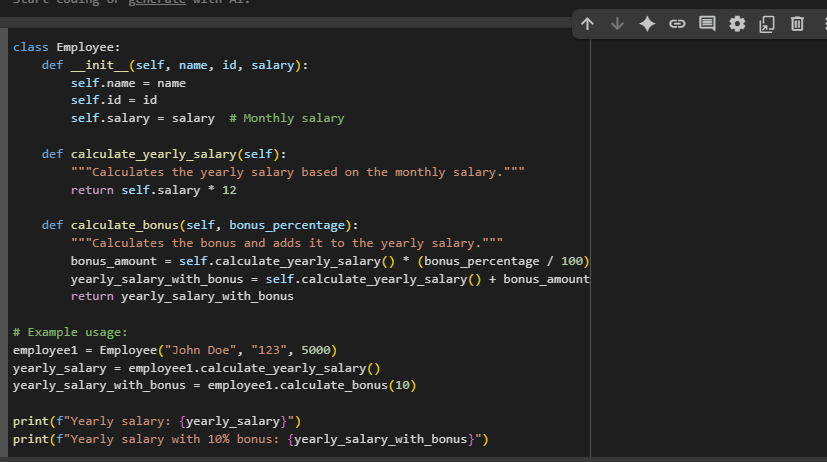
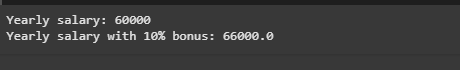
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE** | | | | | **DEPARTMENT OF COMPUTER SCIENCE ENGINEERING** | | | | |
| **Program Name:** B. Tech | | | | **Assignment Type: Lab** | | | **Academic Year:**2025-2026 | | |
| **Course Coordinator Name** | | | | 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) | | | | | | |
| **Course Code** | | | 24CS002PC215 | **Course Title** | | AI Assisted Coding | | | |
| **Year/Sem** | | | II/III | **Regulation** | | R24 | | | |
| **Date and Day**  **of Assignment** | | | Week3 – Monday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicable to**  **Batches** | |  | | | |
| **Assignment Number:6.1**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***Expected Time***  ***to complete*** |  |
|  | 1 | Lab 6: AI-Based Code Completion – Classes, Loops, and Conditionals  **Lab Objectives:**   * To explore AI-powered auto-completion features for core Python constructs. * To analyze how AI suggests logic for class definitions, loops, and conditionals. * To evaluate the completeness and correctness of code generated by AI assistants.   **Lab Outcomes (LOs):**  After completing this lab, students will be able to:   * Use AI tools to generate and complete class definitions and methods. * Understand and assess AI-suggested loops for iterative tasks. * Generate conditional statements through prompt-driven suggestions. * Critically evaluate AI-assisted code for correctness and clarity.   Task Description #1 (Classes – Employee Management)   * Task: Use AI to create an Employee class with attributes (name, id, salary) and a method to calculate yearly salary. * Instructions:   + Prompt AI to generate the Employee class.   + Analyze the generated code for correctness and structure.   + Ask AI to add a method to give a bonus and recalculate salary.   Expected Output #1:   * A class with constructor, display\_details(), and calculate\_bonus() methods.   Task Description #2 (Loops – Automorphic Numbers in a Range)   * Task: Prompt AI to generate a function that displays all Automorphic numbers between 1 and 1000 using a for loop. * Instructions:   + Get AI-generated code to list Automorphic numbers using a for loop.   + Analyze the correctness and efficiency of the generated logic.   + Ask AI to regenerate using a while loop and compare both implementations.   Expected Output #2:   * Correct implementation that lists Automorphic numbers using both loop types, with explanation.   Task Description #3 (Conditional Statements – Online Shopping Feedback Classification)   * Task: Ask AI to write nested if-elif-else conditions to classify online shopping feedback as Positive, Neutral, or Negative based on a numerical rating (1–5). * Instructions:   + Generate initial code using nested if-elif-else.   + Analyze correctness and readability.   + Ask AI to rewrite using dictionary-based or match-case structure.   Expected Output #3:   * Feedback classification function with explanation and an alternative approach.   Task Description #4 (Loops – Prime Numbers in a Range)   * Task: Generate a function using AI that displays all prime numbers within a user-specified range (e.g., 1 to 500). * Instructions:   + Get AI-generated code to list all primes using a for loop.   + Analyze the correctness and efficiency of the prime-checking logic.   + Ask AI to regenerate an optimized version (e.g., using the square root method).   Expected Output #4:   * Python program that lists all prime numbers within a given range, with an optimized version and explanation.   Task Description #5 (Classes – Library System)   * Task: Use AI to build a Library class with methods to add\_book(), issue\_book(), and display\_books(). * Instructions:   + Generate Library class code using AI.   + Analyze if methods handle edge cases (e.g., issuing unavailable books).   + Ask AI to add comments and documentation.   Expected Output #5:   * Library class with all methods, inline comments, and explanation.   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** | | --- | --- | | Class | 1.0 | | Loops | 1.0 | | Conditional Statements | 0.5 | | **Total** | **2.5 Marks** | | | | | | | Week3 - Monday |  |

