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/I	**Regulation**	R24
Date and Day of Assignment	Week5 - Monday	**Time(s)**	
Duration	2 Hours	**Applicable to Batches**	
AssignmentNumber:10.1(Present assignment number)/**24**(Total number of assignments)			

Q.No.	Question	*Expected Time to complete*
1	**Lab 10 – Code Review and Quality: Using AI to Improve Code Quality and Readability** **Lab Objectives** • Use AI for automated code review and quality enhancement. • Identify and fix syntax, logical, performance, and security issues in Python code. • Improve readability and maintainability through structured refactoring and comments.	Week5 - Monday

	• Apply prompt engineering for targeted improvements. • Evaluate AI-generated suggestions against PEP 8 standards and software engineering best practices	

	Task Description #1 – Syntax and Logic Errors
	Task: Use AI to identify and fix syntax and logic errors in a faulty Python script.
	Sample Input Code:
	# Calculate average score of a student
	def calc_average(marks):
	total = 0
	for m in marks:
	total += m
	average = total / len(marks)
	return avrage # Typo here
	
	marks = [85, 90, 78, 92]
	print("Average Score is ", calc_average(marks)
	Expected Output:
	• Corrected and runnable Python code with explanations of the fixes.	

task1.py | task2.py | task3.py | task4.py | task5.py

C: > Users > MEGHANA > OneDrive > Documents > AIAC > ASSIGNMENT-10.1 > task1.py > ...

```python
def calc_average(marks):
    if not marks:
        return 0  # Return 0 if the list is empty to avoid division by ze
    total = 0
    for m in marks:
        total += m
    average = total / len(marks)
    return average  # Fixed typo

marks = [85, 90, 78, 92]
print("Average Score is ", calc_average(marks))

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\MEGHANA\OneDrive\Documents\AIAC\ASSIGNMENT-10.1> ^C
PS C:\Users\MEGHANA\OneDrive\Documents\AIAC\ASSIGNMENT-10.1>
PS C:\Users\MEGHANA\OneDrive\Documents\AIAC\ASSIGNMENT-10.1>  c:; cd 'c:\Users\MEGHA
Users\MEGHANA\.vscode\extensions\ms-python.debugpy-2025.10.0-win32-x64\bundled\libs\
GNMENT-10.1\task1.py'
Could not find platform independent libraries <prefix>
Could not find platform independent libraries <prefix>
Average Score is  86.25
PS C:\Users\MEGHANA\OneDrive\Documents\AIAC\ASSIGNMENT-10.1>
```

Task Description #2 – PEP 8 Compliance

Task: Use AI to refactor Python code to follow PEP 8 style guidelines.

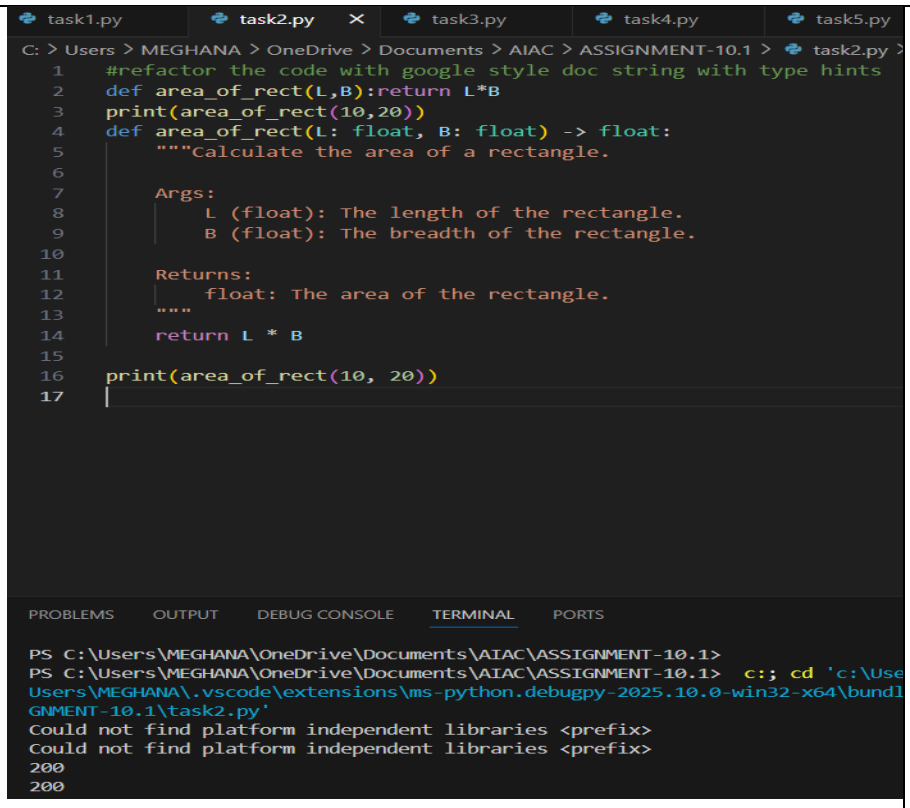
Sample Input Code:

