

LAB ASSIGNMENT: 4.3

NAME: K.SNEHA

ROLL NO: 2403A510F4

BATCH:06

BRANCH :CSE

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING			
ProgramName:B. Tech	Assignment Type: Lab	AcademicYear:2025-2026			
CourseCoordinatorName	Venkataramana Veeramsetty				
Instructor(s)Name	Dr. V. Venkataramana (Co-ordinator) Dr. T. Sampath Kumar Dr. Pramoda Patro Dr. Brij Kishor Tiwari Dr.J.Ravichander Dr. Mohammand Ali Shaik Dr. Anirodh Kumar Mr. S.Naresh Kumar Dr. RAJESH VELPULA Mr. Kundhan Kumar Ms. Ch.Rajitha Mr. M Prakash Mr. B.Raju Intern 1 (Dharma teja) Intern 2 (Sai Prasad) Intern 3 (Sowmya) NS_2 (Mounika)				
CourseCode	24CS002PC215	CourseTitle	AI Assisted Coding		
Year/Sem	II/I	Regulation	R24		
Date and Day of Assignment	Week2 - Wednesday	Time(s)			
Duration	2 Hours	Applicableto Batches			

AssignmentNumber:4.3(Present assignment number)/24(Total number of assignments)

Q.No.	Question	Expected Time to complete
1	<p>Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques</p> <p>Lab Objectives:</p> <ul style="list-style-type: none"> • To explore and apply different levels of prompt examples in AI-assisted code generation. • To understand how zero-shot, one-shot, and few-shot prompting affect AI output quality. • To evaluate the impact of context richness and example quantity on AI performance. • To build awareness of prompt strategy effectiveness for different problem types. <p>Lab Outcomes (LOs):</p> <p>After completing this lab, students will be able to:</p> <ul style="list-style-type: none"> • Use zero-shot prompting to instruct AI with minimal context. • Use one-shot prompting with a single example to guide AI code generation. • Apply few-shot prompting using multiple examples to improve AI responses. • Compare AI outputs across the three prompting strategies. <p>Task Description#1</p> <ul style="list-style-type: none"> • Zero-shot: Prompt AI to write a function that checks whether a given year is a leap year. <p>Expected Output#1</p> <ul style="list-style-type: none"> • AI-generated function with no examples provided <p>Task Description#2</p> <ul style="list-style-type: none"> • One-shot: Give one input-output example to guide AI in writing a function that converts centimeters to inches. <p>Expected Output#2</p> <ul style="list-style-type: none"> • Function with correct conversion logic <p>Task Description#3</p> <ul style="list-style-type: none"> • Few-shot: Provide 2–3 examples to generate a function that formats full names as “Last, First”. <p>Expected Output#3</p> <ul style="list-style-type: none"> • Well-structured function respecting the examples <p>Task Description#4</p> <ul style="list-style-type: none"> • Compare zero-shot and few-shot prompts for writing a function that counts the number of vowels in a string. <p>Expected Output#4</p> <ul style="list-style-type: none"> • Functional output and comparative reflection <p>Task Description#5</p> <ul style="list-style-type: none"> • Use few-shot prompting to generate a function that reads a .txt file and returns the number of lines. <p>Expected Output#5</p> <ul style="list-style-type: none"> • Working file-processing function with AI-guided logic 	Week2 - Wednesday

	<p>Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots</p> <p>Evaluation Criteria:</p> <table border="1"> <thead> <tr> <th>Criteria</th><th>Max Marks</th></tr> </thead> <tbody> <tr> <td>Zero Shot (Task #1)</td><td>0.5</td></tr> <tr> <td>One Shot (Task#2)</td><td>0.5</td></tr> <tr> <td>Few Shot (Task#3 & Task #5)</td><td>1.0</td></tr> <tr> <td>Comparison (Task#4)</td><td>0.5</td></tr> <tr> <td>Total</td><td>2.5 Marks</td></tr> </tbody> </table>	Criteria	Max Marks	Zero Shot (Task #1)	0.5	One Shot (Task#2)	0.5	Few Shot (Task#3 & Task #5)	1.0	Comparison (Task#4)	0.5	Total	2.5 Marks	
Criteria	Max Marks													
Zero Shot (Task #1)	0.5													
One Shot (Task#2)	0.5													
Few Shot (Task#3 & Task #5)	1.0													
Comparison (Task#4)	0.5													
Total	2.5 Marks													

VS code with github copilot

Task Description#1

- Zero-shot: Prompt AI to write a function that checks whether a given year is a leap year.

