| 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 | | Dr. V. Venkataramana (Co-ordinator)<br>Dr. T. Sampath Kumar<br>Dr. Pramoda Patro<br>Dr. Brij Kishor Tiwari<br>Dr.J.Ravichander<br>Dr. Mohammand Ali Shaik<br>Dr. Anirodh Kumar<br>Mr. S.Naresh Kumar<br>Dr. RAJESH VELPULA<br>Mr. Kundhan Kumar<br>Ms. Ch.Rajitha<br>Mr. M Prakash<br>Mr. B.Raju<br>Intern 1 (Dharma teja)<br>Intern 2 (Sai Prasad)<br>Intern 3 (Sowmya)<br>NS_2 ( Mounika) | |
| CourseCode | 24CS002PC215 | CourseTitle | AI Assisted Coding |
| Year/Sem | II/I | Regulation | R24 |
| Date and Day of Assignment | Week1 - Tuesday | Time(s) | |
| Duration | 2 Hours | Applicableto Batches | 24CSBTB01 To 24CSBTB39 |
| AssignmentNumber:**1.2**(Present assignment number)/**24**(Total number of assignments) | | | |

| Q.No. | Question | *Expecte dTime to complet e* |
|---|---|---|
| 1 | Lab 1: Environment Setup – GitHub Copilot and VS Code Integration<br><br>**Lab Objectives:**<br>● To install and configure GitHub Copilot in Visual Studio Code. | Week1 - wednesday |

- To explore AI-assisted code generation using GitHub Copilot.
- To analyse the accuracy and effectiveness of Copilot's code suggestions.
- To understand prompt-based programming using comments and code context

**Lab Outcomes (LOs):**
After completing this lab, students will be able to:

- 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.

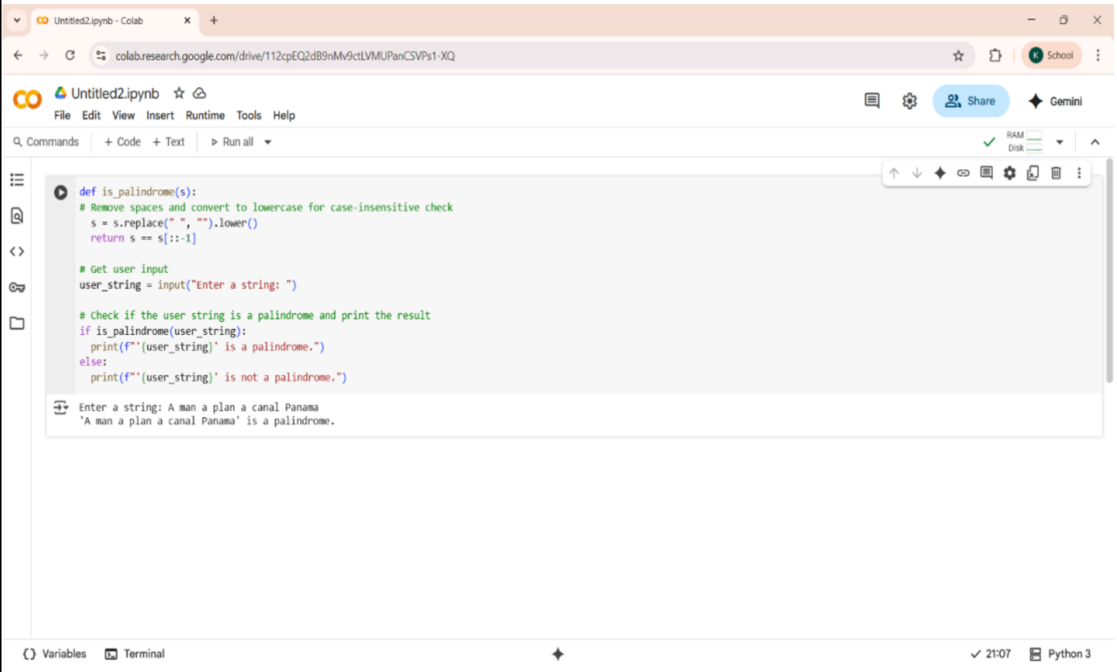
**Task Description#1**
- Write a comment: # Function to check if a string is a valid palindrome (ignoring spaces and case) and allow Copilot to complete it.

**Expected Output#1**
- A function that correctly returns True for phrases like "A man a plan a canal Panama"

**Prompt #1:**
1. Write a function to check if a string is a palindrome.
2. Ignore spaces in the string while checking.

**Observations:**
1. Co-pilot will generate automatic code for palindrome.
2. It should ignore spaces and case while checking.
3. Copilot will likely make a reverse-string check.

