Lab Assignment-3.2

# **Batch -04**

# **AI ASSISTED CODING**

NAME:B.RAMCHARAN

NO.:-2403A510D2

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | | | | 1. Dr. Mohammed Ali Shaik  2. Dr. T Sampath Kumar  3. Mr. S Naresh Kumar  4. Dr. V. Rajesh  5. Dr. Brij Kishore  6. Dr Pramoda Patro  7. Dr. Venkataramana  8. Dr. Ravi Chander  9. Dr. Jagjeeth Singh | | | | | |
| CourseCode | | | 24CS002PC215 | CourseTitle | | AI Assisted Coding | | | |
| Year/Sem | | | II/I | Regulation | | R24 | | | |
| Date and Day  of Assignment | | | Week2-Tuesday | Time(s) | |  | | | |
| Duration | | | 2 Hours | Applicableto  Batches | | 24CSBTB01 To 24CSBTB39 | | | |
| AssignmentNumber:3.2(Present assignment number)/24(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | Q.No. | Question | | | | | | *ExpectedTime*  *to complete* |  |
| e | 1 | Lab 3: Prompt Engineering – Improving Prompts and Context Management  Lab Objectives:  To understand how prompt structure and wording influence AI-generated code.  To explore how context (like comments and function names) helps AI generate relevant output.  To evaluate the quality and accuracy of code based on prompt clarity.  To develop effective prompting strategies for AI-assisted programming.  Lab Outcomes (LOs):  After completing this lab, students will be able to:  Generate Python code using Google Gemini in Google Colab.  Analyze the effectiveness of code explanations and suggestions by Gemini.  Set up and use Cursor AI for AI-powered coding assistance.  Evaluate and refactor code using Cursor AI features.  Compare AI tool behavior and code quality across different platforms.  Task Description#1  Ask AI to write a function to calculate compound interest, starting with only the function name. Then add a docstring, then input-output example  Expected Output#1  Comparison of AI-generated code styles  OBSERVATION:-  #TASK-1:-  \*Using google colab Gemini AI we get our output.\*  sub task1:-  Create function name    Sub task-2:-  Add docstring    Sub task-3:-  Add Example      **TASK 1 COMPLETED**  Task Description#2  Do math stuff, then refine it to: # Write a function to calculate average, median, and mode of a list of numbers.  Expected Output#2  AI-generated function evolves from unclear to accurate multi-statistical operation.  OBSERVATION:-  GOOGLE COLAB GEMINI AI CODE:-    from collections  EXPLANATION:-  Overall, the code demonstrates good practices like handling potential issues (empty list), using appropriate data structures (Counter), and clearly separating the logic for calculating each statistic. It's a good foundation for understanding how to perform basic statistical calculations in Python.  **TASK COMPLETED**  Task Description#3  provide multiple examples of input-output to the AI for convert\_to\_binary(num) function. Observe how AI uses few-shot prompting to generalize.  Expected Output#3  Enhanced AI output with clearer prompts  OBSERVATION:-  USING GEMINI AI,  #CODE:-    def convert\_to\_binary(num):  """Converts a decimal number to its binary representation as a string.  Args:  num: An integer decimal number.  Returns:  A string representing the binary form of the input number.  """  if num == 0:  return "0"  binary = ""  while num > 0:  binary = str(num % 2) + binary  num //= 2  return binary  EXPLANATION:-  observation about the convert\_to\_binary function is how it repeatedly uses the simple idea of dividing by two and looking at the remainder. It's like peeling away the binary digits one by one from right to left. The loop keeps track of the remaining value, and with each step, it figures out the next binary digit by checking if the number is even (remainder 0) or odd (remainder 1) before moving on to the next division. This iterative process, where the result of each step feeds into the next, is a fundamental concept in many programming problems.  **TASK COMPLETED**  Task Description#4  Create an user interface for an hotel to generate bill based on customer requirements  Expected Output#4  Consistent functions with shared logic  EXPLANATION:-  CODE GENERATED:-        #OUTPUT:-    EXPLANATION:-   building this hotel bill generator using ipywidgets is a cool way to see how you can make code interactive right inside the notebook. Instead of just running code and seeing output, you get to create sliders, dropdowns, and checkboxes that you can actually click and type into. It makes the code feel more like a real application, and you can instantly see how the inputs you choose affect the final bill when you hit the "Generate Bill" button. It shows how Python isn't just for calculations, but can also be used to build simple visual tools.    **TASK COMPLETED**  Task Description#5  Analyzing Prompt Specificity: Improving Temperature Conversion Function with Clear Instructions  Expected Output#5  Code quality difference analysis for various prompts  #EXPLANATION:-    OBSERVATION:-  The initial function convert\_temperature only supported converting Celsius to Fahrenheit.  The refined convert\_temperature function now supports conversions between Celsius, Fahrenheit, and Kelvin.Input validation was successfully implemented to check for numeric temperature values and supported units('C','F','K').Errorhandlinusing ValueErrorexceptions was added for invalid inputs.  The function includes a comprehensive docstring with examples demonstrating valid conversions and errorhandling.Testing confirmed the function correctly handles valid conversions between the three units, edge cases like freezing and boiling points, and raisesappropriate ValueError exceptions for invalid inputs.  Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and  Evaluation Criteria:   | Criteria | Max Marks | | --- | --- | | Task#1 | 0.5 | | Task#2 | 0.5 | | Task #3 | 0.5 | | Task #4 | 0.5 | | Task #5 | 0.5 | | Total | 2.5 Marks | | | | | | | 03.08.2025 EOD |  |

**\*\*\*TASKS COMPLETED\*\*\***