Assignment 4.3

Task 1: Zero-Shot Prompting – Leap Year Check

Scenario

Zero-shot prompting involves giving instructions without providing examples.

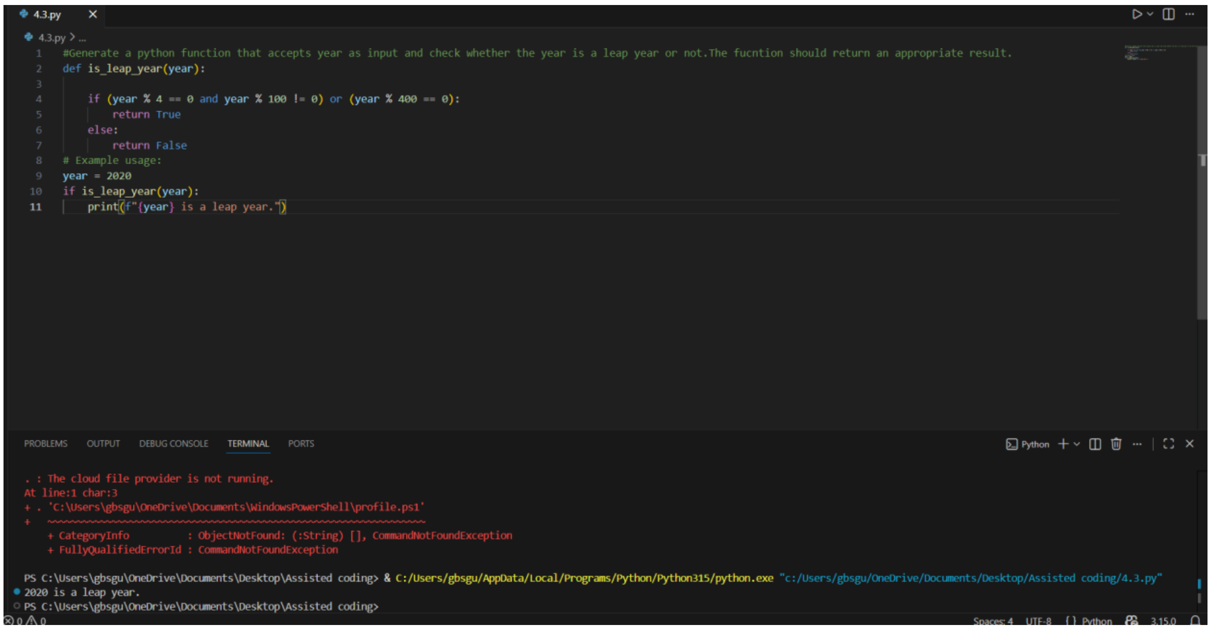
Task Description

Use zero-shot prompting to instruct an AI tool to generate a Python function that:

• Accepts a year as input

• Checks whether the given year is a leap year

• Returns an appropriate result



```python
#Generate a python function that accepts year as input and check whether the year is a leap year or not.The fucntion should return an appropriate result.
def is_leap_year(year):

    if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
        return True
    else:
        return False
# Example usage:
year = 2020
if is_leap_year(year):
    print(f"{year} is a leap year.")
```

```
. : The cloud file provider is not running.
At line:1 char:3
+ . 'C:\Users\gbsgu\OneDrive\Documents\WindowsPowerShell\profile.ps1'
+   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3.py"
2020 is a leap year.
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding>
```

Task 2: One-Shot Prompting – Centimeters to Inches Conversion

Scenario

One-shot prompting guides AI using a single example.