**CODE:**



```
def is_palindrome(s):
# Remove spaces and convert to lowercase for case-insensitive check
  s = s.replace(" ", "").lower()
  return s == s[::-1]

# Get user input
user_string = input("Enter a string: ")

# Check if the user string is a palindrome and print the result
if is_palindrome(user_string):
  print(f"'{user_string}' is a palindrome.")
else:
  print(f"'{user_string}' is not a palindrome.")
```

```
Enter a string: A man a plan a canal Panama
'A man a plan a canal Panama' is a palindrome.
```

**Code Explanation:**

1. user types a string
2. The program removes spaces and makes everything lowercase
3. It checks if the cleaned string is the same as its reverse
4. If same → prints" valid palindrome".
5. If not → prints "not a palindrome ".

**Task Description#2**

- Generate a Python function that returns the Fibonacci sequence up to n terms. Prompt with only a function header and docstring

**Expected Output#2**

- AI completes the function logic using loop or recursion with accurate output

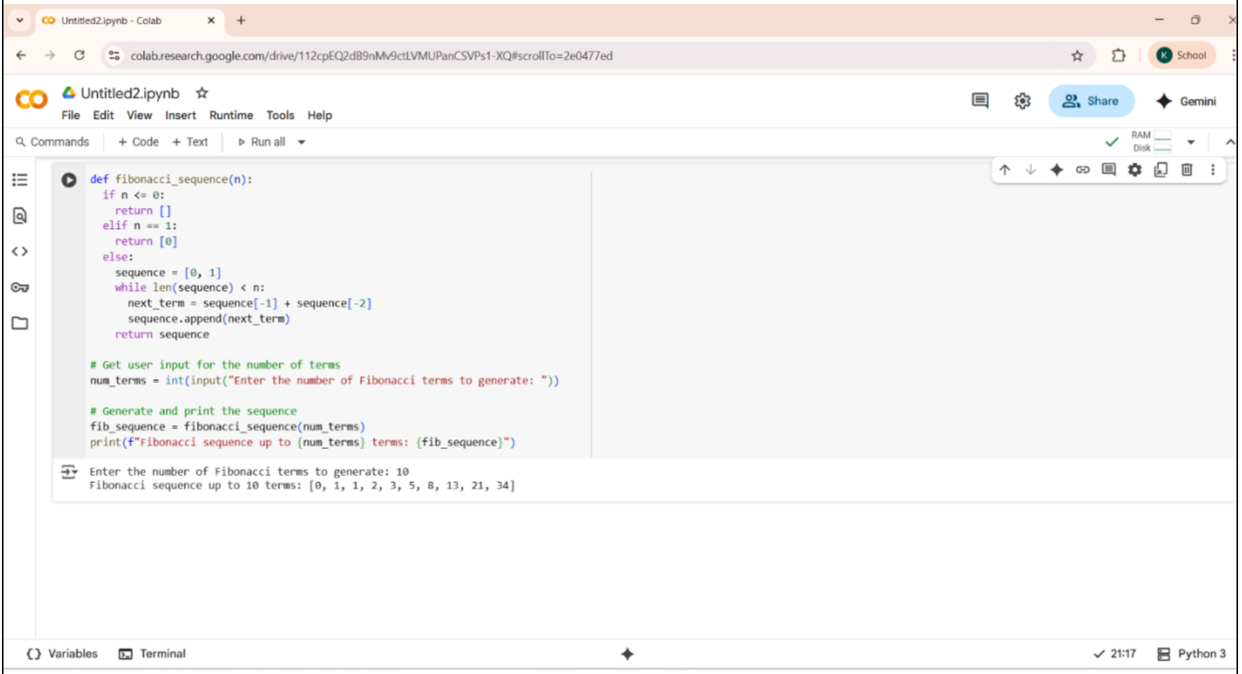
**Prompt #2:**

1. Taken as number of terms
2.Define a function Fibonacci(n).
3.Output should be a list of Fibonacci numbers.

**Observations:**

1. The prompt has only the function header and docstring.
2. Copilot understands it should generate Fibonacci numbers.
3. Output should match the sequence for given n.
4. AI may use 1oop or recursion to solve it.

