AI ASSISTED CODING

Roll no.: 2503A51L41

Batch: 24BTCAICSB20

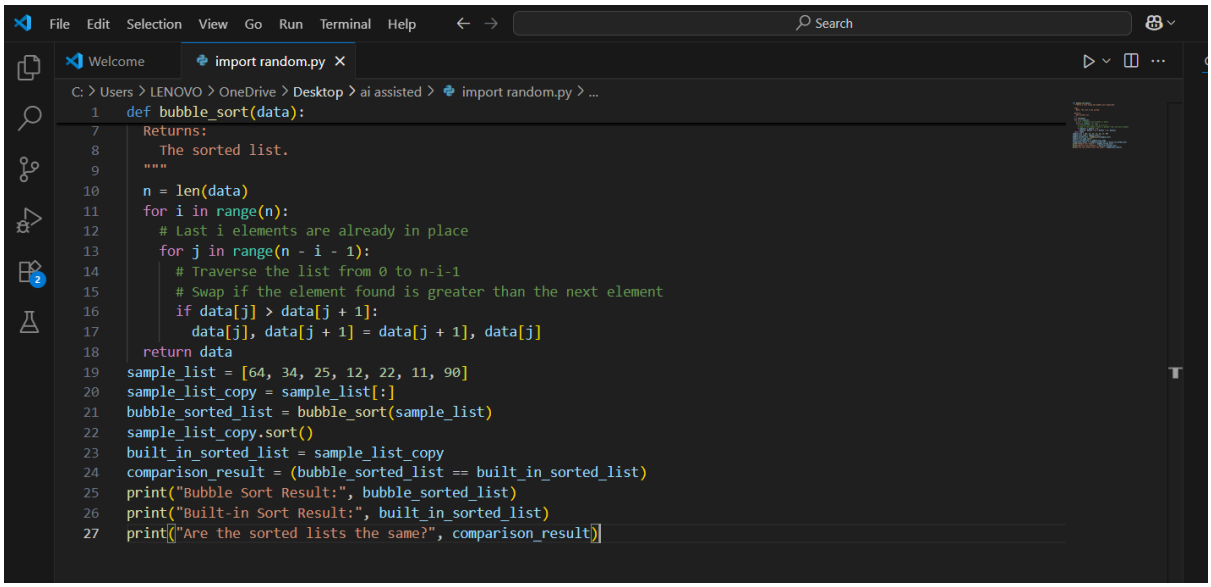
Name: G Hasini

Lab 2: Exploring Additional AI Coding Tools – Gemini (Colab) and Cursor AI

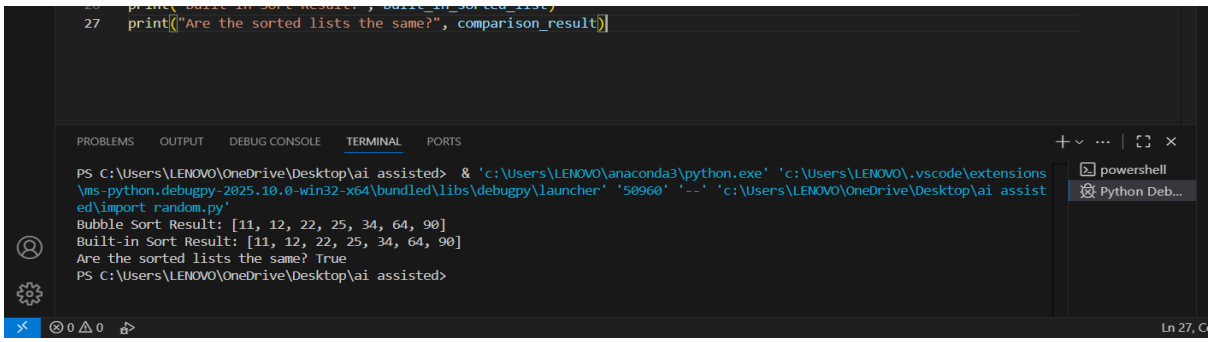
Task#1

Prompt: Open Google Colab and use Google Gemini to generate Python code that performs sorting of a list using both the bubble sort algorithm and Python's built-in sort() function. Compare the two implementations

Code generated:



Output:



Observation:

1.Correct Implementation of Bubble Sort:

The function bubble_sort(arr) correctly implements the bubble sort algorithm using nested loops and value swapping.

