



School of Computer Science and Artificial Intelligence

Lab Assignment-04

Course Title : AI Assistant Coding

Name of Student : Shaik Naved Ahmed

Enrollment No. : 2303A54053

Batch No. : 47-B

QUESTION 1: Zero-Shot Prompting – Leap Year Check

PROMPT:

Write a Python function that accepts a year as input and checks whether it is a **leap year**.

Constraints:

- Use correct leap year rules (divisible by 4, century years divisible by 400).
- Validate the input to ensure it is a positive integer year.
- Return or print an appropriate result indicating whether the year is a leap year.
- Keep the code simple and readable.

SCREENSHOT OF GENERATED CODE:

The screenshot shows a code editor interface with the following details:

- EXPLORER:** Shows a workspace with files: Lab-01.pages, Lab-01.pdf, Lab-02.docx, Lab-02.pages, Lab-02.pdf, lab-3.1.py, Lab-03.pages, Lab-03.pdf, Lab-04.pages, and lab-04.py.
- LAB-04.py Content:**

```
1  '''Write a Python function that accepts a year as input and checks whether it is a leap year.
2  Constraints:
3
4  Use correct leap year rules (divisible by 4, century years divisible by 400).
5
6  Validate the input to ensure it is a positive integer year.
7
8  Return or print an appropriate result indicating whether the year is a leap year.
9
10 Keep the code simple and readable.'''
11 def is_leap_year(year):
12     if not isinstance(year, int) or year <= 0:
13         print("Invalid input: Please enter a positive integer year.")
14         return
15     if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
16         print(f"{year} is a Leap Year")
17     else:
18         print(f"{year} is Not a Leap Year")
19
20 # Example usage:
21 is_leap_year(2020) # 2020 is a Leap Year
22 is_leap_year(1900) # 1900 is Not a Leap Year
23 is_leap_year(2000) # 2000 is a Leap Year
24 is_leap_year(-2020) # Invalid input: Please enter a positive integer year.
25
26 is_leap_year(2021) # 2021 is Not a Leap Year
```
- CHAT:** A sidebar with a "Build with Agent" button and a message: "Build with Agent" and "AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase."
- TERMINAL:** Shows the command: `(base) navedahmedshaik@Naveds-MacBook-Air AIAC %`
- STATUS:** Shows the file `lab-04.py` is open in the terminal tab.

INPUT & OUTPUT:

Input	Output	Reason
2020	2020 is a Leap Year	Divisible by 4 and not divisible by 100.
1900	1900 is Not a Leap Year	Divisible by 100 but not divisible by 400.
2000	2000 is a Leap Year	Divisible by 400, so it is a leap year.
-2020	Invalid input: Please enter a positive integer year.	Year must be a positive integer.

EXPLANATION:

The function first validates the input to ensure it is a positive integer; if not, it prints an error message and exits. It then applies the standard leap year rules: a year is a leap year if it is divisible by 4 but not divisible by 100, or if it is divisible by 400. Based on these conditions, the function prints whether the given year is a Leap Year or Not a Leap Year.

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

PROMPT:

Write a Python function that converts a given value in **centimeters** to **inches** using the correct formula.

