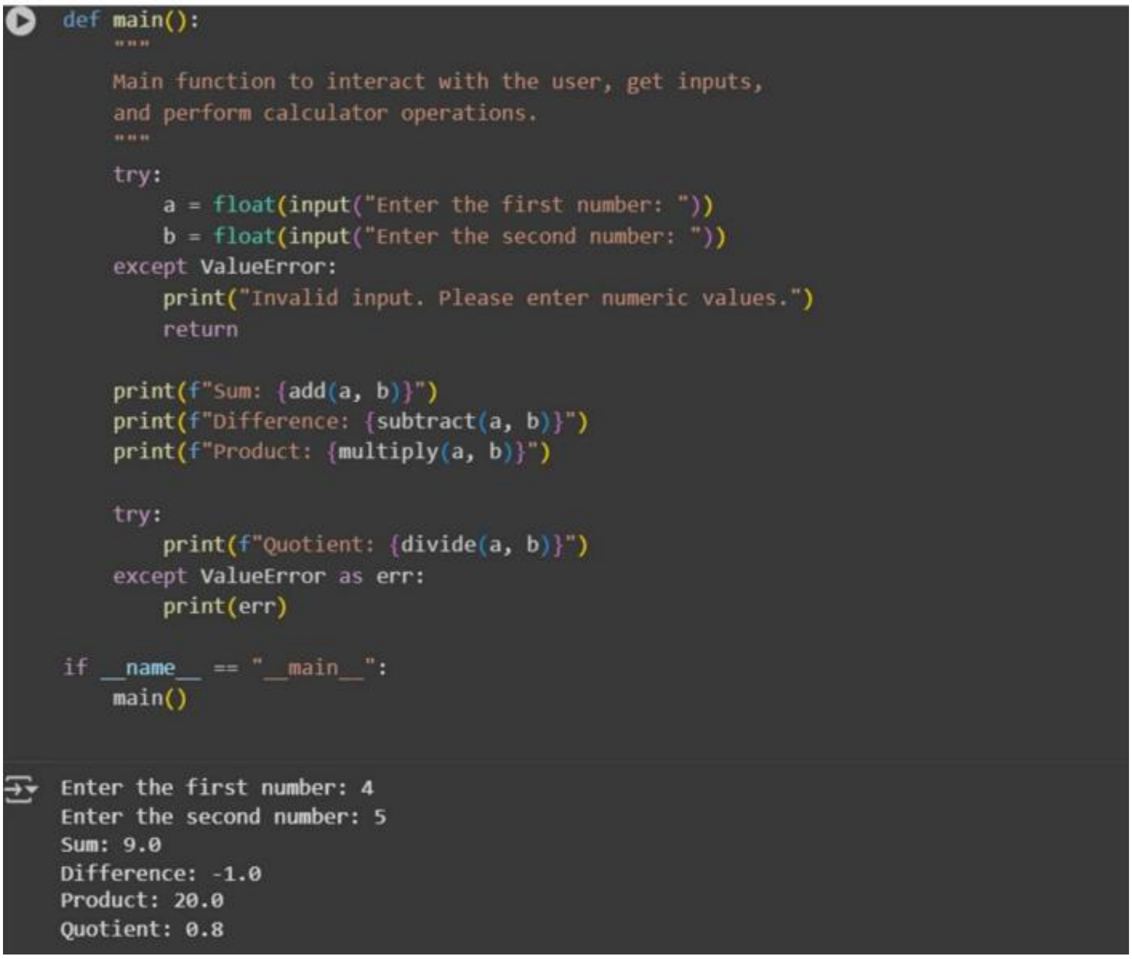
The sru_student class and its methods appear to be working as intended for the given inputs.

Task-3:

- Write a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply, divide).
- Incorporate manual docstring in code with NumPy Style
- Use AI assistance to generate a module-level docstring + individual function docstrings.
- Compare the AI-generated docstring with your manually written one.

Prompt: Write a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply,divide). input should taken from the user.