**ROLLNO:2403A51309 DATE:08-09-2025**

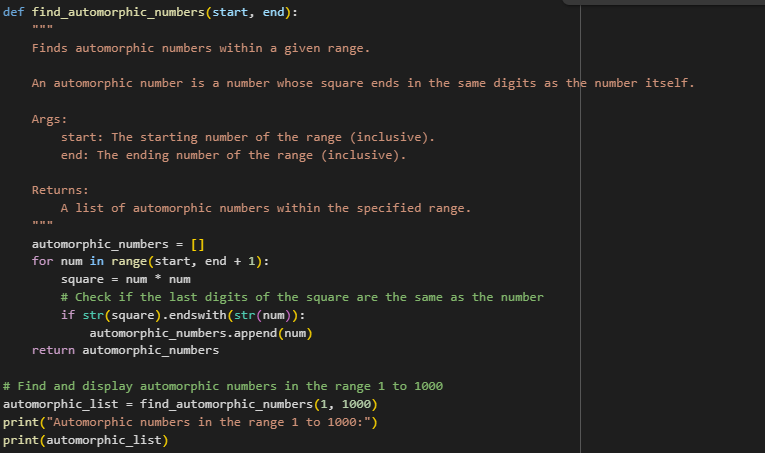
**#TASK1  
PROMPT**: write a python class employee management having attributes name,id,salary and add a method that calculate yearly salary ,other method bonus that calculate bonus and recalculates salary ,display yearly salary and yearly salary after bonus.   
**CODE:**

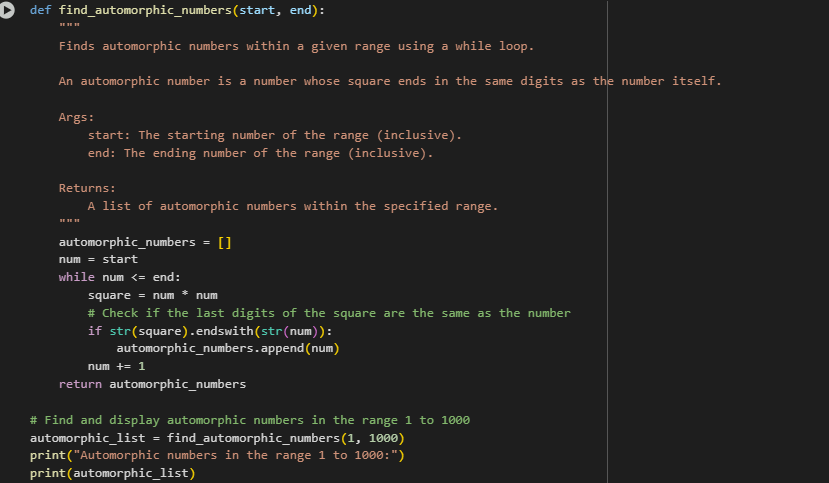
**  
OUTPUT:  
**

**OBSERVATION**: The code defines an Employee class with attributes: name, id, and salary.

1. calculate\_yearly\_salary() – computes yearly salary from the monthly salary.
2. calculate\_bonus(bonus\_percentage) – calculates and adds a bonus percentage to the yearly salary.