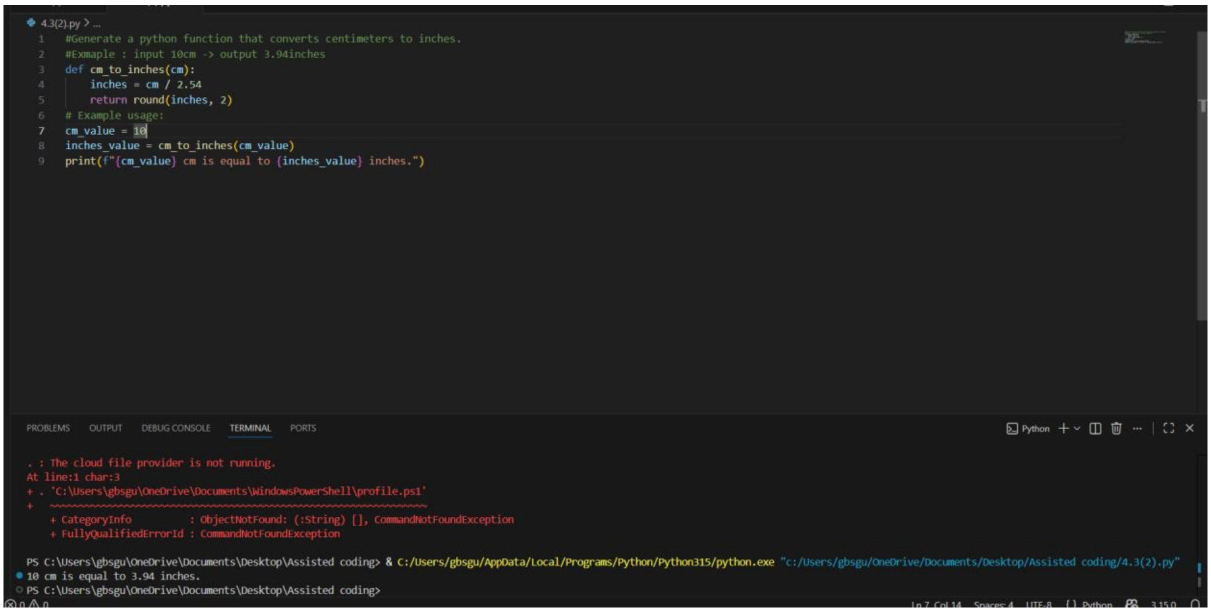
Task Description

Use one-shot prompting by providing one input-output example to generate a Python

function that:

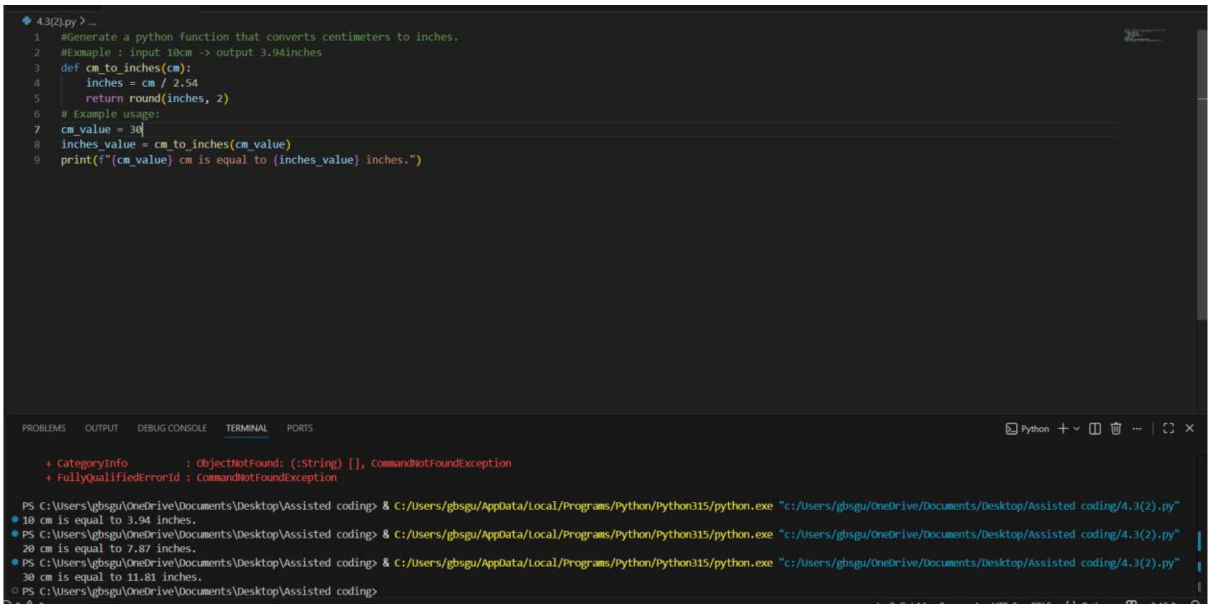
• Converts centimeters to inches

• Uses the correct mathematical formula



```python
#Generate a python function that converts centimeters to inches.
#Exmaple : input 10cm -> output 3.94inches
def cm_to_inches(cm):
    inches = cm / 2.54
    return round(inches, 2)
# Example usage:
cm_value = 10
inches_value = cm_to_inches(cm_value)
print(f"{cm_value} cm is equal to {inches_value} inches.")
```

```
. : The cloud file provider is not running.
At line:1 char:3
+ . 'C:\Users\gbsgu\OneDrive\Documents\WindowsPowerShell\profile.ps1'
+   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(2).py"
10 cm is equal to 3.94 inches.
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding>
```



```python
#Generate a python function that converts centimeters to inches.
#Exmaple : input 10cm -> output 3.94inches
def cm_to_inches(cm):
    inches = cm / 2.54
    return round(inches, 2)
# Example usage:
cm_value = 30
inches_value = cm_to_inches(cm_value)
print(f"{cm_value} cm is equal to {inches_value} inches.")
```