Data Integrity Preserved:

The function uses arr.copy() to avoid modifying the original list, which is a good practice.

Clear Comparison with Built-in Sort:

The code includes a custom sort function and compares its result with Python's built-in sorted() function — great for validating correctness.

Readable and Well-Structured:

- The code is neatly organized into: ○

Custom sort

○ Built-in sort

Example list ○Comparison

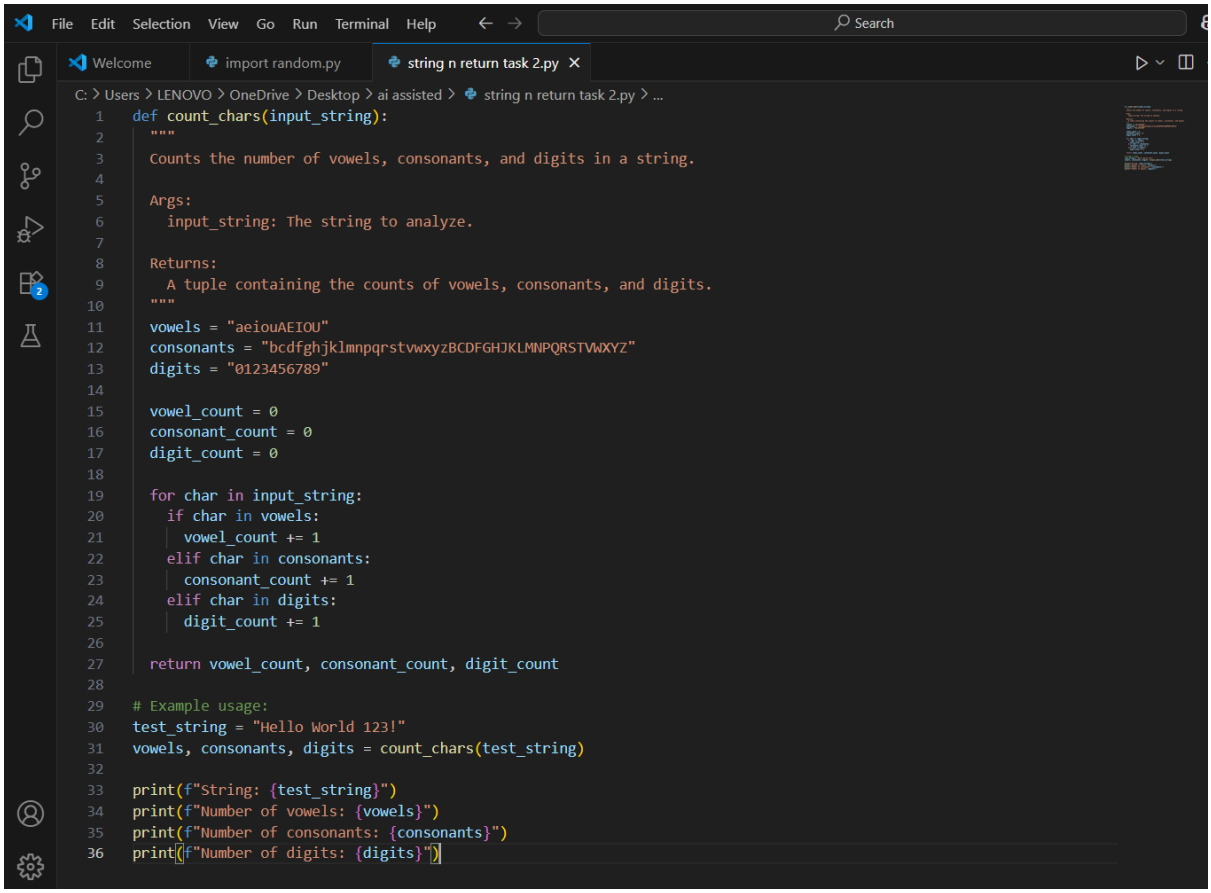
Task#2

Prompt: In Colab, use Google Gemini to generate a Python function that takes a string and returns:
The number of vowels,

