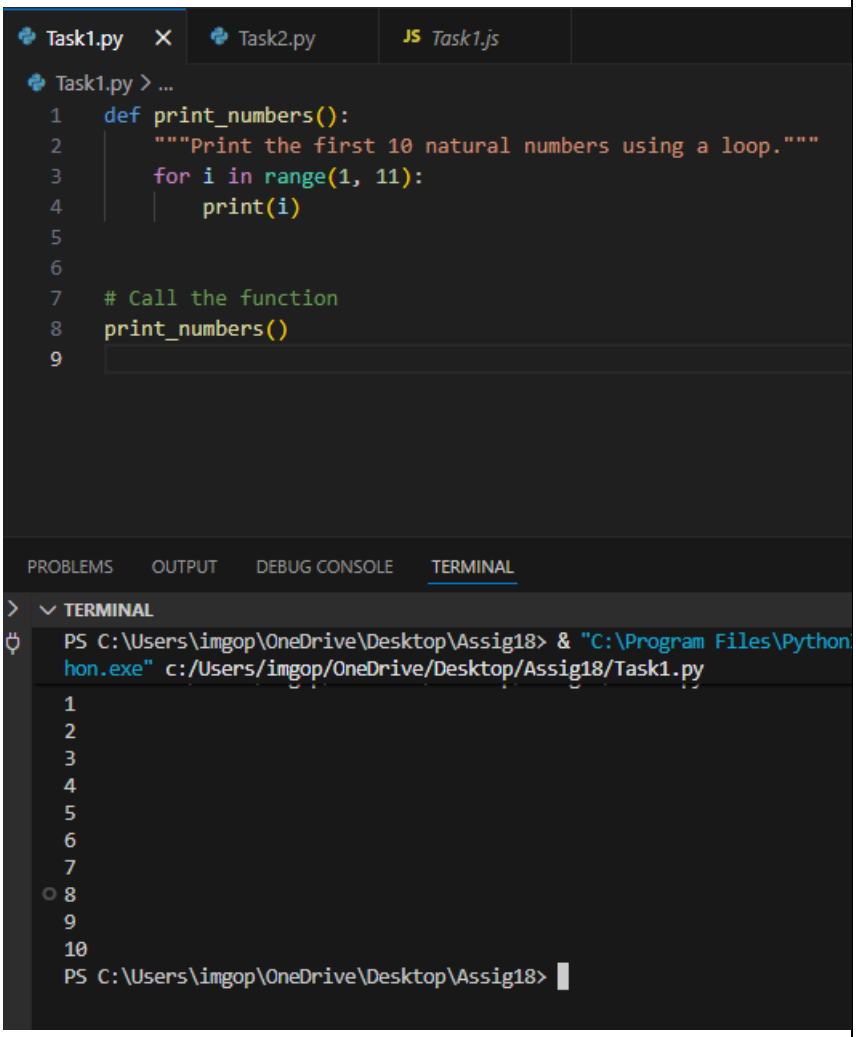
SCHOOLOFCOMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
ProgramName: M. Tech/MCA/MSC		**AssignmentType: Lab**	**AcademicYear:**2025-2026
CourseCoordinatorName		Venkataramana Veeramsetty	
CourseCode		**CourseTitle**	AI Assisted Problem Solving Using Python
Year/Sem	I/I	**Regulation**	R25
DateandDay of Assignment	08.12.2025	**Time(s)**	
Duration	2 Hours	**Applicableto Batches**	
AssignmentNumber:19.4(Presentassignmentnumber)/**24**(Totalnumberofassignments)			

Q.No.	Question	*ExpectedTime to complete*

1	**Lab 19 – Code Translation: Converting Between Programming Languages**
	Lab Objectives:
	• Understand how AI tools can assist in translating code between different programming languages.
	• Learn to verify correctness and functionality after translation.
	• Explore syntactic and semantic differences between languages (e.g., Python, Java, C++).
	• Practice debugging and optimizing AI-translated code.
	Task 1: Translate a Simple Program (Python → JavaScript)
	• **Instructions:**
	  • Write a Python function print_numbers() that prints the first 10 natural numbers using a loop.
	  • Translate the function into JavaScript as a reusable function printNumbers().
	  • Call the function in both languages to display results.
	• **Expected Output:**
	• **1**
	• **2**
	• **3**
		08.12.2025

- **...10**



Task 2: Convert Conditional Statements (Java → Python)

- **Instructions:**
  - Write a Java method checkNumber(int num) that checks if a number is positive, negative, or zero.
  - Translate the method into a Python function check_number(num).
  - Call the function/method with different inputs and compare outputs.
- **Expected Output:**
  - Input: -5 → Output: The number is negative
  - Input: 0 → Output: The number is zero
  - Input: 7 → Output: The number is positive

Task2.py > ...

```python
def check_number(num):
    """Check if a number is positive, negative, or zero."""
    if num > 0:
        return "The number is positive"
    elif num < 0:
        return "The number is negative"
    else:
        return "The number is zero"


# Test with different inputs
test_inputs = [-5, 0, 7]

print("Python Output:")
for num in test_inputs:
    result = check_number(num)
    print(f"Input: {num} → Output: {result}")

```

```
PS C:\Users\imgop\OneDrive\Desktop\Assig18> node "c:\Users\imgop\OneDrive\De
2.js"
2.js"
JavaScript Output:
Input: -5 → Output: The number is negative
Input: 0 → Output: The number is zero
Input: 7 → Output: The number is positive
PS C:\Users\imgop\OneDrive\Desktop\Assig18>
```

Task 3: Translate Recursive Function (Python → C++)

- **Instructions:**
  - Write a Python function factorial(n) that calculates factorial of a number using recursion.
  - Translate the same into a C++ function int factorial(int n).
  - Call the function in both languages with inputs 5 and 0.
- **Expected Output:**
  - **Input: 5 → Output: Factorial = 120**
  - **Input: 0 → Output: Factorial = 1**

.