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

PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(2).py"
10 cm is equal to 3.94 inches.
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(2).py"
20 cm is equal to 7.87 inches.
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(2).py"
30 cm is equal to 11.81 inches.
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding>
```

Task 3: Few-Shot Prompting – Name Formatting

Scenario

Few-shot prompting improves accuracy by providing multiple examples.

Task Description

Use few-shot prompting with 2–3 examples to generate a Python function that:

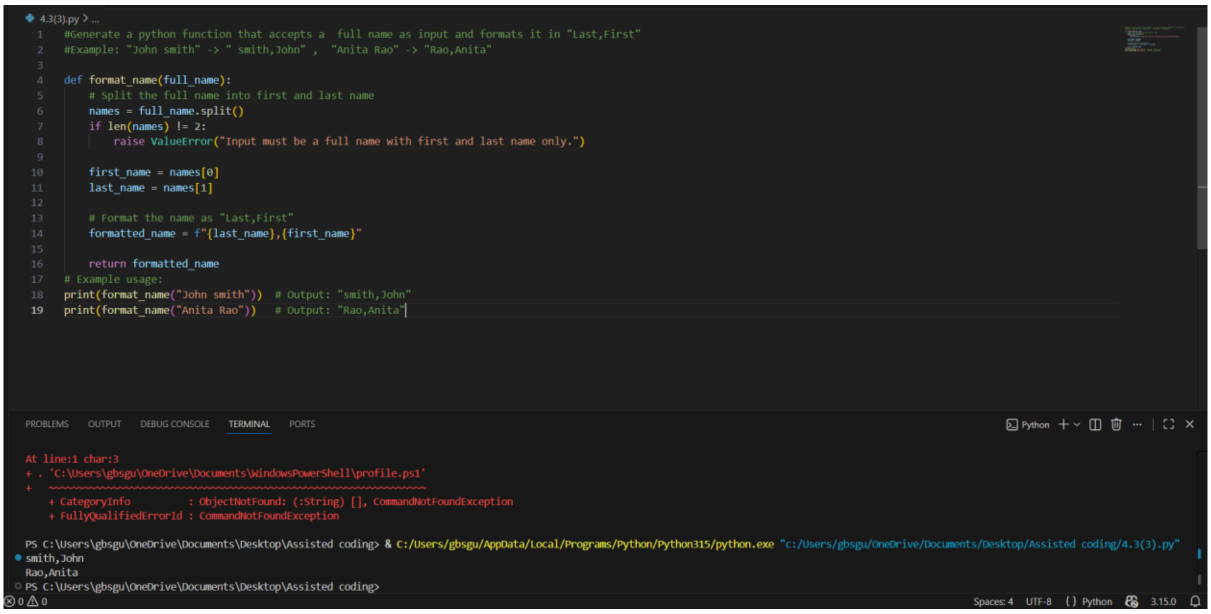
• Accepts a full name as input

• Formats it as “Last, First”

Example formats:

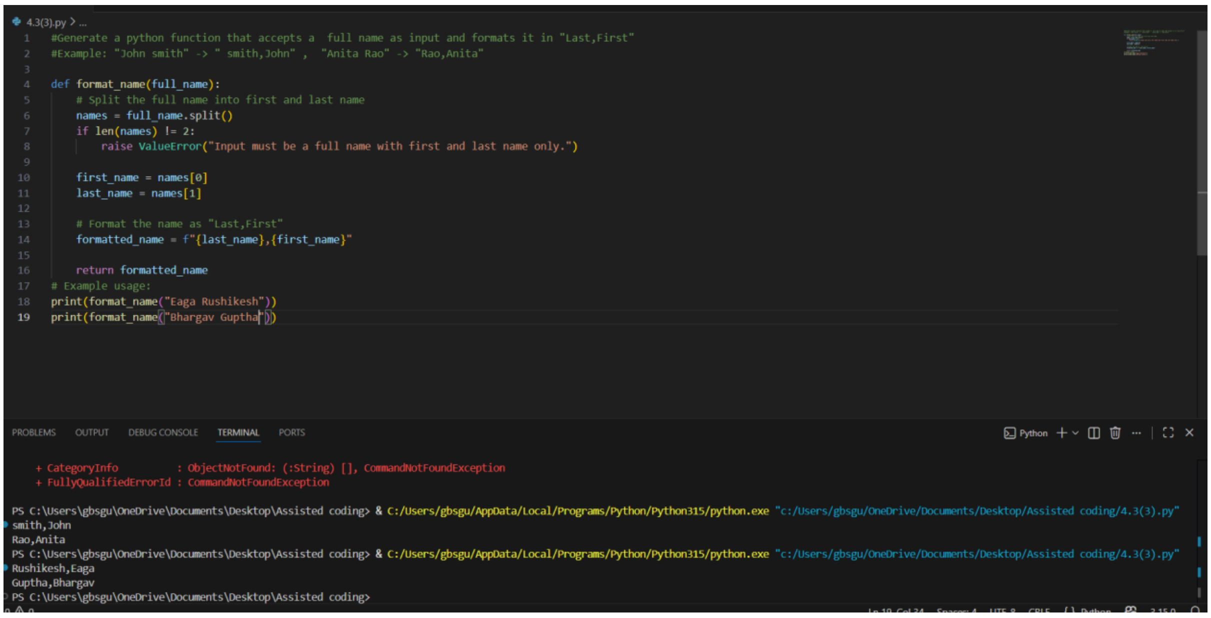
• "John Smith" → "Smith, John"

• "Anita Rao" → "Rao, Anita"



```python
#Generate a python function that accepts a  full name as input and formats it in "Last,First"
#Example: "John smith" -> " smith,John" ,  "Anita Rao" -> "Rao,Anita"

def format_name(full_name):
    # Split the full name into first and last name
    names = full_name.split()
    if len(names) != 2:
        raise ValueError("Input must be a full name with first and last name only.")

    first_name = names[0]
    last_name = names[1]

