2303A53011(B46)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE** | | | | | **DEPARTMENT OF COMPUTER SCIENCE ENGINEERING** | | | | |
| **Program Name:** B. Tech | | | | **Assignment Type: Lab** | | | **Academic Year:**2025-2026 | | |
| **Course Coordinator Name** | | | | Dr. Rishabh Mittal | | | | | |
| **Instructor(s) Name** | | | | |  | | --- | | Mr. S Naresh Kumar | | Ms. B. Swathi | | Dr. Sasanko Shekhar Gantayat | | Mr. Md Sallauddin | | Dr. Mathivanan | | Mr. Y Srikanth | | Ms. N Shilpa | | Dr. Rishabh Mittal (Coordinator) | | Dr. R. Prashant Kumar | | Mr. Ankushavali MD | | Mr. B Viswanath | | Ms. Sujitha Reddy | | Ms. A. Anitha | | Ms. M.Madhuri | | Ms. Katherashala Swetha | | Ms. Velpula sumalatha | | Mr. Bingi Raju | | | | | | |
| **CourseCode** | | | 23CS002PC304 | **Course Title** | | AI Assisted Coding | | | |
| **Year/Sem** | | | III/II | **Regulation** | | R23 | | | |
| **Date and Day**  **of Assignment** | | | **Week2 – Monday** | **Time(s)** | | 23CSBTB01 To 23CSBTB52 | | | |
| **Duration** | | | 2 Hours | **Applicable to**  **Batches** | | All batches | | | |
| **Assignment Number:3.1**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***Expected Time***  ***to complete*** |  |
|  | 1 | **Lab Experiment: Prompt Engineering – Improving Prompts and Context Management**  **Lab Objectives**   1. To understand and apply different prompt engineering techniques for generating Python programs using AI-assisted tools. 2. To analyze the impact of context and examples on the accuracy and efficiency of AI-generated code. 3. To develop and refine real-world Python applications through iterative prompt improvement.   **Lab Outcomes**   1. Students will be able to design effective prompts to generate correct and optimized Python code. 2. Students will be able to compare and evaluate AI-generated solutions produced using different prompting strategies. 3. Students will be able to implement and document real-world Python applications using AI-assisted coding tools.   **Experiment – Prompt Engineering Techniques**  **Task Description**  Design and refine prompts using different prompting strategies to generate Python programs for basic computational problems.  **Question 1: Zero-Shot Prompting (Palindrome Number Program)**  Write a **zero-shot prompt** (without providing any examples) to generate a Python function that checks whether a given number is a palindrome.  **Task:**   * Record the AI-generated code. * Test the code with multiple inputs. * Identify any logical errors or missing edge-case handling.   **Zero-Shot Prompt:**  Write a Python function that takes an integer as input and returns True if the number is a palindrome and False otherwise. The function should not convert the number to a string and should handle edge cases correctly.  Code:    Output:      **Question 2: One-Shot Prompting (Factorial Calculation)**  Write a **one-shot prompt** by providing one input-output example and ask the AI to generate a Python function to compute the factorial of a given number.  **Example:** Input: 5 → Output: 120  **Task:**   * Compare the generated code with a zero-shot solution. * Examine improvements in clarity and correctness.       **Comparison: Zero-Shot vs One-Shot**   | **Aspect** | **Zero-Shot** | **One-Shot** | | --- | --- | --- | | Correctness | Partial | Robust | | Handles negative input | No | Yes | | Handles invalid types | No | Yes | | Risk of runtime errors | High | Low | | Readability | Basic | Clear & documented | | Scalability | Poor (recursion) | Better (iterative) |   **Question 3: Few-Shot Prompting (Armstrong Number Check)**  Write a **few-shot prompt** by providing multiple input-output examples to guide the AI in generating a Python function to check whether a given number is an Armstrong number.  **Examples:**   * Input: 153 → Output: Armstrong Number * Input: 370 → Output: Armstrong Number * Input: 123 → Output: Not an Armstrong Number   **Task:**   * Analyze how multiple examples influence code structure and accuracy. * Test the function with boundary values and invalid inputs.       ***(Optional Extension)***  **Question 4: Context-Managed Prompting (Optimized Number Classification)**  Design a **context-managed prompt** with clear instructions and constraints to generate an optimized Python program that classifies a number as **prime, composite, or neither**.  **Task:**   * Ensure proper input validation. * Optimize the logic for efficiency. * Compare the output with earlier prompting strategies.   **Context-Managed Prompt:**  You are a Python developer writing production-quality code.  Task: Write an **optimized Python program** that classifies a given number as:   * **Prime** * **Composite** * **Neither prime nor composite**   Constraints:   * Accept **only integers** as valid input * Handle **negative numbers, 0, and 1 correctly** * Optimize for performance (avoid unnecessary iterations) * Return clear, human-readable output * Code must be readable, efficient, and safe for large inputs       **Question 5: Zero-Shot Prompting (Perfect Number Check)**  Write a zero-shot prompt (without providing any examples) to generate a Python function that checks whether a given number is a perfect number.  Task:   * Record the AI-generated code. * Test the program with multiple inputs. * Identify any missing conditions or inefficiencies in the logic.     **Question 6: Few-Shot Prompting (Even or Odd Classification with Validation)**  Write a few-shot prompt by providing multiple input-output examples to guide the AI in generating a Python program that determines whether a given number is even or odd, including proper input validation.  Examples:   * Input: 8 → Output: Even * Input: 15 → Output: Odd * Input: 0 → Output: Even   Task:   * Analyze how examples improve input handling and output clarity. * Test the program with negative numbers and non-integer inputs   . | | | | | | Week2 - Monday |  |