|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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.       **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.       **Expected Output#2**   * Function with correct conversion logic   **Logic of the Code: Centimeters to Inches Conversion**   * **The code defines a function to convert a length from centimeters to inches.** * **Conversion rule: 1 inch equals 2.54 centimeters.​** * **The function receives a value in centimeters as input.** * **It divides this value by 2.54 to get the equivalent length in inches.** * **The function returns the result (the length in inches).** * **You can test the function by providing sample centimeters values and printing the results for each test case.**   **Example Calculation:**   * **If you input 10 centimeters:  inches.**     **Task Description#3**   * Few-shot: Provide 2–3 examples to generate a function that formats full names as * “Last, First” * Prompt: Few-shot: Provide 2–3 examples to generate a function that formats full names as “Last, First”     The code takes a person's full name as a string input, which contains two words: the first name and the last name.   1. It splits the full name string into two parts based on the space between the first and last names. 2. The first word is stored as the first name, and the second word is stored as the last name. 3. The output is created by joining the last name first, then a comma and space, and then the first name. 4. Finally, it returns or prints the reformatted name string in the format "Last, First".   **Expected Output#3**   * Well-structured function respecting the examples     **Output :**    **Task Description#4**   * Compare zero-shot and few-shot prompts for writing a function that counts the number of vowels in a string. * **Zero-shot prompting means you give the AI a plain instruction without any examples. The AI generates code based only on its pre-trained knowledge. It tests the AI’s ability to solve the problem without extra guidance. This often results in simpler or less accurate outputs.** * **Few-shot prompting means you provide the AI with multiple examples (usually 2-5) of input-output pairs along with the instruction. This helps the AI learn the exact format, style, and logic expected. Few-shot usually leads to better, more accurate, and contextually appropriate results because the AI can generalize from the given examples.**     **Zero Shot :**  **Out put for Zero Shot:**  **Code for count of vowels with few shot of examples**      **Out put :**  **After comparing both zero and Few Shot Prompting the diff is in tabular form**   | **Aspect** | **Zero-Shot Prompting** | **Few-Shot Prompting** | | --- | --- | --- | | Context Provided | No example given; only instruction is provided | Multiple examples given with the instruction | | Dependence on Training | Fully dependent on pre-trained knowledge | Uses training + example context for better guidance | | Task Understanding | General understanding, can be ambiguous | More accurate understanding due to examples | | Output Quality | Basic, sometimes less accurate or incomplete | Usually more accurate, context-aware, and detailed | | Flexibility | Highly flexible for general tasks | Less flexible but more precise for given task type | | Complex Task Handling | Can struggle with complex or nuanced tasks | Better handles complex, nuanced, or structured tasks | | Prompt Length | Short prompt | Longer prompt due to examples included | | Best Use Case | Simple or well-known problems | Complex tasks requiring precision and examples |   **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 * **Solution (step-by-step logic used)** * **Validate input: ensure the provided filename is a string; otherwise raise TypeError.** * **Open the file in text read mode with UTF-8 encoding inside a with-statement to ensure it is closed automatically.** * **Read all lines using file.readlines() (or iterate and count) — this returns every line including empty/blank lines.** * **Count lines using len() on the list of lines and return that integer (0 for empty files).** * **Catch FileNotFoundError and IOError to provide user-friendly messages or propagate them.** * **In main: prompt the user with input(), strip whitespace, call the counting function, and print the result. Handle exceptions and show helpful messages**   **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 |  |