def area_of_rect(L,B):return L*B

print(area_of_rect(10,20))

Expected Output:

- Well-formatted PEP 8-compliant Python code.



Task Description #3 – Readability Enhancement

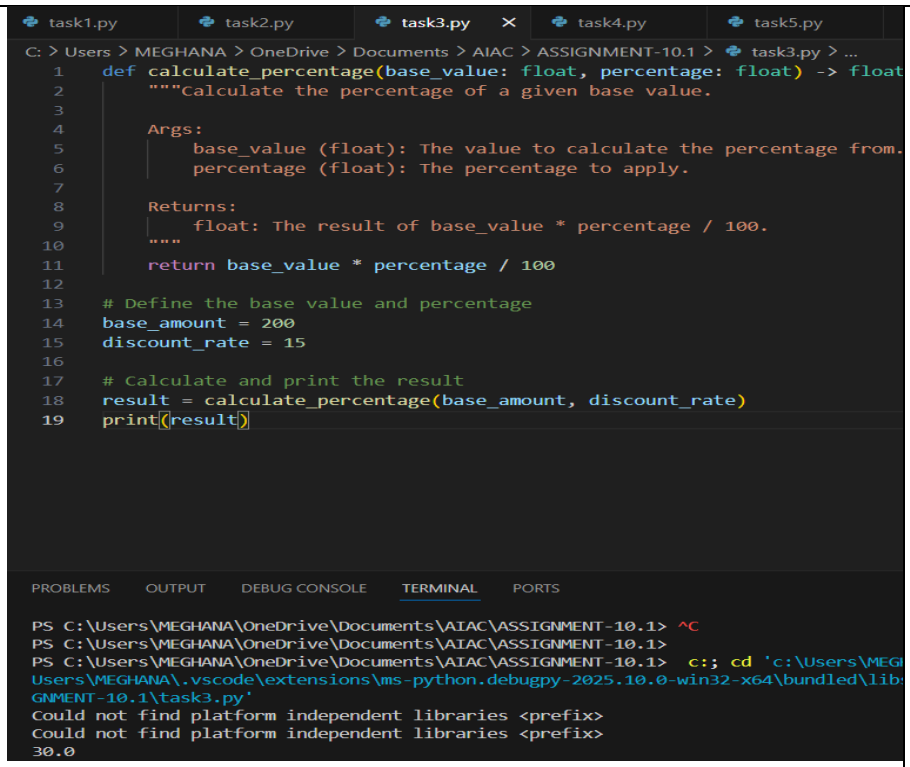
Task: Use AI to make code more readable without changing its logic**.**

Sample Input Code:

```
def c(x,y):
 return x*y/100
a=200
b=15
print(c(a,b))
```

Expected Output:

- Python code with descriptive variable names, inline comments, and clear formatting.



Task Description #4 – Refactoring for Maintainability

Task: Use AI to break repetitive or long code into reusable functions.

Sample Input Code:

students = ["Alice", "Bob", "Charlie"]