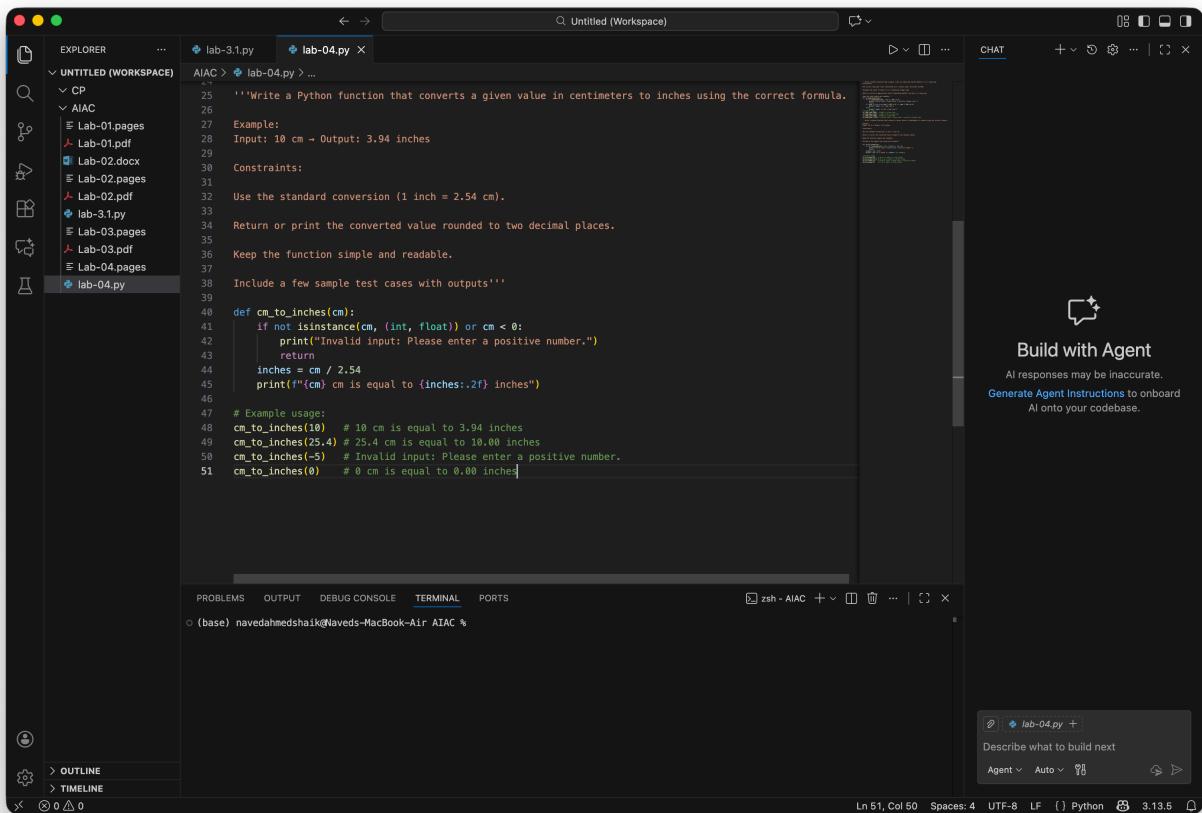
Example:

Input: 10 cm → Output: 3.94 inches

Constraints:

- Use the standard conversion (1 inch = 2.54 cm).
- Return or print the converted value rounded to two decimal places.
- Keep the function simple and readable.
- Include a few sample test cases with outputs

SCREENSHOT OF GENERATED CODE:



The screenshot shows a code editor interface with a dark theme. The left sidebar shows a file tree with various files and folders, including 'lab-01.pages', 'Lab-01.pdf', 'Lab-02.pages', 'Lab-02.pdf', 'Lab-03.pages', 'Lab-03.pdf', 'Lab-04.pages', and 'lab-04.py'. The main editor area contains the following Python code:

```
25  '''Write a Python function that converts a given value in centimeters to inches using the correct formula.
26
27 Example:
28 Input: 10 cm - Output: 3.94 inches
29
30 Constraints:
31
32 Use the standard conversion (1 inch = 2.54 cm).
33
34 Return or print the converted value rounded to two decimal places.
35
36 Keep the function simple and readable.
37
38 Include a few sample test cases with outputs'''
39
40 def cm_to_inches(cm):
41     if not isinstance(cm, (int, float)) or cm < 0:
42         print("Invalid input: Please enter a positive number.")
43         return
44     inches = cm / 2.54
45     print(f"{cm} cm is equal to {inches:.2f} inches")
46
47 # Example usage:
48 cm_to_inches(10) # 10 cm is equal to 3.94 inches
49 cm_to_inches(25.4) # 25.4 cm is equal to 10.00 inches
50 cm_to_inches(-5) # Invalid input: Please enter a positive number.
51 cm_to_inches(0) # 0 cm is equal to 0.00 inches
```