Expected Output#1

- AI-generated function with no examples provided

Prompt:

write code to check whether given year is leap or not.

The screenshot shows a code editor with a Python script named `task1copilot.py`. The script defines a function `is_leap_year` to check if a year is a leap year. It uses the modulo operator to determine if a year is divisible by 4, 100, or 400. The main part of the script prompts the user for a year, calls the function, and prints the result. A try-except block handles `ValueError` for non-integer inputs.

```
task1copilot.py > ...
1  # Function to check if a given year is a leap year
2  def is_leap_year(year):
3      """Return True if the given year is a leap year, else False."""
4      if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
5          return True
6      else:
7          return False
8
9  if __name__ == "__main__":
10     try:
11         year = int(input("Enter a year: "))
12         if is_leap_year(year):
13             print(f"{year} is a leap year.")
14         else:
15             print(f"{year} is not a leap year.")
16     except ValueError:
17         print("Invalid input. Please enter a valid year.")
18
```

The terminal window below shows the script being run in a PowerShell session. The user enters `2024`, and the script outputs `2024 is a leap year.`

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS + × ⋮ [ ] ×
PS C:\Users\thoop\lab4> & 'c:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\thoop\vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62745' '--' 'c:\Users\thoop\lab4\task1copilot.py'
Enter a year: 2024
2024 is a leap year.
PS C:\Users\thoop\lab4>
```

Task Description#2

- One-shot: Give one input-output example to guide AI in writing a function that converts centimeters to inches.

Expected Output#2

- Function with correct conversion logic

Prompt:

Write a function to convert centimeters to inches.

Example: 10.0 centimeters is equal to 3.912345 inches.

read input from the user.

```
task2copilot.py > ...
1  def cm_to_inches(cm):
2  |     return cm / 2.54
3
4  cm = float(input("Enter length in centimeters: "))
5  inches = cm_to_inches(cm)
6  print(f"{cm} centimeters is equal to {inches} inches.")

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

PS C:\Users\thoop\lab4> & 'c:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\thoop\.vs\code\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '62927' '--' 'c:\Users\thoop\lab4\task2copilot.py'
Enter length in centimeters: 100
100.0 centimeters is equal to 39.37007874015748 inches.
PS C:\Users\thoop\lab4>
```

Task Description#3

- Few-shot: Provide 2–3 examples to generate a function that formats full names as “Last, First”.

Expected Output#3

- Well-structured function respecting the examples

Prompt:

Write a python function that generates formats full names as “last,first”.

Example: peter parker =parker peter.

 Harry potter=Potter Harry

Read full name from the user.

Code:

```
task3copilot.py > ...
1
2     # Function to format full name as "Last First"
3     def format_full_name(full_name):
4         """Return the full name formatted as 'Last First' with proper capitalization."""
5         parts = full_name.strip().split()
6         if len(parts) < 2:
7             return "Invalid input. Please enter both first and last name."
8         first = parts[0].capitalize()
9         last = parts[-1].capitalize()
10        return f"{last} {first}"
11
12    if __name__ == "__main__":
13        full_name = input("Enter full name (first last): ")
14        formatted = format_full_name(full_name)
15        print(f"Formatted name: {formatted}")
16

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS
```

PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe c:/Users/thoop/lab4/task3copilot.py
Enter full name (first last): harichandana thoopukari
Formatted name: Thoopukari Harichandana
PS C:\Users\thoop\lab4> []

Task Description#4

- Compare zero-shot and few-shot prompts for writing a function that counts the number of vowels in a string.

Expected Output#4

- Functional output and comparative reflection

Prompt:

1.Zero shot:

Write a python program to count number of vowels in a string. read input from the user.

2.few-shot:

Write a python program to count number of vowels in a given string.

Example: sruniversity

Vowel count=4.

Read input from the user.

Zero shot

```
task4copilot.py > ...
1 def count_vowels(s):
2     vowels = 'aeiouAEIOU'
3     return sum(1 for char in s if char in vowels)
4
5 user_input = input("Enter a string: ")
6 vowel_count = count_vowels(user_input)
7 print(f"Number of vowels: {vowel_count}")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe c:/Users/thoop/lab4/task4copilot.py
Enter a string: sruniversity
Number of vowels: 4
PS C:\Users\thoop\lab4> 
```

few-shot