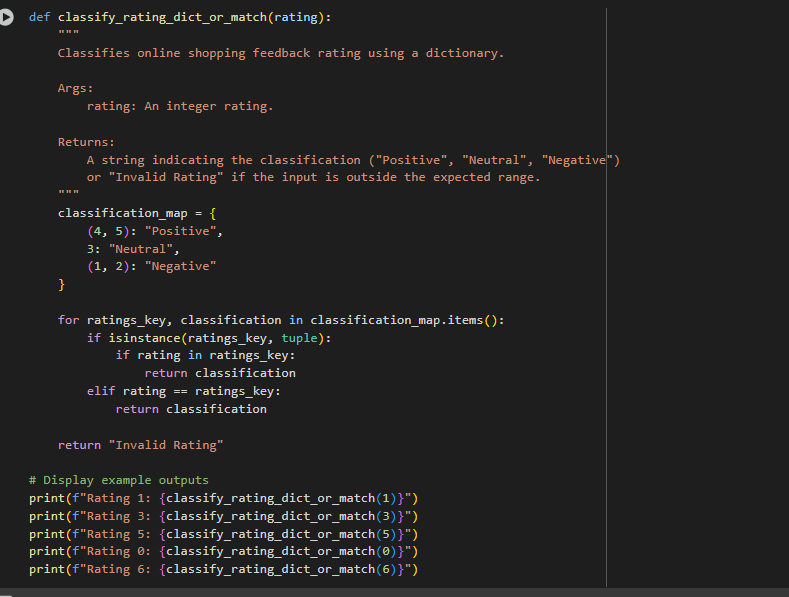
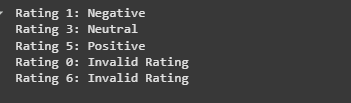
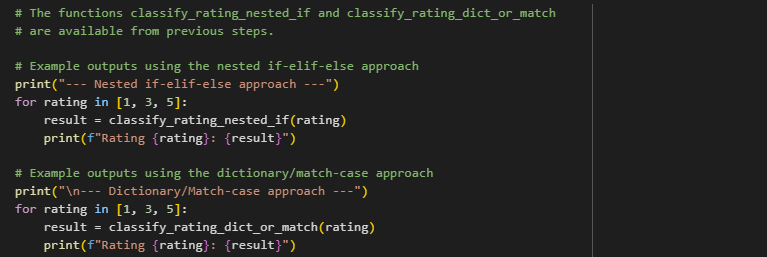
**#TASK2:  
PROMPT:** WRITE A PYTHON FUNCTION THAT CALCULATES AUTOMORPHIC NUMBERS IN RANGE 1 TO 1000 USING FOR LOOP ,DISPLAY THE LIST OF AUTOMORPHIC NUMBERS IN RANGE.

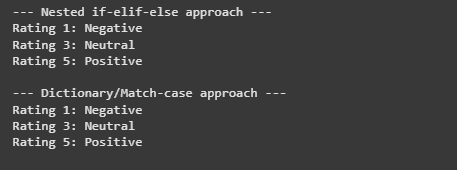
**CODE:(USING FOR LOOP)** **OUT PUT:**

**CODE:(USING WHILE LOOP)** **OUTPUT:**

**OBSERVATION:** The program correctly identifies automorphic numbers in the range 1 to 1000 using both for loop and while loop.  
The result [1, 5, 6, 25, 76, 376, 625] is consistent in both methods, proving the logic works for different looping constructs.

**#TASK3:  
PROMPT:** You are tasked with writing a Python program to classify online shopping feedback based on a numerical rating (1–5).

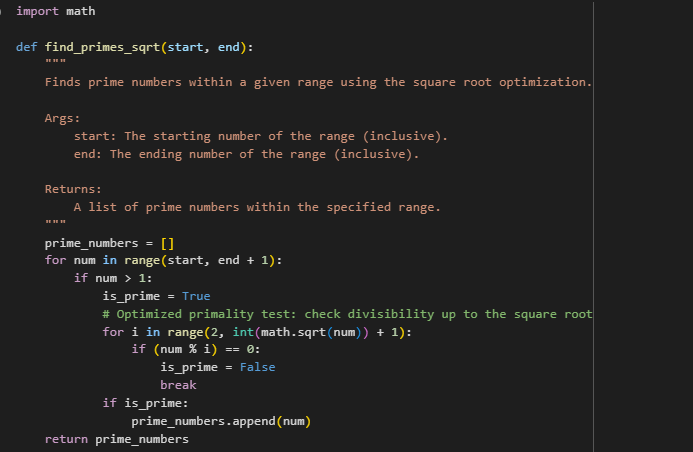
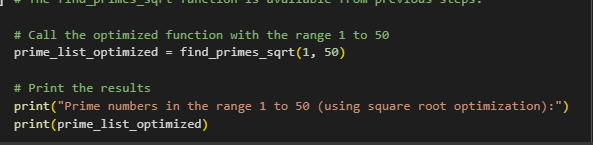
1. First, write code using nested if-elif-else conditions to classify ratings:
   * 4 or 5 → "Positive"
   * 3 → "Neutral"
   * 1 or 2 → "Negative"
2. Analyze your code for correctness and readability.
3. Rewrite the solution using either a dictionary-based approach or a match-case structure (Python 3.10+).
4. Display example outputs for different inputs (e.g., 1, 3, 5).  
   **CODE:(USING DICTIONARY)** **OUT PUT:(NESTED IF-ELSE/MATCH-CASE APPROCH..)** **CODE:**