The right side of the interface features a 'Build with Agent' panel with a message: 'Build with Agent' and 'AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase.' Below this is a terminal window showing the command '(base) navedahmedshaik@Naveds-MacBook-Air AIAC %' and a status bar at the bottom with 'Ln 51, Col 50' and 'Spaces: 4'.

INPUT & OUTPUT:

Input	Output	Reason
10	10 cm is equal to 3.94 inches	Converted using $10 \div 2.54 \approx 3.94$.
25.4	25.4 cm is equal to 10.00 inches	25.4 cm equals exactly 10 inches.
-5	Invalid input: Please enter a positive number.	Negative values are not valid for length.
0	0 cm is equal to 0.00 inches	Zero is a valid input and converts to zero inches.

EXPLANATION:

The function converts a length from centimeters to inches. It first validates the input to ensure it is a non-negative number (integer or float). If the input is invalid or negative, it prints an error message and stops execution. For valid inputs, it converts centimeters to inches using the standard conversion factor (1 inch = 2.54 cm) and prints the result rounded to two decimal places.

Question 3: Few-Shot Prompting – Name Formatting

PROMPT:

Write a Python function that takes a full name as input and formats it as **Last, First**.

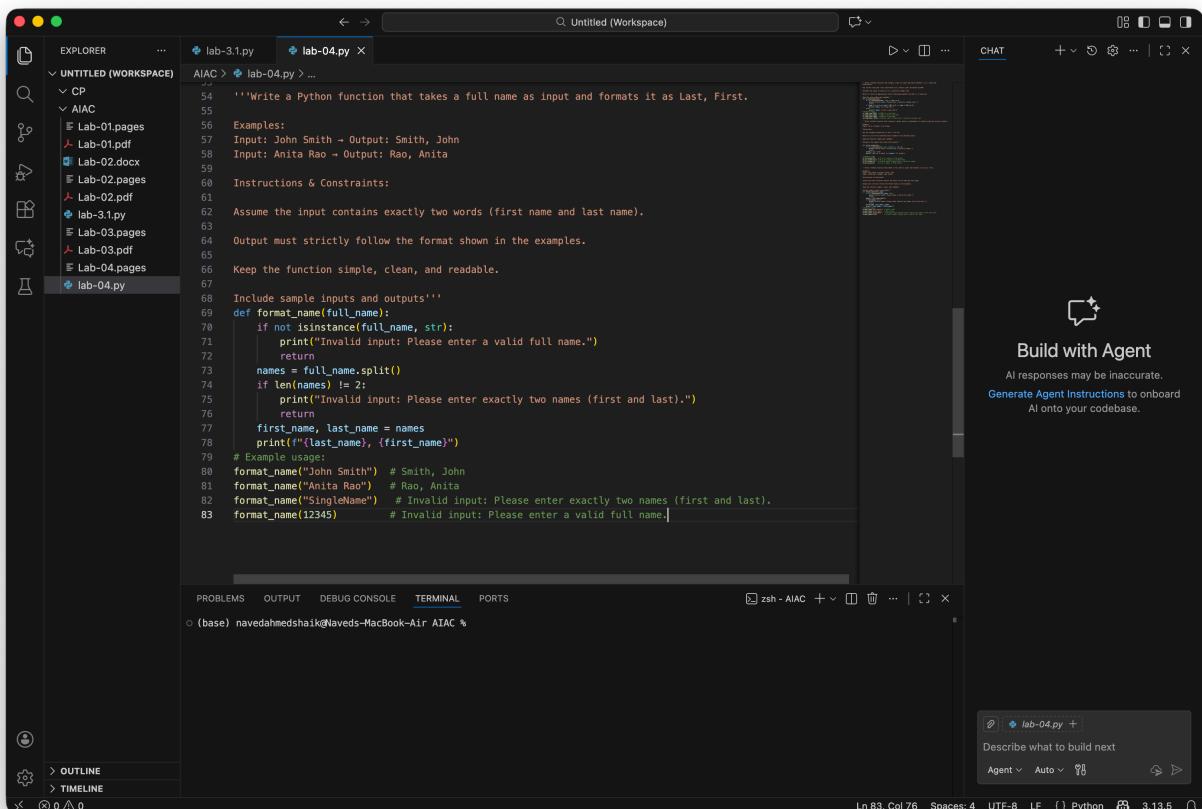
Examples:

Input: John Smith → Output: Smith, John
Input: Anita Rao → Output: Rao, Anita

Instructions & Constraints:

- Assume the input contains exactly two words (first name and last name).
- Output must strictly follow the format shown in the examples.
- Keep the function simple, clean, and readable.
- Include sample inputs and outputs

SCREENSHOT OF GENERATED CODE:



The screenshot shows a dark-themed VS Code interface. The Explorer sidebar on the left lists files like 'Lab-01.pages', 'Lab-02.pages', 'Lab-03.pages', and 'Lab-04.pages'. The 'lab-04.py' file is open in the main editor area. The code is as follows:

```
'''Write a Python function that takes a full name as input and formats it as Last, First.
Examples:
Input: John Smith -> Output: Smith, John
Input: Anita Rao -> Output: Rao, Anita
Instructions & Constraints:
Assume the input contains exactly two words (first name and last name).
Output must strictly follow the format shown in the examples.
Keep the function simple, clean, and readable.
Include sample inputs and outputs'''
def format_name(full_name):
    if not isinstance(full_name, str):
        print("Invalid input: Please enter a valid full name.")
        return
    names = full_name.split()
    if len(names) != 2:
        print("Invalid input: Please enter exactly two names (first and last).")
        return
    first_name, last_name = names
    print(f"{last_name}, {first_name}")
# Example usage:
format_name("John Smith") # Smith, John
format_name("Anita Rao") # Rao, Anita
format_name("SingleName") # Invalid input: Please enter exactly two names (first and last).
format_name(12345) # Invalid input: Please enter a valid full name.
```

The right-hand sidebar features an 'AI' icon and a 'Build with Agent' button. The status bar at the bottom shows '(base) navedahmedshaik@Naveds-MacBook-Air AIAC'.

INPUT & OUTPUT:

Input	Output	Reason
John Smith	Smith, John	Input contains exactly two names and is formatted correctly.
Anita Rao	Rao, Anita	Valid first and last name, order is reversed
SingleName	Invalid input: Please enter exactly two names (first and last).	Only one name is provided instead of two.
12345	Invalid input: Please enter a valid full	Input is not a string.

EXPLANATION:

The function checks whether the input is a valid string representing a full name. It then splits the string into parts based on spaces. If the input does not contain exactly two words (a first name and a last name), the function prints an error message. When the input is valid, it rearranges the names and prints them in the format “LastName, FirstName”