    # Format the name as "Last,First"
    formatted_name = f"{last_name},{first_name}"

    return formatted_name
# Example usage:
print(format_name("John smith"))  # Output: "smith,John"
print(format_name("Anita Rao"))   # Output: "Rao,Anita"
```

```
At line:1 char:3
+ . 'C:\Users\gbsgu\OneDrive\Documents\WindowsPowerShell\profile.ps1'
+   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(3).py"
smith,John
Rao,Anita
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding>
```



```python
#Generate a python function that accepts a  full name as input and formats it in "Last,First"
#Example: "John smith" -> " smith,John" ,  "Anita Rao" -> "Rao,Anita"

def format_name(full_name):
    # Split the full name into first and last name
    names = full_name.split()
    if len(names) != 2:
        raise ValueError("Input must be a full name with first and last name only.")

    first_name = names[0]
    last_name = names[1]

    # Format the name as "Last,First"
    formatted_name = f"{last_name},{first_name}"

    return formatted_name
# Example usage:
print(format_name("Eaga Rushikesh"))
print(format_name("Bhargav Guptha"))
```

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

PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(3).py"
smith,John
Rao,Anita
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(3).py"
Rushikesh,Eaga
Guptha,Bhargav
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding>
```

Task 4: Comparative Analysis – Zero-Shot vs Few-Shot

Scenario

Different prompt strategies may produce different code quality.

Task Description• Use zero-shot prompting to generate a function that counts vowels in a string

• Use few-shot prompting for the same problem

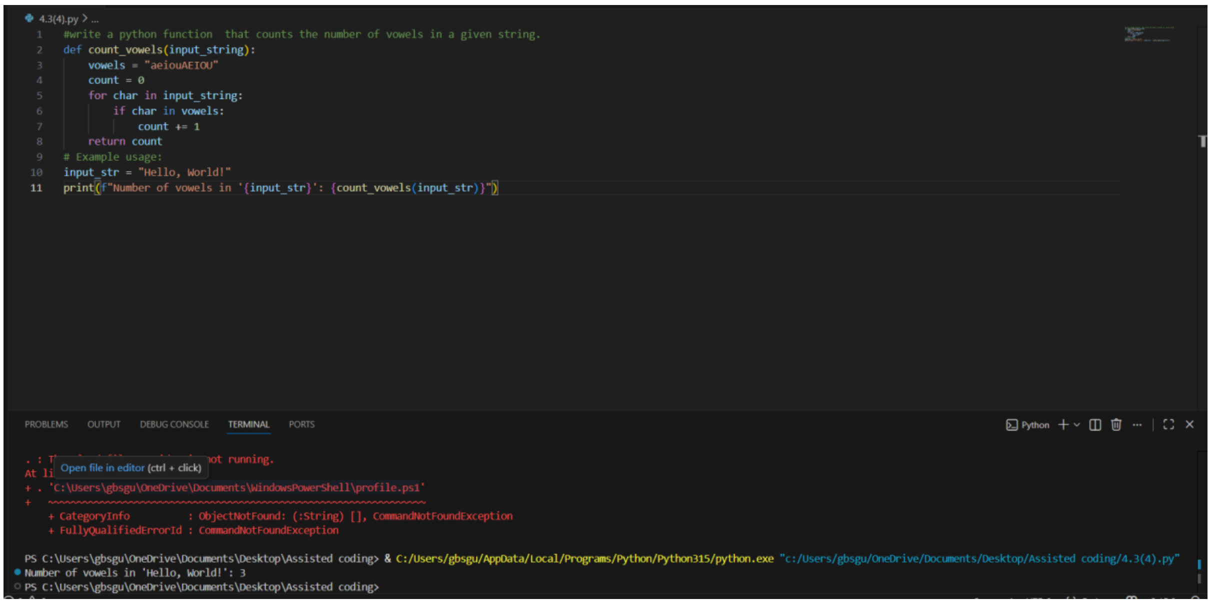
• Compare both outputs based on:

o Accuracy

o Readability

o Logical clarity

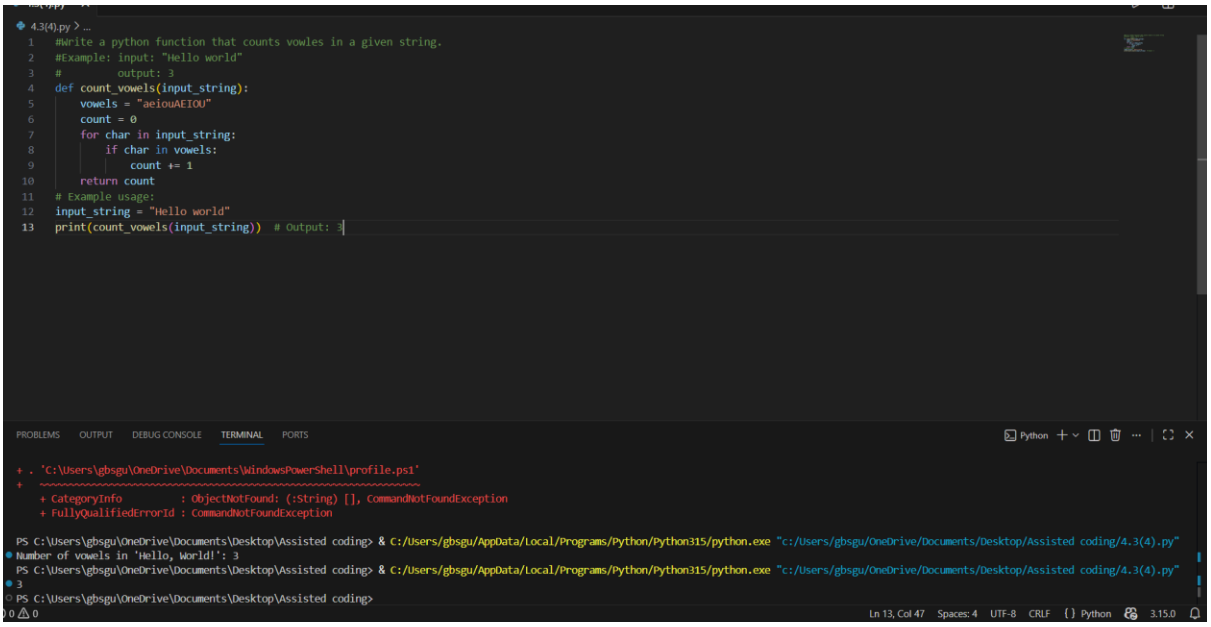
Zero shot prompt:



```python
#write a python function  that counts the number of vowels in a given string.
def count_vowels(input_string):
    vowels = "aeiouAEIOU"
    count = 0
    for char in input_string:
        if char in vowels:
            count += 1
    return count
# Example usage:
input_str = "Hello, World!"
print(f"Number of vowels in '{input_str}': {count_vowels(input_str)}")
```

```
. : T                                    ot running.
At li
+ . 'C:\Users\gbsgu\OneDrive\Documents\WindowsPowerShell\profile.ps1'
+   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(4).py"
Number of vowels in 'Hello, World!': 3
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding>
```

Few shot prompt:



```python
#Write a python function that counts vowles in a given string.
#Example: input: "Hello world"
#         output: 3
def count_vowels(input_string):
    vowels = "aeiouAEIOU"
    count = 0
    for char in input_string:
        if char in vowels:
            count += 1
    return count
# Example usage:
input_string = "Hello world"
print(count_vowels(input_string))  # Output: 3
```

```
+ . 'C:\Users\gbsgu\OneDrive\Documents\WindowsPowerShell\profile.ps1'
+   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(4).py"
Number of vowels in 'Hello, World!': 3
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(4).py"
3
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding>
```