```
task4copilot.py > ...
1 # Program to count number of vowels in a given string
2
3 # Read input from the user
4 input_str = input("Enter a string: ")
5
6 # Define vowels
7 vowels = "aeiouAEIOU"
8
9 # Count vowels
10 vowel_count = sum(1 for char in input_str if char in vowels)
11
12 print(f"Vowels count-{vowel_count}-")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe c:/Users/thoop/lab4/task4copilot.py
Enter a string: srujanika
Number of vowels: 4
PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe c:/Users/thoop/lab4/task4copilot.py
Enter a string: harichandana
Vowel count-5-
PS C:\Users\thoop\lab4> 
```

Task Description#5

- Use few-shot prompting to generate a function that reads a .txt file and returns the number of lines.

Expected Output#5

- Working file-processing function with AI-guided logic

Prompt:

Generate a function that reads the given a.txt file and returns the number of lines

Example:hello sr university.hanmakonda warangal.

Top engineering college.

Offers wide variety of courses.

No.of lines in a.txt=3.

```
task5copilot.py > ...
1 def count_lines_in_file(filename):
2     with open(filename, 'r', encoding='utf-8') as f:
3         return len(f.readlines())
4
5 if __name__ == "__main__":
6     num_lines = count_lines_in_file('a.txt')
7     print(f"Number of lines in a.txt: {num_lines}")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

sk5copilot.py
Number of lines in a.txt: 8
Number of lines in a.txt: 8
PS C:\Users\thoop\lab4> & C:\Users\thoop\AppData\Local\Programs\Python\Python313\python.exe c:/Users/thoop/lab4/sk5copilot.py
Number of lines in a.txt: 8
Number of lines in a.txt: 8
PS C:\Users\thoop\lab4> 
```

Cursor AI

Task Description#1

- Zero-shot: Prompt AI to write a function that checks whether a given year is a leap year.

Expected Output#1

- AI-generated function with no examples provided

Prompt:

write code to check whether given year is leap or not.

```
1 def is_leap_year(year):| TAB to jump here
2     """
3         Check if a given year is a leap year.
4
5         Leap year rules:
6         1. If a year is divisible by 4, it's a leap year
7         2. However, if it's also divisible by 100, it's NOT a leap year
8         3. Unless it's also divisible by 400, then it IS a leap year
9
10        Args:
11            year (int): The year to check
12
13        Returns:
14            bool: True if it's a leap year, False otherwise
15        """
16
17        if year % 4 == 0:
18            if year % 100 == 0:
19                if year % 400 == 0:
20                    return True
21                else:
22                    return False
23            else:
24                return True
25        else:
26            return False
27
28    def main():
29        """Main function to demonstrate the leap year checker."""
30        print("Leap Year Checker")
31        print("=" * 20)
32
33        # Test cases
34        test_years = [2000, 2020, 2024, 1900, 2100, 2023, 2025]
35
36        for year in test_years:
37            if is_leap_year(year):
38                print(f"{year} is a leap year")
39            else:
40                print(f"{year} is NOT a leap year")
41
42        print("\n" + "=" * 20)
43
44        # Interactive input
45        try:
46            user_year = int(input("Enter a year to check: "))
47            if is_leap_year(user_year):
48                print(f"{user_year} is a leap year!")
49            else:
50                print(f"{user_year} is NOT a leap year.")
51        except ValueError:
52            print("please enter a valid integer year.")
```

Problems Output Debug Console Terminal Ports

PS C:\Users\thoop\lab4> & C:/Users/thoop/AppData/Local/Programs/Python/Python313/python.exe c:/Users/Enter a year: & c:/Users/thoop/AppData/Local/Programs/Python/Python313/python.exe c:/Users/thoop/lab4Invalid input. Please enter a valid year.
PS C:\Users\thoop\lab4> & C:/Users/thoop/AppData/Local/Programs/Python/Python313/python.exe c:/Users/Enter a year: 2024
2024 is a leap year.
PS C:\Users\thoop\lab4>

Task Description#2

- One-shot: Give one input-output example to guide AI in writing a function that converts centimeters to inches.

Expected Output#2

- Function with correct conversion logic

Prompt:

Write a function to convert centimeters to inches.

Example: 10.0 centimeters is equal to 3.912345 inches.

read input from the user.