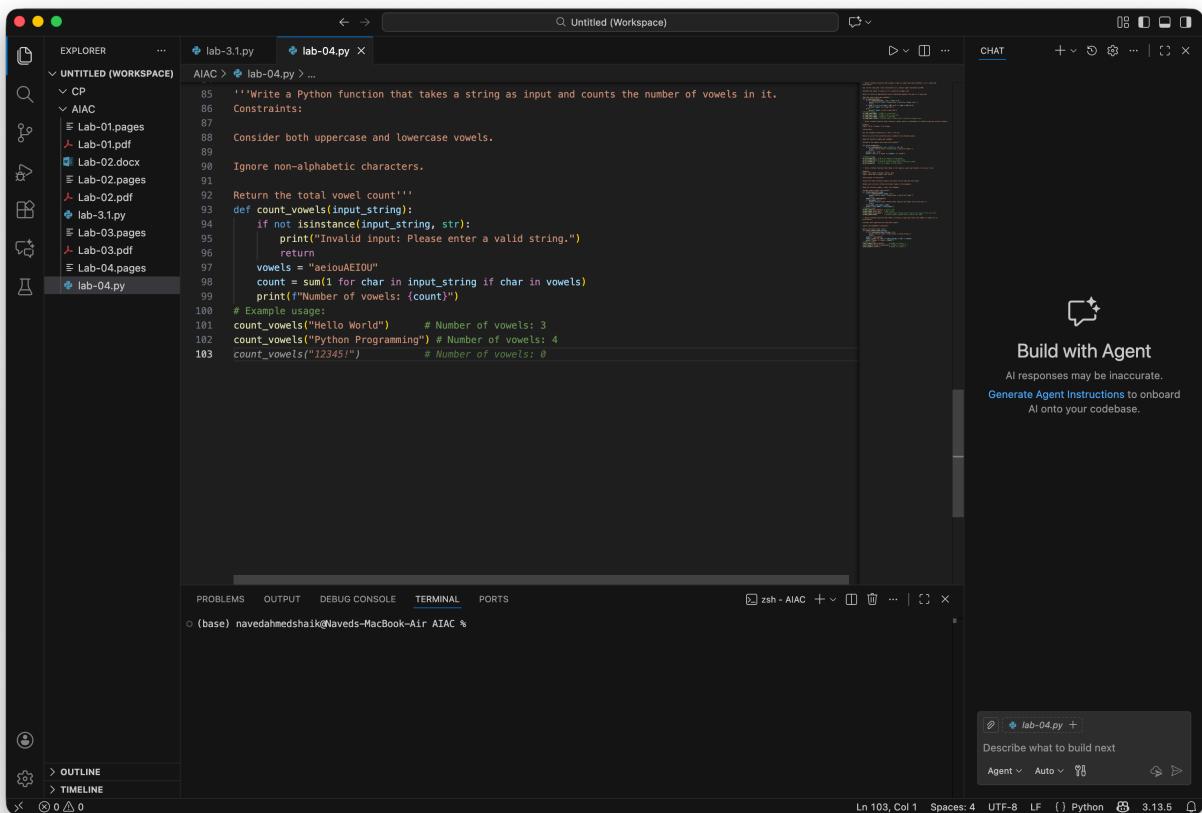
Question 4: Comparative Analysis – Zero-Shot vs Few-Shot

Zero-Shot PROMPT:

Write a Python function that takes a string as input and counts the number of vowels in it.
Constraints:

- Consider both uppercase and lowercase vowels.
- Ignore non-alphabetic characters.
- Return the total vowel count.

Zero-Shot SCREENSHOT OF GENERATED CODE:



The screenshot shows a code editor interface with a dark theme. The left sidebar shows a file tree with various files like 'Lab-01.pdf', 'Lab-02.docx', 'Lab-03.pages', and 'Lab-04.pages'. The main editor area contains the following Python code:

```
85 '''Write a Python function that takes a string as input and counts the number of vowels in it.
86 Constraints:
87
88 Consider both uppercase and lowercase vowels.
89
90 Ignore non-alphabetic characters.
91
92 Return the total vowel count'''
93 def count_vowels(input_string):
94     if not isinstance(input_string, str):
95         print("Invalid input: Please enter a valid string.")
96         return
97     vowels = "aeiouAEIOU"
98     count = sum(1 for char in input_string if char in vowels)
99     print("Number of vowels: {}".format(count))
100
101 count_vowels("Hello World")      # Number of vowels: 3
102 count_vowels("Python Programming") # Number of vowels: 4
103 count_vowels("12345!")          # Number of vowels: 0
```

The code editor has tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', and 'PORTS'. The terminal tab shows the command '(base) navedahmedshaik@Naveds-MacBook-Air AIAC %'. The status bar at the bottom right shows 'Ln 103, Col 1' and 'Python 3.13.5'.