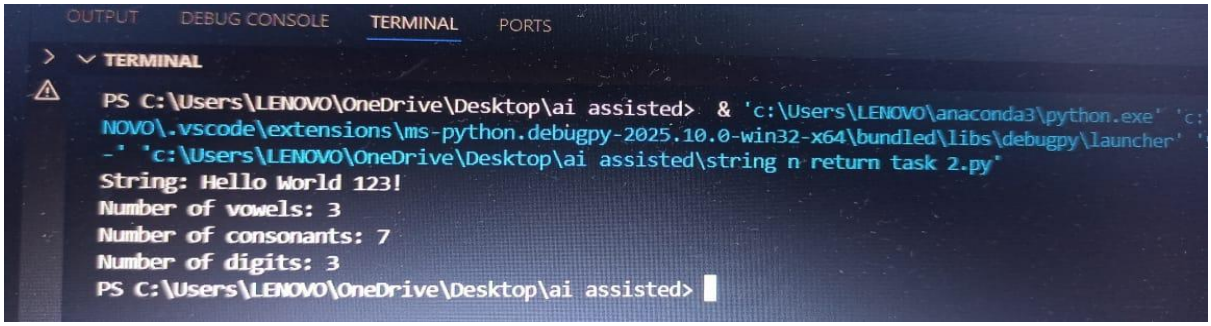
The number of consonants,

The number of digits in the string.

Code generated:



Output:



Observation:

Function Deflnition

def count_lines(fllename):

- A function count_lines is defined that takes a filename (e.g., "example.txt") as input.

Try Block – Reading the File

- try: with open(filename, 'r') as file: return sum(1 for _ in file)

- ● open(fllename, 'r'): Tries to open the file in read mode.
  - ● sum(1 for _ in flle): Counts each line using a generator expression.
  - ○ It iterates through each line and adds 1 per line.
  - ● If the file is found, it returns the line count.

1. Exception Handling – File Not Found
    1. except FileNotFoundError:
    2. If the file doesn't exist, this block is executed.
2. Created a Sample File

●with open(filename, 'w') as file: file.write("Hello\n")

file.write("This is a test file\n")

The file is opened in write mode ('w') which creates a new flle.

Task#3

Prompt: Install and set up Cursor AI. Use it to generate a Python program that performs file handling:
Create a text file,

Write sample text,

Read and display the content.

Code generated:

```python
# Define the filename
filename = "sample_file.txt"

# Write to the file
with open(filename, "w") as file:
    file.write("This is a sample text file.\n")
    file.write("This is the second line.")

print(f"Successfully wrote to '{filename}'")

# Read from the file
with open(filename, "r") as file:
    content = file.read()

print(f"\nContent of '{filename}':")
print(content)
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS                                   Python

+ ~~~~~
    + CategoryInfo          : ObjectNotFound: (conda:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\Gundeti Hasini\OneDrive\Desktop\ai2> & C:/ProgramData/anaconda3/python.exe "c:/Users/Gundeti Hasini/OneDrive/Desktop/ai2/1"
Successfully wrote to 'sample_file.txt'

