

| SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE | | DEPARTMENT OF COMPUTER SCIENCE ENGINEERING | |
|---|--|--|--|
| Program Name: M. Tech/MCA | | Assignment Type: Lab | AcademicYear:2025-2026 |
| Course Coordinator Name | | Venkataramana Veeramsetty | |
| Course Code | | Course Title | AI Assisted Problem Solving Using Python |
| Year/Sem | I/I | Regulation | R24 |
| Date and Day of Assignment | Week1 - Monday | Time(s) | |
| Duration | 2 Hours | Applicable to Batches | M. Tech/MCA |
| AssignmentNumber:1.3(Present assignment number)/24(Total number of assignments) | | | |
| Q.No. | Question | | Expected Time to complete |
| 1 | <p>Lab 1: Environment Setup – GitHub Copilot and VS Code Integration</p> <p>Lab Objectives:</p> <ul style="list-style-type: none"> To install and configure GitHub Copilot in Visual Studio Code. To explore AI-assisted code generation using GitHub Copilot. To analyze the accuracy and effectiveness of Copilot's code suggestions. To understand prompt-based programming using comments and code context <p>Lab Outcomes (LOs): After completing this lab, students will be able to:</p> <ul style="list-style-type: none"> Set up GitHub Copilot in VS Code successfully. Use inline comments and context to generate code with Copilot. Evaluate AI-generated code for correctness and readability. Compare code suggestions based on different prompts and programming styles. <p>Task Description#1</p> <ul style="list-style-type: none"> Install and configure GitHub Copilot in VS Code. Take screenshots of each step. <p>Expected Output#1</p> <ul style="list-style-type: none"> Install and configure GitHub Copilot in VS Code. Take screenshots of each step. | | Week1 - Wednesday |

Task Description#2

- Use Copilot to generate a is_prime() Python function.

Expected Output#2

- Function to check primality with correct logic.

Prompt-1: Write a python function(is_prime) to check whether the given number prime or not and also able to take user input

Task1.py:

```
def is_prime(n):
    """Check if a number is prime."""
    if n <= 1:
        return False
    if n == 2:
        return True
    if n % 2 == 0:
        return False
    for i in range(3, int(n ** 0.5) + 1, 2):
        if n % i == 0:
            return False
    return True

# Take user input
try:
    num = int(input("Enter a number to check if it is prime: "))
    if is_prime(num):
        print(f"{num} is a prime number.")
    else:
        print(f"{num} is not a prime number.")
except ValueError:
    print("Please enter a valid integer.")
```

OUTPUT:

```
PS C:\Users\moham\Desktop\Python> & "C:/Program Files/Python314/python.exe" c:/Users/moham/Desktop/Python/Code.p
Enter a number to check if it is prime: 3
3 is a prime number.
PS C:\Users\moham\Desktop\Python> & "C:/Program Files/Python314/python.exe" c:/Users/moham/Desktop/Python/Code.p
Enter a number to check if it is prime: 15
15 is not a prime number.
```

Task Description#3

- Write a comment like # Function to reverse a string and use Copilot to generate the function.

Expected Output#3

- Auto-completed reverse function

Prompt-1: Write a python Function to reverse a string by taking input from user

Task3.py:

```
def reverse_string(s):  
    """Return the reversed string."""  
    return s[::-1]  
  
# Take user input  
user_input = input("Enter a string to reverse: ")  
reversed_str = reverse_string(user_input)  
print(f"Reversed string: {reversed_str}")
```

```
PS C:\Users\moham\Desktop\Python> & "C:/Program Files/Python314/python.exe" c:/Users/moham/Desktop/Python/Code.py  
Enter a string to reverse: Madam  
Reversed string: madaM  
PS C:\Users\moham\Desktop\Python> & "C:/Program Files/Python314/python.exe" c:/Users/moham/Desktop/Python/Code.py  
Enter a string to reverse: Nizam  
Reversed string: maziN
```

Task Description#4

- Generate both recursive and iterative versions of a factorial function using comments..

Expected Output#4

- Two working factorial implementations

Prompt-1: write a python function for both recursive and iterative versions of a factorial function using comments and taking input from user

Task4.py

```
# Recursive version of factorial
def factorial_recursive(n):
    """Calculate factorial recursively."""
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial_recursive(n - 1)

# Iterative version of factorial
def factorial_iterative(n):
    """Calculate factorial iteratively."""
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

# Take user input
try:
    num = int(input("Enter a non-negative integer: "))
    if num < 0:
        print("Please enter a non-negative integer.")
    else:
        print(f"Recursive factorial of {num}: {factorial_recursive(num)}")
        print(f"Iterative factorial of {num}: {factorial_iterative(num)}")
except ValueError:
    print("Please enter a valid integer.")
```

OUTPUT:

```
PS C:\Users\moham\Desktop\Python> & "C:/Program Files/Python314/python.exe" c:/Users/moham/Desktop/Python/Code.py
Enter a non-negative integer: 5
Recursive factorial of 5: 120
Iterative factorial of 5: 120
```

Task Description#5

- Use Copilot to find the largest number in a list. Assess code quality and efficiency.

Expected Output#5

- A valid function with your review

Prompt-1: write a python function to find the largest number in a list. Assess code quality and efficiency by taking input from user

Task5.py

```

def find_largest(numbers):
    """Return the largest number in a list."""
    if not numbers:
        return None # Handle empty list
    largest = numbers[0]
    for num in numbers[1:]:
        if num > largest:
            largest = num
    return largest

# Take user input
try:
    user_input = input("Enter numbers separated by spaces: ")
    num_list = [float(x) for x in user_input.split()]
    if not num_list:
        print("List is empty. Please enter at least one number.")
    else:
        largest = find_largest(num_list)
        print(f"The largest number in the list is: {largest}")
except ValueError:
    print("Please enter valid numbers.")

# Assessment:
# - The function uses a single pass (O(n)), which is efficient for this task.
# - Handles empty list input gracefully.
# - Uses float conversion for broader numeric input support.
# - Code is readable and well-commented.

```

OUTPUT:

```

PS C:\Users\moham\Desktop\Python> & "C:/Program Files/Python314/python.exe" c:/Users/moham/Desktop/Python/Code.py
Enter numbers separated by spaces: 5 2 9 6 3 14 1 12
The largest number in the list is: 14.0

```

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 |
|---------------------------------------|-----------------|
| Successful Setup of Copilot (Task #1) | 2 |
| is_prime() Python function (Task #2) | 2 |
| Reverse a string function (Task #3) | 2 |
| Factorial Function (Task #4) | 2 |
| Find the largest number (Task #5) | 2 |
| Total | 10 Marks |