Zero-Shot INPUT & OUTPUT:

Input	Output	Reason
"Hello World"	Number of vowels: 3	Vowels are e, o, o.
"Python Programming"	Number of vowels: 4	Vowels are o, o, a, i.

Zero-Shot EXPLANATION:

The function first checks whether the input is a valid string; if not, it prints an error message and stops execution. It defines a string containing all lowercase and uppercase vowels. Using a generator expression with the `sum()` function, it iterates through each character in the input string and counts how many characters are vowels. Finally, it prints the total number of vowels found in the string.

Few-Shot PROMPT:

Write a Python function that counts vowels in a string.

Examples:

Input: 'hello' → Output: 2

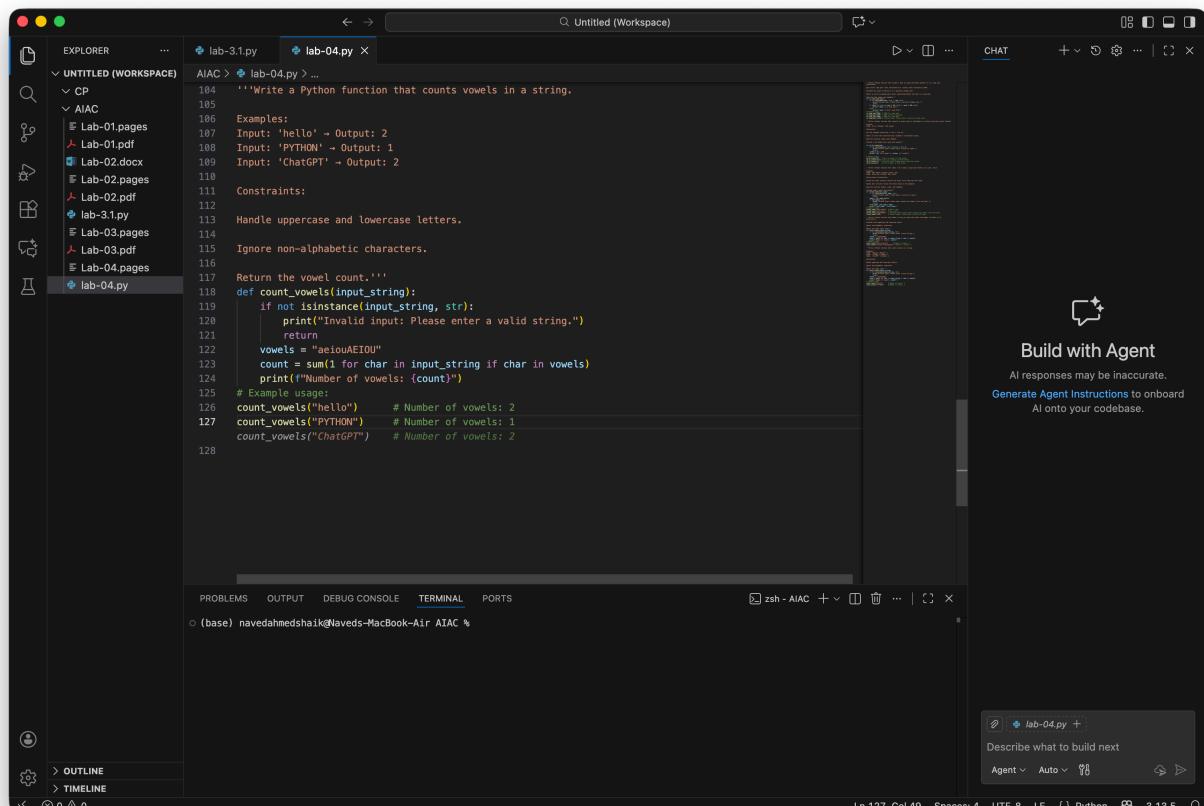
Input: 'PYTHON' → Output: 1

Input: 'ChatGPT' → Output: 2

Constraints:

- Handle uppercase and lowercase letters.
- Ignore non-alphabetic characters.
- Return the vowel count.