Task3.cpp > ...

```cpp
#include <iostream>
using namespace std;

// Function to calculate factorial using recursion
int factorial(int n) {
    if (n < 0) {
        cout << "Error: Factorial not defined for negative numbers" <<
        return -1;
    } else if (n == 0 || n == 1) {
        return 1;
    } else {
        return n * factorial(n - 1);
    }
}

int main() {
    // Test with different inputs
    int testInputs[] = {5, 0};

    cout << "C++ Output:" << endl;
    for (int num : testInputs) {
        int result = factorial(num);
        cout << "Input: " << num << " → Output: Factorial = " << resul
    }

    return 0;
```

```
PS C:\Users\imgop\OneDrive\Desktop\Assig18>


PS C:\Users\imgop\OneDrive\Desktop\Assig18> python "c:\Users\imgop\OneDrive\D
esktop\Assig18\Task3.py" …
Python Output:
Input: 5 → Output: Factorial = 120
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python "c:\Users\imgop\OneDrive\D
esktop\Assig18\Task3.py"
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python "c:\Users\imgop\OneDrive\D
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python "c:\Users\imgop\OneDrive\D
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python "c:\Users\imgop\OneDrive\D
esktop\Assig18\Task3.py"
Python Output:
Input: 5 → Output: Factorial = 120
Input: 0 → Output: Factorial = 1
PS C:\Users\imgop\OneDrive\Desktop\Assig18> ^C
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python Task3.py
Python Output:
Python Output:
Input: 5 → Output: Factorial = 120
Input: 0 → Output: Factorial = 1
```

Task3.py > ...

```python
def factorial(n):
    """Calculate the factorial of a number using recursion."""
    if n < 0:
        return "Error: Factorial not defined for negative numbers"
    elif n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)


# Test with different inputs
test_inputs = [5, 0]

print("Python Output:")
for num in test_inputs:
    result = factorial(num)
    print(f"Input: {num} → Output: Factorial = {result}")
```

Task 4: Data Structures with Functions (JavaScript → Python)

- **Instructions:**
  - Write a JavaScript function printStudents(students) that takes an array of student names and prints each name.
  - Translate it into a Python function print_students(students) using a list.
  - Test both functions with sample student names.
- **Expected Output:**
- Student List:
- Alice
- Bob
- Charlie

Task4.py > ...

```python
def print_students(students):
    """Print each student name from a list."""
    print("Student List:")
    for student in students:
        print(student)


# Test with sample student names
student_names = ["Alice", "Bob", "Charlie"]
print_students(student_names)

```

Task4.js > ...

```javascript
function printStudents(students) {
    // Print each student name from an array
    console.l let student: any );
    for (let student of students) {
        console.log(student);
    }
}

// Test with sample student names
const studentNames = ["Alice", "Bob", "Charlie"];
printStudents(studentNames);

```

```
TERMINAL

Student List:
Alice
Bob
Charlie
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python Task4.py
Student List:
Alice
Bob
Charlie
PS C:\Users\imgop\OneDrive\Desktop\Assig18>

Student List:
Alice
Bob
Charlie
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python Task4.py
Student List:
Alice
Bob
Charlie
Student List:
Alice
Bob
Charlie
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python Task4.py
Student List:
Alice
```

Task 5: Class & Object Translation (Python → Java)

- **Instructions:**
    1. Write a **Python class** Car with attributes: brand, model, year.
    2. Add a **method**display_details() that prints car details.
    3. Translate the same into a **Java class** Car with attributes and a **method**displayDetails().
    4. Create an object in both languages and call the method.
- **Expected Output:**
- Car Details:
- Brand: Toyota
- Model: Corolla

Year: 2020

Task5.py > ...

```python
class Car:
    """A class to represent a car."""

    def __init__(self, brand, model, year):
        """Initialize car attributes."""
        self.brand = brand
        self.model = model
        self.year = year

    def display_details(self):
        """Display car details."""
        print("Car Details:")
        print(f"Brand: {self.brand}")
        print(f"Model: {self.model}")
        print(f"Year: {self.year}")


# Create an object and call the method
car = Car("Toyota", "Corolla", 2020)
car.display_details()
```

TERMINAL

```
Charlie
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python Task4.py
Student List:
Alice
Bob ...
Charlie
PS C:\Users\imgop\OneDrive\Desktop\Assig18>

Charlie
PS C:\Users\imgop\OneDrive\Desktop\Assig18>
Charlie
Charlie
PS C:\Users\imgop\OneDrive\Desktop\Assig18> ^C
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python "c:\Users\imgop\OneDrive\Desktop\Assig18\Task5.py"
Car Details:
Brand: Toyota
Model: Corolla
Year: 2020
PS C:\Users\imgop\OneDrive\Desktop\Assig18> python Task5.py
Car Details:
Brand: Toyota
Model: Corolla
Year: 2020
PS C:\Users\imgop\OneDrive\Desktop\Assig18>
```

✅ Deliverables (For All Tasks)

1. AI-generated prompts for code and test case generation.
2. At least 3 assert test cases for each task.

	3. AI-generated initial code and execution screenshots. 4. Analysis of whether code passes all tests. 5. Improved final version with inline comments and explanation. 6. Compiled report (Word/PDF) with prompts, test cases, assertions, code, and output.	