```
task2cursor.py > main
 1  def cm_to_inches(cm):
 2
 3      """Convert centimeters to inches.
 4      Args:
 5          cm (float): Length in centimeters
 6      Returns:
 7          float: Length in inches
 8          """
 9      # 1 inch = 2.54 centimeters
10      inches = cm / 2.54
11      return inches
12
13  def main():
14      """Main function to demonstrate the centimeter to inch converter."""
15      print("Centimeter to Inch Converter")
16      print("-" * 30)
17      # Test case
18      test_cm = 10.0
19      test_inches = cm_to_inches(test_cm)
20      print(f"Example: {test_cm} centimeters is equal to {test_inches:.6f} inches.")
21      print("\n" + "=" * 30)
22      # Interactive input
23      try:
24          user_cm = float(input("Enter length in centimeters: "))
25          user_inches = cm_to_inches(user_cm)
26          print(f"{user_cm} centimeters is equal to {user_inches:.6f} inches.")
27      except ValueError:
28          print("Please enter a valid number.")
29
if __name__ == "__main__":
    main()
```

Problems Output Debug Console Terminal Ports Python ▲ + ×

```
Centimeter to Inch Converter
=====
Example: 10.0 centimeters is equal to 3.937008 inches.

=====
Enter length in centimeters: 100
Ctrl+K to generate a command
```

Task Description#3

- Few-shot: Provide 2–3 examples to generate a function that formats full names as “Last, First”.

Expected Output#3

- Well-structured function respecting the examples

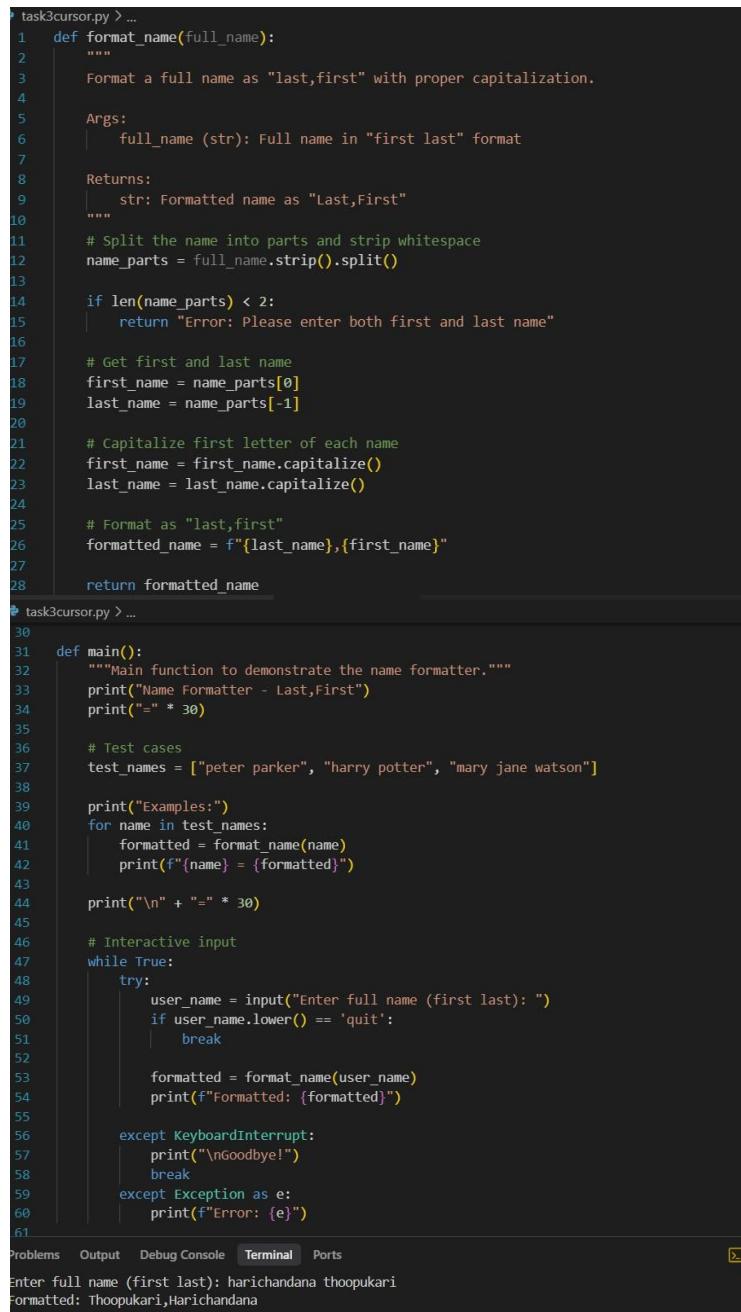
Prompt:

Write a python function that generates formats full names as “last,first”.

Example: peter parker =parker peter.

Harry potter=Potter Harry

Read full name from the user.



The screenshot shows a code editor with two tabs: 'task3cursor.py' and 'task3cursor.py'. The code is as follows:

```
# task3cursor.py > ...
1 def format_name(full_name):
2     """
3         Format a full name as "last,first" with proper capitalization.
4
5     Args:
6         full_name (str): Full name in "first last" format
7
8     Returns:
9         str: Formatted name as "Last,First"
10    """
11    # Split the name into parts and strip whitespace
12    name_parts = full_name.strip().split()
13
14    if len(name_parts) < 2:
15        return "Error: Please enter both first and last name"
16
17    # Get first and last name
18    first_name = name_parts[0]
19    last_name = name_parts[-1]
20
21    # Capitalize first letter of each name
22    first_name = first_name.capitalize()
23    last_name = last_name.capitalize()
24
25    # Format as "last,first"
26    formatted_name = f'{last_name},{first_name}'
27
28    return formatted_name
# task3cursor.py > ...
30
31 def main():
32     """Main function to demonstrate the name formatter."""
33     print("Name Formatter - Last,First")
34     print("=" * 30)
35
36     # Test cases
37     test_names = ["peter parker", "harry potter", "mary jane watson"]
38
39     print("Examples:")
40     for name in test_names:
41         formatted = format_name(name)
42         print(f'{name} = {formatted}')
43
44     print("\n" + "=" * 30)
45
46     # Interactive input
47     while True:
48         try:
49             user_name = input("Enter full name (first last): ")
50             if user_name.lower() == 'quit':
51                 break
52
53             formatted = format_name(user_name)
54             print(f"Formatted: {formatted}")
55
56         except KeyboardInterrupt:
57             print("\nGoodbye!")
58             break
59         except Exception as e:
60             print(f"Error: {e}")
61
```

The terminal tab shows the output of running the script:

```
Problems Output Debug Console Terminal Ports
Enter full name (first last): harichandana thoopukari
Formatted: Thoopukari,Harichandana
```

Task Description#4

- Compare zero-shot and few-shot prompts for writing a function that counts the number of vowels in a string.

Expected Output#4

- Functional output and comparative reflection

Prompt:

1. Zero shot:

Write a python program to count number of vowels in a string. read

input from the user.

2. few-shot:

Write a python program to count number of vowels in a given string.

Example: sruuniversity

Vowel count=4.

Read input from the user.

Zero-shot

```
task4cursor.py > ...
1 def format_name(full_name):
2     """
3         Format a full name as "last,first" with proper capitalization.
4
5     Args:
6         full_name (str): Full name in "first last" format
7
8     Returns:
9         str: Formatted name as "Last,First"
10    """
11    # Split the name into parts and strip whitespace
12    name_parts = full_name.strip().split()
13
14    if len(name_parts) < 2:
15        return "Error: Please enter both first and last name"
16
17    # Get first and last name
18    first_name = name_parts[0]
19    last_name = name_parts[-1]
20
21    # Capitalize first letter of each name
22    first_name = first_name.capitalize()
23    last_name = last_name.capitalize()
24
25    # Format as "last,first"
26    formatted_name = f"{last_name},{first_name}"
27
28    return formatted_name
...
def main():
    """Main function to get user input and count vowels."""
    print("Vowel Counter")
    print("=" * 20)

    # Get input from user
    user_input = input("Enter a string: ")

    # Count vowels
    vowel_count = count_vowels(user_input)

    # Display result
    print(f"Number of vowels in '{user_input}': {vowel_count}")

    # Show which vowels were found
    vowels_found = [char for char in user_input if char.lower() in 'aeiou']
    if vowels_found:
        print(f"Vowels found: {vowels_found}")
    else:
        print("No vowels found in the string.")

if __name__ == "__main__":
    main()