Few-Shot SCREENSHOT OF GENERATED CODE:



The screenshot shows a code editor interface with a dark theme. The left sidebar shows a file tree with various files and folders, including 'lab-3.1.py' and 'lab-4.py'. The main editor area contains the following Python code:

```
AIAC > lab-04.py > ...
104     '''Write a Python function that counts vowels in a string.
105
106     Examples:
107     Input: 'hello' → Output: 2
108     Input: 'PYTHON' → Output: 1
109     Input: 'ChatGPT' → Output: 2
110
111     Constraints:
112
113     Handle uppercase and lowercase letters.
114
115     Ignore non-alphabetic characters.
116
117     Return the vowel count.'''
118     def count_vowels(input_string):
119         if not isinstance(input_string, str):
120             print("Invalid input: Please enter a valid string.")
121             return
122
123         vowels = "aeiouAEIOU"
124         count = sum(1 for char in input_string if char in vowels)
125         print(f"Number of vowels: {count}")
126
127     # Example usage:
128     count_vowels("hello")      # Number of vowels: 2
     count_vowels("PYTHON")     # Number of vowels: 1
     count_vowels("ChatGPT")    # Number of vowels: 2
```

The code editor has a 'CHAT' tab on the right, which includes a 'Build with Agent' button and a message: 'Build with Agent AI responses may be inaccurate. Generate Agent Instructions to onboard AI onto your codebase.' Below the code editor is a terminal window showing the command: '(base) navedahmedshaik@Naveds-MacBook-Air AIAC %'. The bottom status bar shows: 'Ln 127, Col 49 Spaces: 4 UTF-8 LF () Python 3.13.5'.

Few-Shot INPUT & OUTPUT:

Input	Output	Reason
"hello"	Number of vowels: 2	Vowels e and o are present in the string.
"PYTHON"	Number of vowels: 1	Only O is a vowel; other letters are consonants.

Few-Shot EXPLANATION:

The function first verifies that the input is a string; if not, it prints an error message and exits. It then defines a list of vowels containing both lowercase and uppercase letters. Using a generator expression inside the sum() function, it iterates through each character in the string and counts how many of them are vowels. Finally, it prints the total number of vowels found in the given string

Zero-Shot vs Few-Shot Comparison:

Criteria	Zero-Shot	Few-Shot
Accuracy	Correct	Correct
Readability	Good	Better
Logical Clarity	Simple	Clearer due to examples
Handling Case Sensitivity	Explicit	Simplified using .lower()

Question 5: Few-Shot Prompting – File Handling

PROMPT:

Write a Python function that reads a .txt file and returns the number of lines.

Examples:

File content:

Hello

World

→ Output: 2

File content:

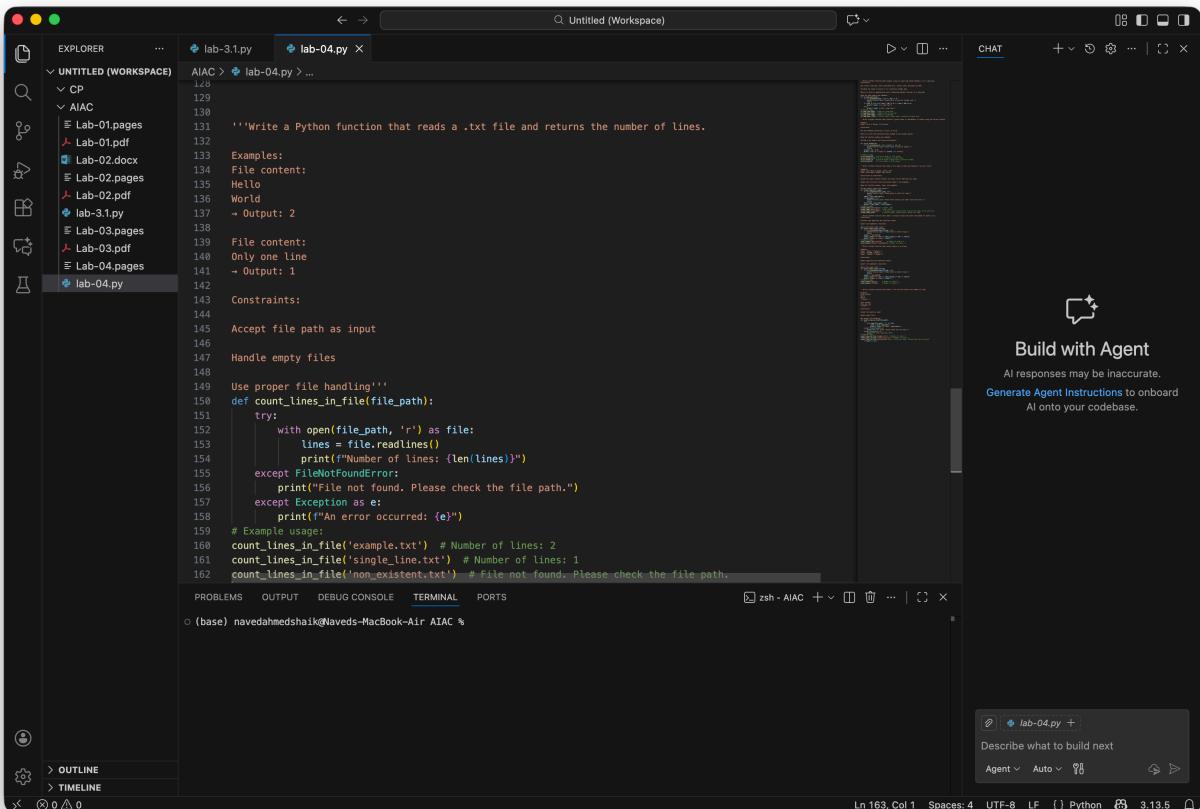
Only one line

→ Output: 1

Constraints:

- Accept file path as input
- Handle empty files
- Use proper file handling

SCREENSHOT OF GENERATED CODE:



```
128
129
130
131 """Write a Python function that reads a .txt file and returns the number of lines.
132
133 Examples:
134     File content:
135     Hello
136     World
137     -> Output: 2
138
139     File content:
140     Only one line
141     -> Output: 1
142
143 Constraints:
144
145     Accept file path as input
146
147     Handle empty files
148
149     Use proper file handling"""
150 def count_lines_in_file(file_path):
151     try:
152         with open(file_path, 'r') as file:
153             lines = file.readlines()
154             print(f"Number of lines: {len(lines)}")
155     except FileNotFoundError:
156         print(f"File not found. Please check the file path.")
157     except Exception as e:
158         print(f"An error occurred: {e}")
159
160 # Example usage:
161 count_lines_in_file('example.txt') # Number of lines: 2
162 count_lines_in_file('single_line.txt') # Number of lines: 1
163 count_lines_in_file('non_existent.txt') # File not found. Please check the file path.
```

INPUT & OUTPUT:

Input (File Path)	Output	Reason
example.txt	Number of lines: 2	File exists and contains two lines.
single_line.txt	Number of lines: 1	File exists with only one line.
non_existent.txt	File not found. Please check the file path.	File does not exist at the given path.

EXPLANATION:

The function takes a file path as input and attempts to open the file in read mode. If the file exists, it reads all the lines into a list and prints the total number of lines using the length of that list. If the specified file does not exist, the function catches a `FileNotFoundException` and prints a clear error message. Any other unexpected errors are also handled using a general exception block to prevent the program from crashing.