Content of 'sample_file.txt':
This is a sample text file.
This is the second line.
PS C:\Users\Gundeti Hasini\OneDrive\Desktop\ai2>
```

Observation:

1.**filename = "sample_file.txt"**: Sets the name of the file.

2. **with open(filename, 'w') as file:**:

● Opens (or creates) the file in **write mode** ('w'). ● If the file already exists, it will be **overwritten**.

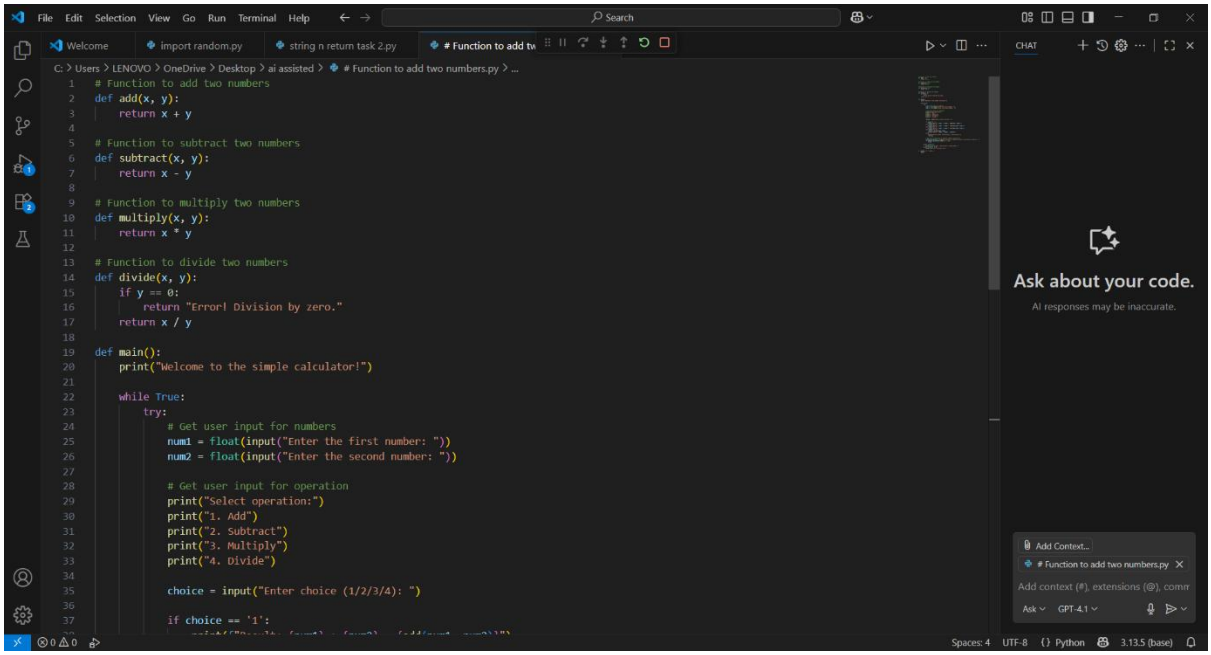