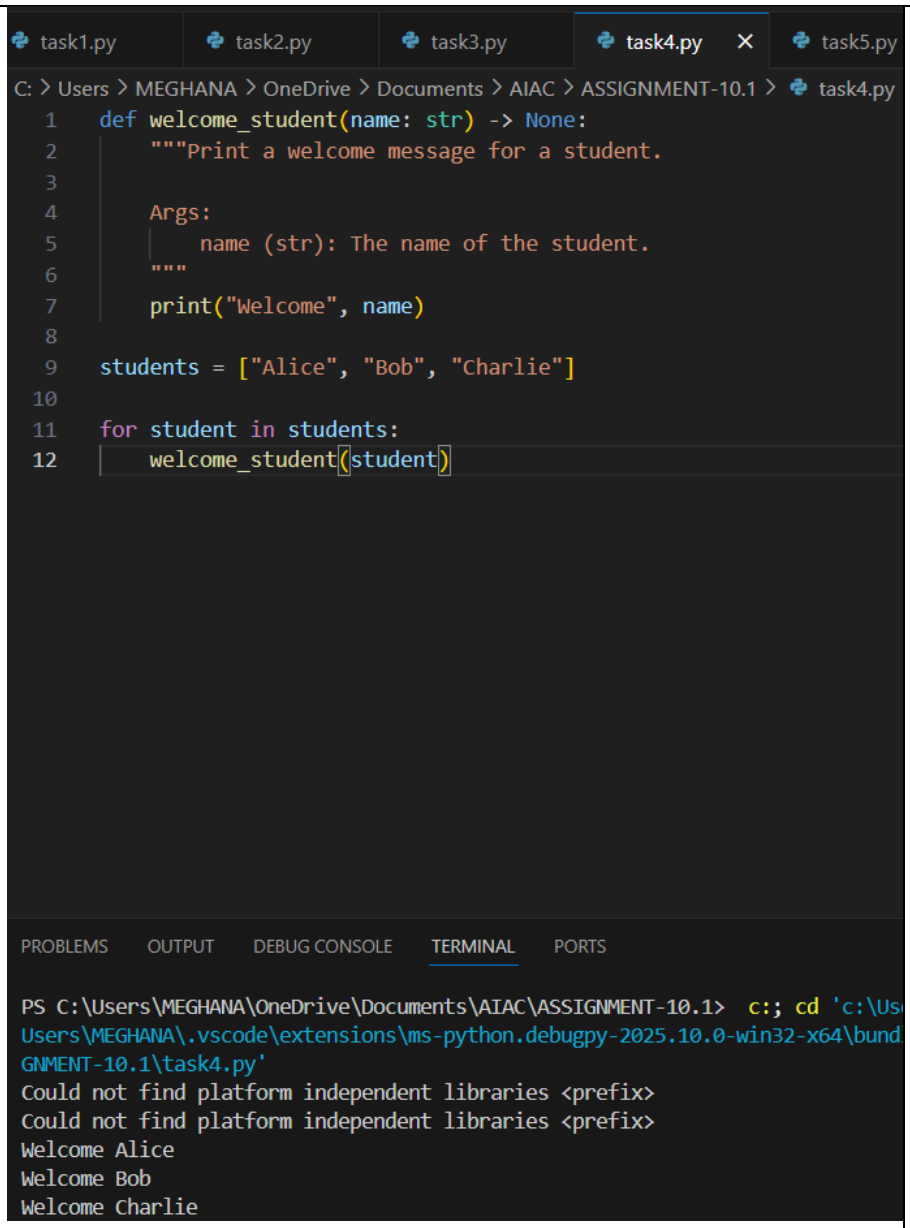
print("Welcome", students[0])

print("Welcome", students[1])

print("Welcome", students[2])

Expected Output:

- Modular code with reusable functions.



Task Description #5 – Performance Optimization

Task: Use AI to make the code run faster.