Ctrl+L to chat, Ctrl+K to generate
```

few-shot

```
task4_cursor.py > ...
1 def count_vowels(text):
2     """
3         Count the number of vowels in a given string.
4
5     Args:
6         text (str): Input string to count vowels in
7
8     Returns:
9         int: Number of vowels in the string
10    """
11    vowels = 'aeiouAEIOU'
12    count = 0
13
14    for char in text:
15        if char in vowels:
16            count += 1
17
18    return count
19
20
21 def main():
22     """Main function to demonstrate vowel counting with example and user input."""
23     print("Vowel Counter")
24     print("=" * 30)

task4_cursor.py > ⚡ main
21 def main():
22     # Show the example
23     example = "sruuniversity"
24     example_count = count_vowels(example)
25     print(f"Example: {example}")
26     print(f"Vowel count = {example_count}")
27
28     # Show the vowels found in the example
29     vowels_found = [char for char in example if char.lower() in 'aeiou']
30     print(f"Vowels found: {vowels_found}")
31     print("\n" + "=" * 30)
32
33     # Get input from user
34     user_input = input("Enter a string: ")
35
36     # Count vowels in user input
37     user_count = count_vowels(user_input)
38
39     # Display result
40     print(f"Vowel count = {user_count}")
41
42     # Show which vowels were found
43     user_vowels = [char for char in user_input if char.lower() in 'aeiou']
44     if user_vowels:
45         print(f"Vowels found: {user_vowels}")
46     else:
47         print("No vowels found in the string.")
48
49 if __name__ == "__main__":
50     main()

Problems Output Debug Console Terminal Ports
Vowel Counter
=====
Example: sruuniversity
Vowel count = 4
Vowels found: ['u', 'i', 'e', 'i']

=====
Enter a string: harichandana
Enter a string: harichandana
Vowel count = 5
```

Task Description#5

- Use few-shot prompting to generate a function that reads a .txt file and returns the number of lines.

Expected Output#5

- Working file-processing function with AI-guided logic

Prompt:

Generate a function that reads the given a.txt file and returns the number of lines

Example:hello sr university.hanmakonda warangal.

Top engineering college.

Offers wide variety of courses.

No.of lines in a.txt=3.

```
1 def count_lines_in_file(filename):
2     """
3         Read a text file and return the number of lines.
4
5     Args:
6         filename (str): Name of the file to read
7
8     Returns:
9         int: Number of lines in the file
10    """
11    try:
12        with open(filename, 'r', encoding='utf-8') as file:
13            lines = file.readlines()
14            return len(lines)
15    except FileNotFoundError:
16        print(f"Error: File '{filename}' not found.")
17        return -1
18    except Exception as e:
19        print(f"Error reading file: {e}")
20        return -1
21
22
23 def display_file_content(filename):
24     """
25         Display the content of a file with line numbers.
26
27     Args:
28         filename (str): Name of the file to display
29     """
30    try:
31        with open(filename, 'r', encoding='utf-8') as file:
32            lines = file.readlines()
33            print(f"Content of {filename}:")
34            print("-" * 40)
35            for i, line in enumerate(lines, 1):
36                print(f"Line {i}: {line.strip()}")
37            print("-" * 40)
38    except FileNotFoundError:
39        print(f"Error: File '{filename}' not found.")
40    except Exception as e:
41        print(f"Error reading file: {e}")
42
43 def main():
44     """Main function to demonstrate line counting with example and file reading."""
45     print("File Line Counter")
46     print("-" * 40)
47     # Show example
48     print("Example:")
49     example_lines = [
50         "Hello SR University.Hanmakonda Warangal.",
51         "Top Engineering College.",
52         "Offers wide variety of courses."
53     ]
54     print("Content:")
55     for i, line in enumerate(example_lines, 1):
56         print(f"Line {i}: {line}")
57
58     print(f"No. of lines in a.txt = {len(example_lines)}")
59     print("\n" + "=" * 40)
60
61     # Read actual a.txt file
62     filename = "a.txt"
63     line_count = count_lines_in_file(filename)
64
65     if line_count >= 0:
66         print(f"Reading file: {filename}")
67         display_file_content(filename)
68         print(f"No. of lines in {filename} = {line_count}.")
69     else:
70         print("Could not read the file.")
71
72
73 problems Output Debug Console Terminal Ports Python + ⌂ ⌂ ⌂
```