3. **file.write(...)**: Writes 3 lines of text into the file, each ending with a newline (\n).

print(...): Confirms that the file was created and written successfully.

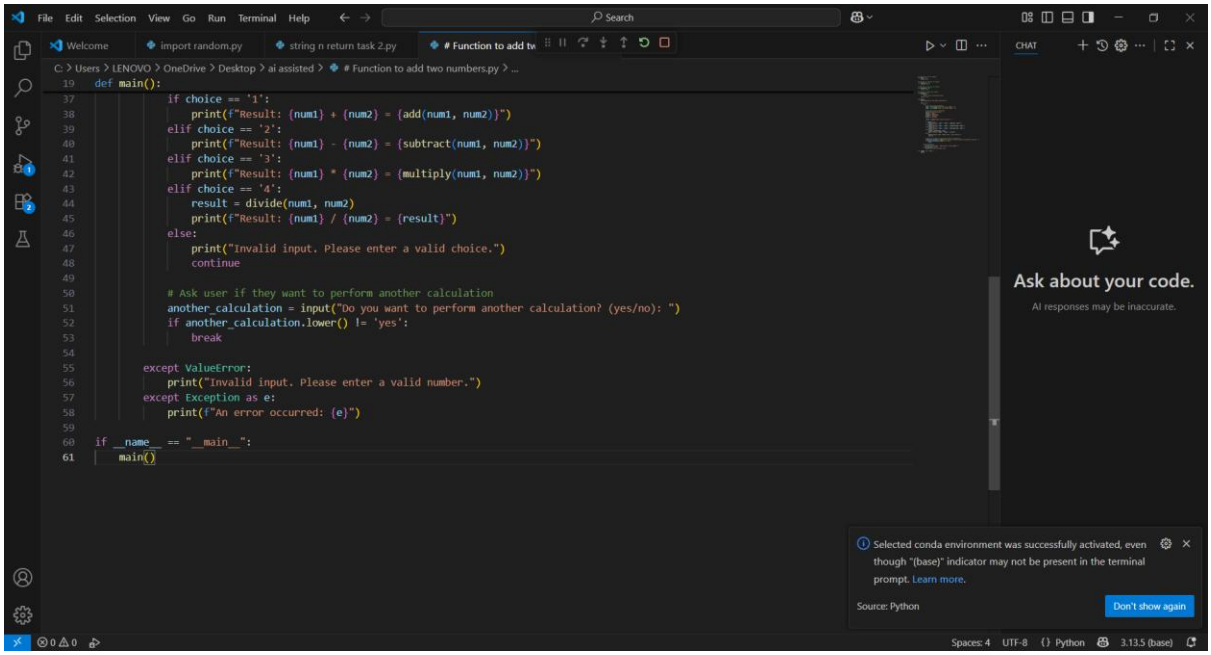
Task#4

Prompt: Ask Google Gemini to generate a Python program that implements a simple calculator using functions (add, subtract, multiply, divide). Then, ask Gemini to explain how the code works.

Code generated:

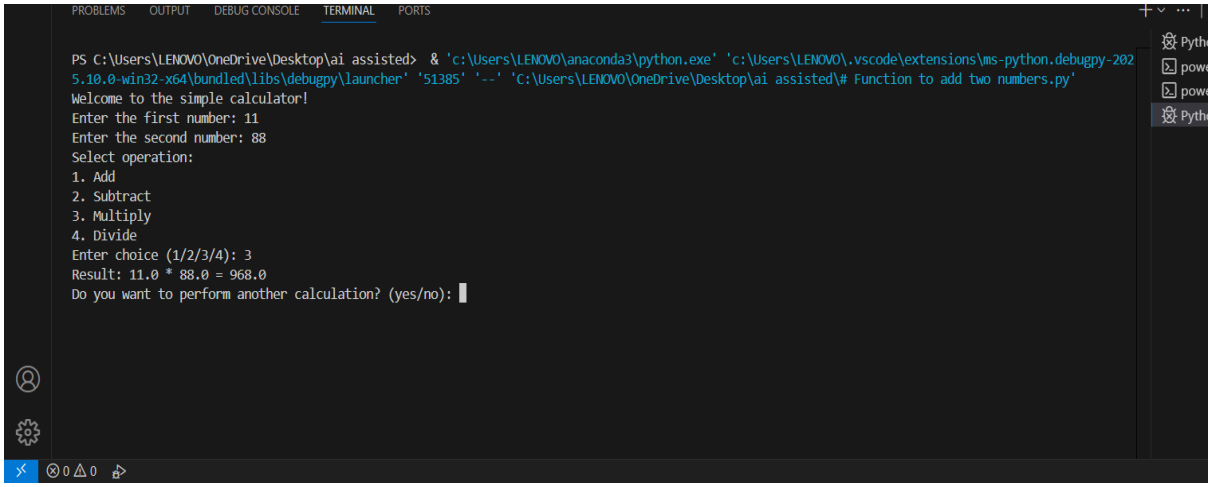