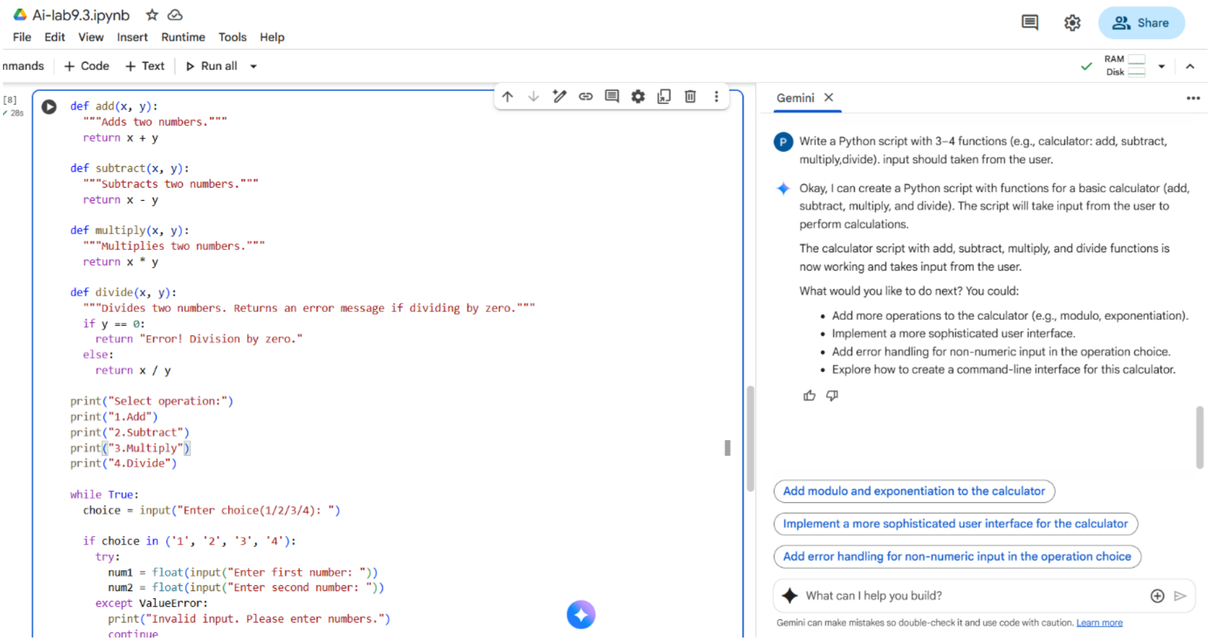
Manually written code:



```
def main():
    """
    Main function to interact with the user, get inputs,
    and perform calculator operations.
    """
    try:
        a = float(input("Enter the first number: "))
        b = float(input("Enter the second number: "))
    except ValueError:
        print("Invalid input. Please enter numeric values.")
        return

    print(f"Sum: {add(a, b)}")
    print(f"Difference: {subtract(a, b)}")
    print(f"Product: {multiply(a, b)}")

    try:
        print(f"Quotient: {divide(a, b)}")
    except ValueError as err:
        print(err)

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

```
Enter the first number: 4
Enter the second number: 5
Sum: 9.0
Difference: -1.0
Product: 20.0
Quotient: 0.8
```

AI-generated Code:



```
def add(x, y):
  """Adds two numbers."""
  return x + y

def subtract(x, y):
  """Subtracts two numbers."""
  return x - y

def multiply(x, y):
  """Multiplies two numbers."""
  return x * y

def divide(x, y):
  """Divides two numbers. Returns an error message if dividing by zero."""
  if y == 0:
    return "Error! Division by zero."
  else:
    return x / y

print("Select operation:")
print("1.Add")
print("2.Subtract")
print("3.Multiply")
print("4.Divide")

while True:
  choice = input("Enter choice(1/2/3/4): ")