**OUTPUT:**   
**OBSERVATION:** The program correctly classifies online shopping feedback using nested if-elif-else conditions.  
It was later rewritten using dictionary/match-case for improved readability and maintainability.  
Both approaches give accurate results, demonstrating different conditional constructs in Python.

**#TASK4:  
PROMPT:** Write a Python function that displays all prime numbers within a user-specified range (for example, 1 to 500).

1. First, generate code using a for loop to check and list prime numbers in the range.
2. Analyze the correctness and efficiency of the prime-checking logic.
3. Then, regenerate an optimized version of the function using the square root method for prime checking.
4. Show example outputs for a range such as 1 to 50.

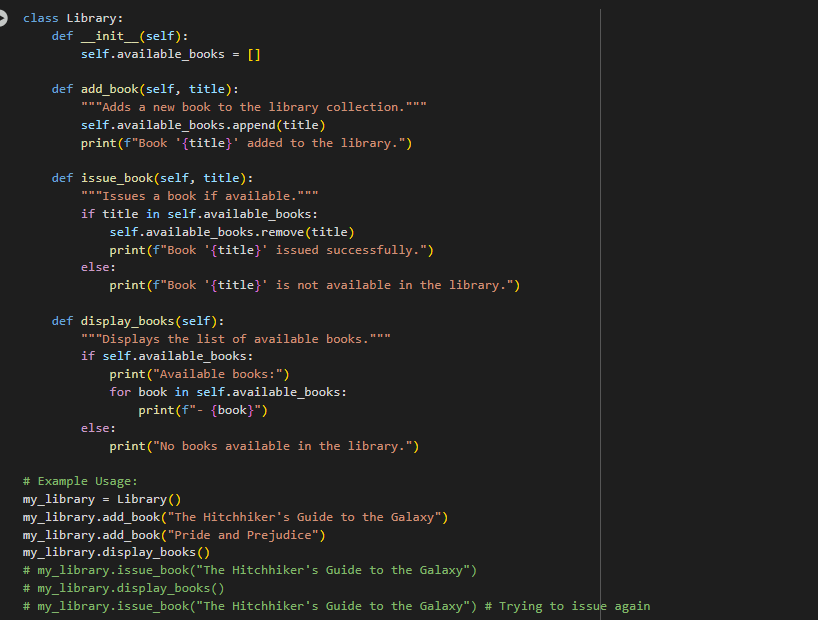
**CODE:**

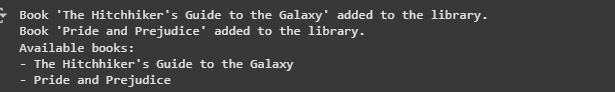
  
**  
OUTPUT:** **OBSERVATION:** The program successfully generates prime numbers in a given range using a for loop.  
The optimized version with the square root method improves efficiency by reducing unnecessary checks.  
This demonstrates the use of loops, conditional statements, and optimization techniques in Python.

**#TASK5:  
PROMPT:** Write a Python program that defines a Library class with the following methods:

1. add\_book(title) → to add a new book to the library collection.
2. issue\_book(title) → to issue a book if available, otherwise show an appropriate message.
3. display\_books() → to display the list of available books.

**CODE:**



**OUT PUT:**

**OBSERVATION:** The Library class effectively demonstrates object-oriented programming concepts such as class, methods, and encapsulation.  
Edge cases like issuing unavailable books are properly handled with informative messages.