Task 5: Few-Shot Prompting – File Handling

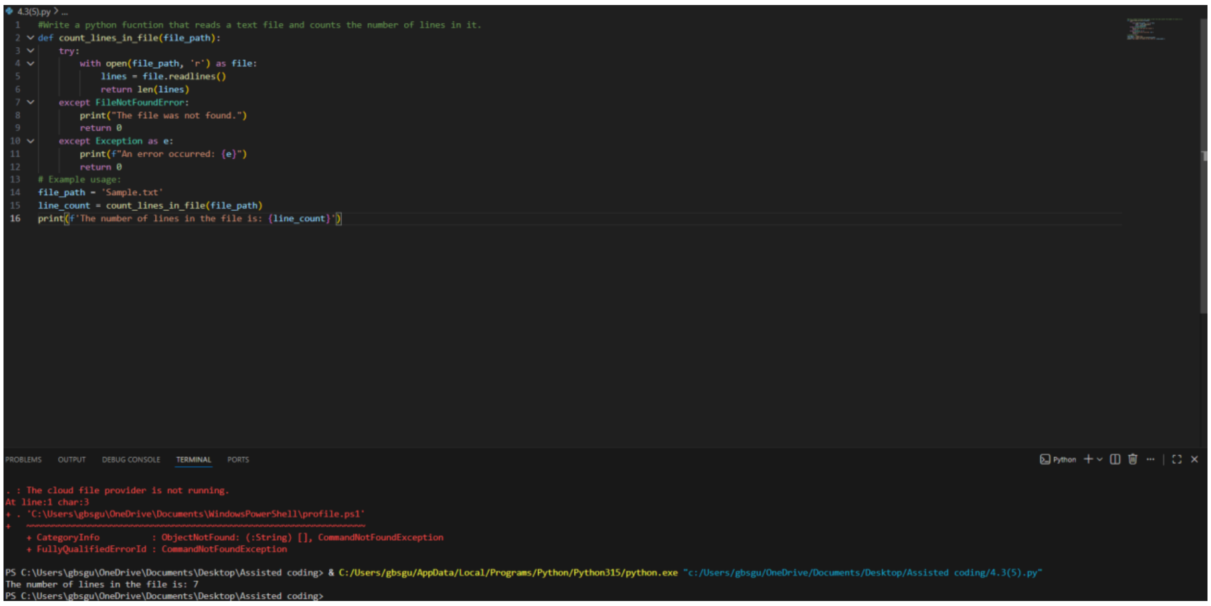
Scenario

File processing requires clear logical understanding.

Task Description

Use few-shot prompting to generate a Python function that:

- Reads a .txt file
- Counts the number of lines in the file
- Returns the line count



```python
#Write a python fucntion that reads a text file and counts the number of lines in it.
def count_lines_in_file(file_path):
    try:
        with open(file_path, 'r') as file:
            lines = file.readlines()
            return len(lines)
    except FileNotFoundError:
        print("The file was not found.")
        return 0
    except Exception as e:
        print(f"An error occurred: {e}")
        return 0
# Example usage:
file_path = 'Sample.txt'
line_count = count_lines_in_file(file_path)
print(f'The number of lines in the file is: {line_count}')
```

```
. : The cloud file provider is not running.
At line:1 char:3
+ . 'C:\Users\gbsgu\OneDrive\Documents\WindowsPowerShell\profile.ps1'
+   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding> & C:/Users/gbsgu/AppData/Local/Programs/Python/Python315/python.exe "c:/Users/gbsgu/OneDrive/Documents/Desktop/Assisted coding/4.3(5).py"
The number of lines in the file is: 7
PS C:\Users\gbsgu\OneDrive\Documents\Desktop\Assisted coding>
```

The AI understands the requirement through few-shot prompting by observing example input and output.
It opens the text file safely using with open() in read mode.
The file is read line by line, avoiding loading the entire file into memory.
Each line increments a counter to determine the total number of lines.
The final count is returned as the output.