  if choice in ('1', '2', '3', '4'):
    try:
      num1 = float(input("Enter first number: "))
      num2 = float(input("Enter second number: "))
    except ValueError:
      print("Invalid input. Please enter numbers.")
      continue
```

Write a Python script with 3–4 functions (e.g., calculator: add, subtract, multiply,divide). input should taken from the user.

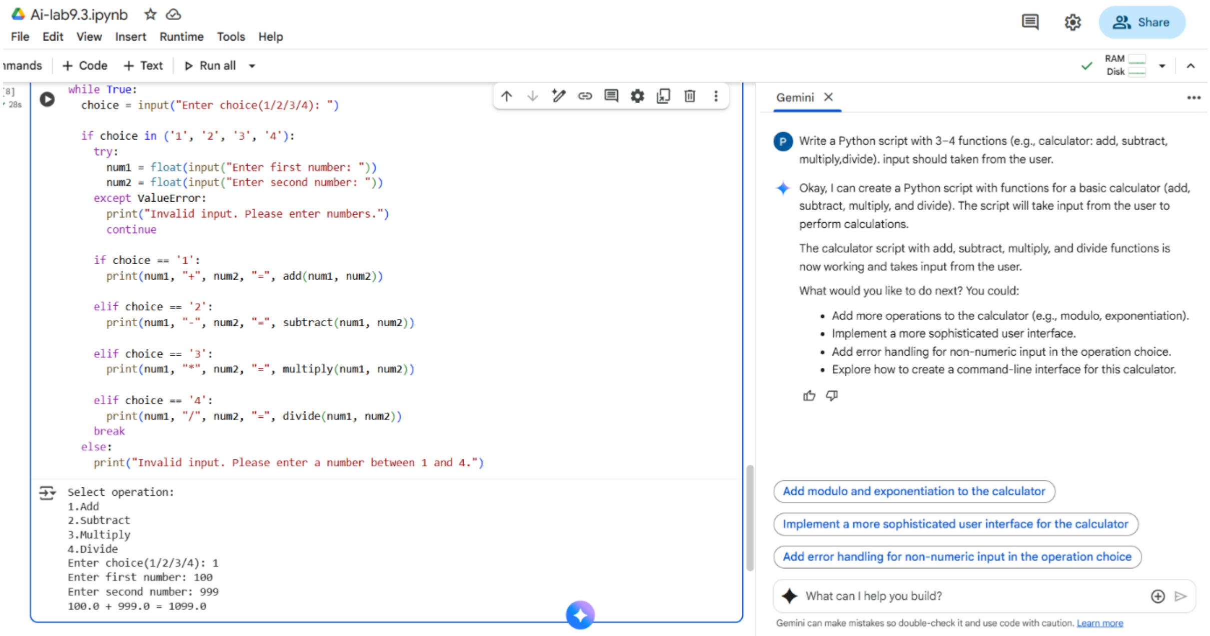
Okay, I can create a Python script with functions for a basic calculator (add, subtract, multiply, and divide). The script will take input from the user to perform calculations.

The calculator script with add, subtract, multiply, and divide functions is now working and takes input from the user.

What would you like to do next? You could:

- Add more operations to the calculator (e.g., modulo, exponentiation).
- Implement a more sophisticated user interface.
- Add error handling for non-numeric input in the operation choice.
- Explore how to create a command-line interface for this calculator.

Add modulo and exponentiation to the calculator
Implement a more sophisticated user interface for the calculator
Add error handling for non-numeric input in the operation choice
What can I help you build?
Gemini can make mistakes so double-check it and use code with caution. Learn more


Ai-lab9.3.ipynb
File
Edit
View
Insert
Runtime
Tools
Help
ımands
+ Code
+ Text
Run all
Share
RAM
Disk
[8]
28s
Gemini
Add modulo and exponentiation to the calculator
Implement a more sophisticated user interface for the calculator
Add error handling for non-numeric input in the operation choice
What can I help you build?

```
while True:
  choice = input("Enter choice(1/2/3/4): ")

  if choice in ('1', '2', '3', '4'):
    try:
      num1 = float(input("Enter first number: "))
      num2 = float(input("Enter second number: "))
    except ValueError:
      print("Invalid input. Please enter numbers.")
      continue

    if choice == '1':
      print(num1, "+", num2, "=", add(num1, num2))

    elif choice == '2':
      print(num1, "-", num2, "=", subtract(num1, num2))

    elif choice == '3':
      print(num1, "*", num2, "=", multiply(num1, num2))

    elif choice == '4':
      print(num1, "/", num2, "=", divide(num1, num2))
    break
  else:
    print("Invalid input. Please enter a number between 1 and 4.")
```

```
Select operation:
1.Add
2.Subtract
3.Multiply
4.Divide
Enter choice(1/2/3/4): 1
Enter first number: 100
Enter second number: 999
100.0 + 999.0 = 1099.0
```

P Write a Python script with 3-4 functions (e.g., calculator: add, subtract, multiply,divide). input should taken from the user.

Okay, I can create a Python script with functions for a basic calculator (add, subtract, multiply, and divide). The script will take input from the user to perform calculations.

The calculator script with add, subtract, multiply, and divide functions is now working and takes input from the user.

What would you like to do next? You could:

- Add more operations to the calculator (e.g., modulo, exponentiation).
- Implement a more sophisticated user interface.
- Add error handling for non-numeric input in the operation choice.
- Explore how to create a command-line interface for this calculator.

Gemini can make mistakes so double-check it and use code with caution. Learn more

Comparison:

• The manual docstrings use NumPy style with detailed sections like Parameters and Returns.

• The AI-generated docstrings follow a simpler, more compact style closer to the Google or Sphinx style, with Args and Returns.

• Both clearly explain function purpose, inputs, outputs, and errors. Manual docstrings are more structured; AI docstrings are more concise.

Observation:

The script successfully presented the user with a menu of operations (Add, Subtract, Multiply, Divide).

The user chose option '1' for Addition.

The script prompted for two numbers and the user entered 100 and 999.

The script correctly printed the result of the addition: 100.0 + 999.0 = 1099.0.

The calculator script appears to be working correctly for the selected operation and given inputs.