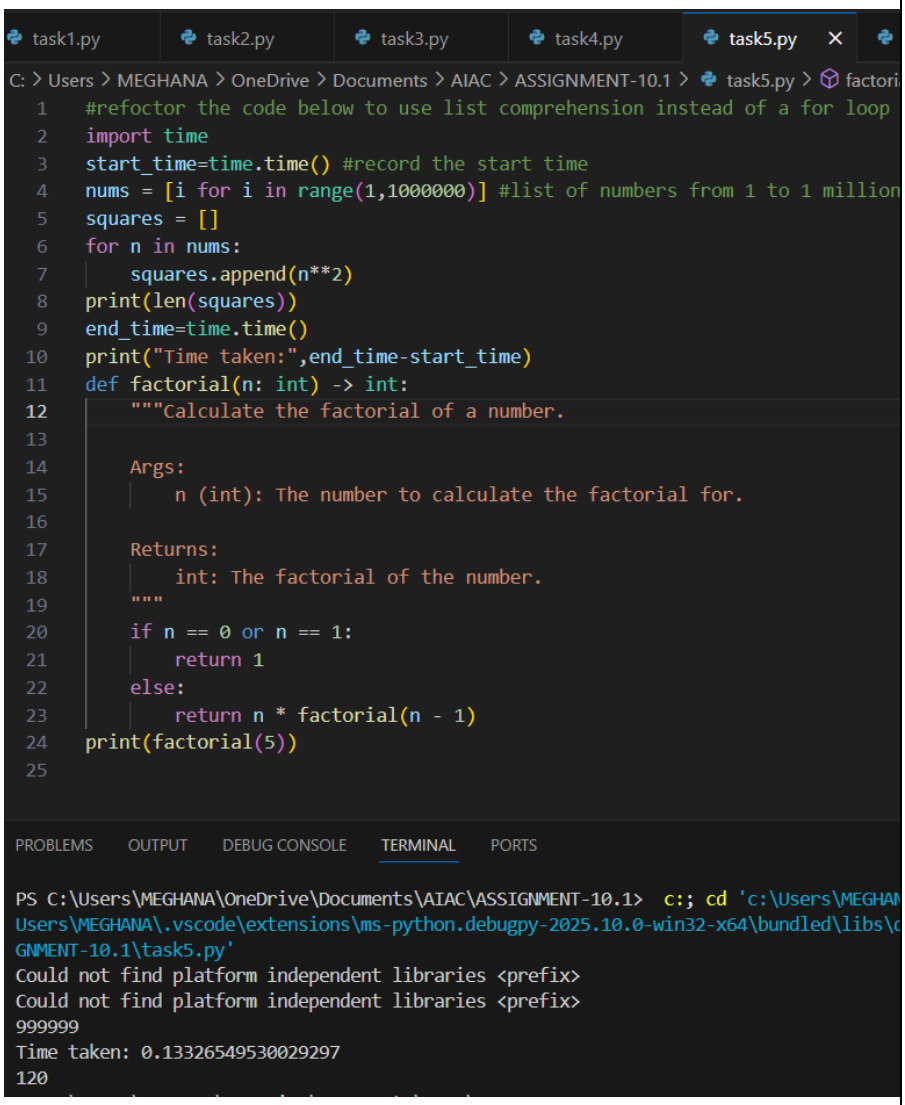
Sample Input Code:

```
# Find squares of numbers
nums = [i for i in range(1,1000000)]
squares = []
for n in nums:
    squares.append(n**2)
```

print(len(squares))

Expected Output:

- Optimized code using list comprehensions or vectorized operations.



Task Description #6 – Complexity Reduction

Task: Use AI to simplify overly complex logic.

Sample Input Code:

```
def grade(score):
    if score >= 90:
        return "A"
    else:
        if score >= 80:
```

```
        return "B"
    else:
        if score >= 70:
            return "C"
        else:
            if score >= 60:
                return "D"
            else:
                return "F"
```

Expected Output:

- Cleaner logic using elif or dictionary mapping.

task1.py | task2.py | task3.py | task4.py | task5.py | task6.py

C: > Users > MEGHANA > OneDrive > Documents > AIAC > ASSIGNMENT-10.1 > task6.py > ...

```python
def grade(score):
    if score >= 90:
        return "A"
    else:
        if score >= 80:
            return "B"
        else:
            if score >= 70:
                return "C"
            else:
                if score >= 60:
                    return "D"
                else:
                    return "F"
                # Cleaner logic using elif
                # Option 1: Using elif
                # Remove all nested else blocks and use elif for clarity

                # Option 1 implementation:
                # (Place this as the entire function body)

                if score >= 90:
                    return "A"
                elif score >= 80:
                    return "B"
                elif score >= 70:
                    return "C"
                elif score >= 60:
                    return "D"
                else:
                    return "F"
print(grade(55))  # Output: "F"
print(grade(95))  # Output: "A"
print(grade(85))  # Output: "B"
print(grade(75))  # Output: "C"
print(grade(65))  # Output: "D"
print(grade(55))  # Output: "F"
```

```python
print(grade(55))  # Output: "F"

                # Option 2: Using dictionary mapping with ranges
                # (Alternative approach)

                # grades = {range(90, 101): "A", range(80, 90): "B", range(70, 80): "C", range(60, 70): "D"}
                # for r, grade_letter in grades.items():
                #     if score in r:
                #         return grade_letter
                # return "F"
```

OUTPUT:

```
F
A
B
C
D
F
PS C:\Users\MEGHANA\OneDrive\Documents\AIAC\ASSIGNMENT-10.1>
```