|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE** | | | | | **DEPARTMENT OF COMPUTER SCIENCE ENGINEERING** | | | |
| **Program Name:** M.Tech. and MCA | | | | **Assignment Type: Lab** | | | **AcademicYear:**2025-2026 | |
| **Course Coordinator Name** | | | | Venkataramana Veeramsetty | | | | |
| **Course Code** | | |  | **Course Title** | | AI Assisted Problem Solving Using Python | | |
| **Year/Sem** | | | I/I | **Regulation** | | R24 | | |
| **Date and Day**  **of Assignment** | | | Week3 - Monday | **Time(s)** | |  | | |
| **Duration** | | | 2 Hours | **Applicable to**  **Batches** | | M.Tech. and MCA | | |
| **AssignmentNumber:4.3**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | |
|  | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***Expected Time***  ***to complete*** |  |
|  | 1 | Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques  **Lab Objectives:**   * 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.   **Lab Outcomes (LOs):**  After completing this lab, students will be able to:   * 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.   **Task Description#1**   * Zero-shot: Prompt AI to write a function that checks whether a given year is a leap year.   **Prompt:**  Write a Python function that checks whether a given year is a leap year  Code  def is\_leap\_year(year):      if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):          return True      else:          return False  # Example  print(is\_leap\_year(2024))  print(is\_leap\_year(1900))  print(is\_leap\_year(2000))  **Expected Output#1**   * AI-generated function with no examples provided     **Task Description#2**   * One-shot: Give one input-output example to guide AI in writing a function that converts centimeters to inches.   **Prompt**  “Example: 10 cm = 3.937 inches. Write a Python function to convert centimeters to inches.”  **Code**  def cm\_to\_inches(cm):      return cm / 2.54  # Example  print(cm\_to\_inches(10))  # Expected 3.937  **Expected Output#2**   * Function with correct conversion logic     **Task Description#3**  Few-shot: Provide 2–3 examples to generate a function that formats full names as “Last, First”.  **Prompt**  “Examples: Input: John Doe → Output: Doe, John Input: Mary Ann Smith → Output: Smith, Mary Ann Input: A. B. Clarke → Output: Clarke, A. B. Write a function that formats full names as ‘Last, First’.”  **Code**  def format\_name(full\_name):      parts = full\_name.split()      last = parts[-1]      first = " ".join(parts[:-1])      return f"{last}, {first}"  # Examples  print(format\_name("John Doe"))  print(format\_name("Mary Ann Smith"))  print(format\_name("A. B. Clarke"))  **Expected Output#3**   * Well-structured function respecting the examples     **Task Description#4**   * Compare zero-shot and few-shot prompts for writing a function that counts the number of vowels in a string.   **Prompt:**  “Write a Python function that counts vowels in a string.”  **Few-Shot Prompt:**  “Examples: Input: hello → Output: 2 Input: Python → Output: 1 Input: AI → Output: 2 Write a function that counts vowels in a string.”  code  def count\_vowels(s):      vowels = "aeiouAEIOU"      count = sum(1 for char in s if char in vowels)      return count  # Example test cases  print(count\_vowels("hello"))   # Output: 2  print(count\_vowels("Python"))  # Output: 1  print(count\_vowels("AI"))      # Output: 2  **Expected Output#4**   * Functional output and comparative reflection     **Task Description#5**   * Use few-shot prompting to generate a function that reads a .txt file and returns the number of lines.   **Prompt**  “Examples: File: sample.txt → contains 5 lines → Output: 5 File: empty.txt → contains 0 lines → Output: 0 File: notes.txt → contains 12 lines → Output: 12 Write a Python function that reads a .txt file and returns the number of lines.”  **code** def count\_lines\_in\_file():      filename = input("Enter the file name: ")      try:          with open(filename, 'r') as file:              lines = file.readlines()              print(f"Number of lines in '{filename}': {len(lines)}")      except FileNotFoundError:          print("File not found. Please check the file name and try again.")  # Run the function  count\_lines\_in\_file()  **Expected Output#5**   * Working file-processing function with AI-guided logic     **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**  **Evaluation Criteria:**   | **Criteria** | **Max Marks** | | --- | --- | | Zero Shot (Task #1) | 2.5 | | One Shot (Task#2) | 2.5 | | Few Shot (Task#3 & Task #5) | 2.5 | | Comparison (Task#4) | 2.5 | | **Total** | **10 Marks** | | | | | | | Week3 - Monday |  |