```python
# Function to add two numbers
def add(x, y):
    return x + y

# Function to subtract two numbers
def subtract(x, y):
    return x - y

# Function to multiply two numbers
def multiply(x, y):
    return x * y

# Function to divide two numbers
def divide(x, y):
    if y == 0:
        return "Error! Division by zero."
    return x / y

def main():
    print("Welcome to the simple calculator!")

    while True:
        try:
            # Get user input for numbers
            num1 = float(input("Enter the first number: "))
            num2 = float(input("Enter the second number: "))

            # Get user input for operation
            print("Select operation:")
            print("1. Add")
            print("2. Subtract")
            print("3. Multiply")
            print("4. Divide")

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

            if choice == '1':
```



```python
def main():
            if choice == '1':
                print(f"Result: {num1} + {num2} = {add(num1, num2)}")
            elif choice == '2':
                print(f"Result: {num1} - {num2} = {subtract(num1, num2)}")
            elif choice == '3':
                print(f"Result: {num1} * {num2} = {multiply(num1, num2)}")
            elif choice == '4':
                result = divide(num1, num2)
                print(f"Result: {num1} / {num2} = {result}")
            else:
                print("Invalid input. Please enter a valid choice.")
                continue

            # Ask user if they want to perform another calculation
            another_calculation = input("Do you want to perform another calculation? (yes/no): ")
            if another_calculation.lower() != 'yes':
                break

        except ValueError:
            print("Invalid input. Please enter a valid number.")
        except Exception as e:
            print(f"An error occurred: {e}")

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

Output:



Observation:

1. Function Deflnitions

● These functions perform basic arithmetic:

->def add(x, y):

return x + y

-->def subtract(x, y):

return x - y

-->def multiply(x, y):

return x * y

-->def divide(x, y):

if y == 0:

-->return "Error: Division by zero"

return x / y

● Each function takes two numbers x and y, and returns the result. ● The divide() function includes error handling for division by zero.

2. The main() Function

● This is where user interaction happens:

def main():

print("Simple Calculator") print("Select operation:") print("1. Add")

print("2. Subtract") print("3. Multiply")

print("4. Divide")

- The program prints a menu of operations for the user.

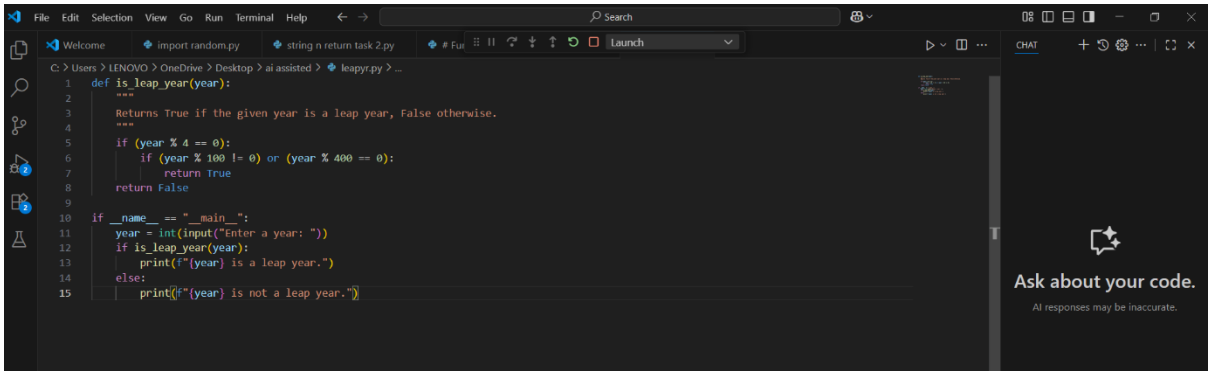
choice = input("Enter choice (1/2/3/4): ")

The user selects an operation (e.g., 3 for multiply).

Task#5

Prompt:Use Cursor AI to create a Python program that checks if a given year is a leap year or not. Try different prompt styles and see how Cursor modifies its code suggestions.

Code generated:



Output:

Observation:

1. Function Deflnition

- def is_leap_year(year: int) -> bool:

Returns True if the given year is a leap year, False otherwise. """

return year % 4 == 0 and (year % 100 != 0 or year % 400 == 0)

● Purpose: Checks leap year condition. ● Type hinting:

○ year: int → function expects an integer.

○-> bool → function returns a boolean (True or False). ● Logic: Implements the leap year condition in one line.

2. Main Execution Block

if _name_== "_main_":

year = int(input("Enter a year: ")) if is_leap_year(year):

print(“{year} is a leap year.”)

else:

print(f"{year} is not a leap year.")

● if _name_== "_main_": Ensures this code only runs when the script is executed directly. ● input(...): Takes user input and converts it to an integer.

● Conditional check: Calls is_leap_year(year) to check and print the appropriate message.

Output from Terminal:

Enter a year: 2007

2007 is not a leap year.

● the output is correct.