**CODE:**


Untitled2.ipynb - Colab
colab.research.google.com/drive/112cpEQ2dB9nMv9ctLVMUPanCSVPs1-XQ#scrollTo=2e0477ed
Untitled2.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share
Gemini
Variables
Terminal
21:17
Python 3

```python
def fibonacci_sequence(n):
  if n <= 0:
    return []
  elif n == 1:
    return [0]
  else:
    sequence = [0, 1]
    while len(sequence) < n:
      next_term = sequence[-1] + sequence[-2]
      sequence.append(next_term)
    return sequence

# Get user input for the number of terms
num_terms = int(input("Enter the number of Fibonacci terms to generate: "))

# Generate and print the sequence
fib_sequence = fibonacci_sequence(num_terms)
print(f"Fibonacci sequence up to {num_terms} terms: {fib_sequence}")
```

```
Enter the number of Fibonacci terms to generate: 10
Fibonacci sequence up to 10 terms: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

**Code Explanation:**

This program starts with 0 and 1, then keeps adding the last two numbers to get the next one, until it prints as many terms as you asked for.

**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 #3:**

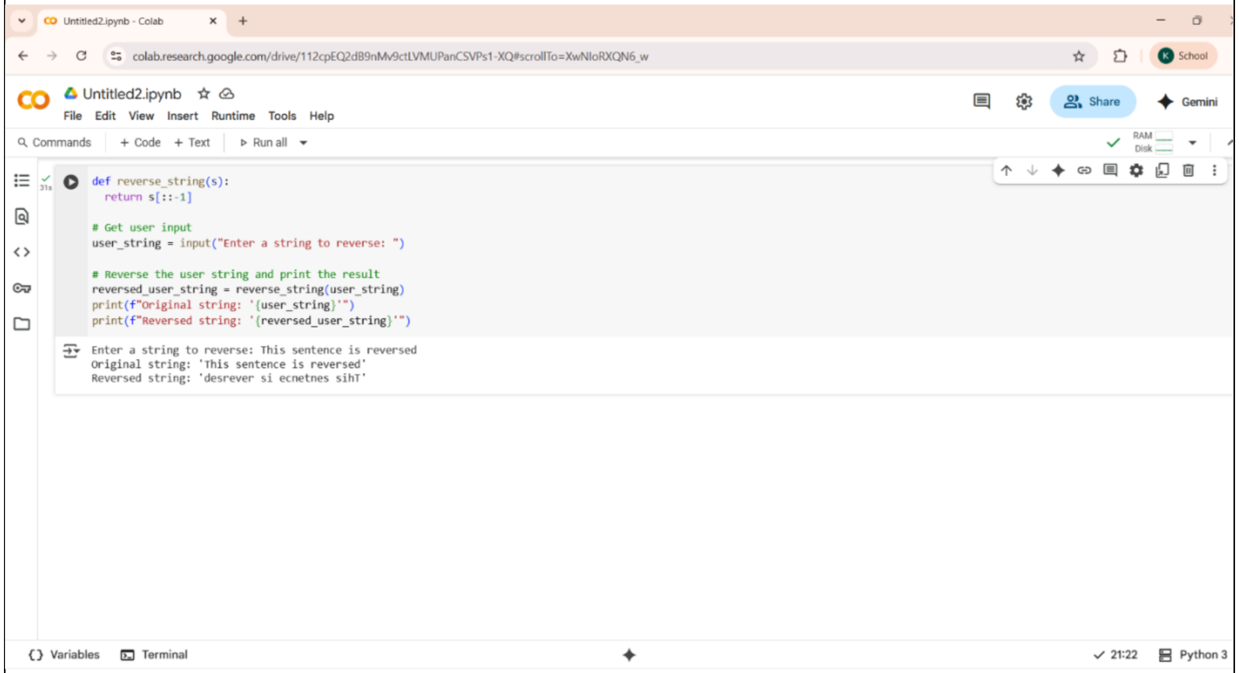
1.Function to reverse a string

2.Take input string 3. Return reversed string

**Observations:**

1. This tells Copilot to reverse a string.
2. Copilot will likely use slicing or a loop.
3. Works for any string input.
4. Should be tested with empty and single-character strings.

**CODE :**



```python
def reverse_string(s):
  return s[::-1]

# Get user input
user_string = input("Enter a string to reverse: ")

# Reverse the user string and print the result
reversed_user_string = reverse_string(user_string)
print(f"Original string: '{user_string}'")
print(f"Reversed string: '{reversed_user_string}'")
```

```
Enter a string to reverse: This sentence is reversed
Original string: 'This sentence is reversed'
Reversed string: 'desrever si ecnetnes sihT'
```

**Code Explantion:**

This program takes a word from you, flips it backward using Python slicing ([::-1]), and shows you the reversed result.

**Task Description#4**

- Generate a program that simulates a basic calculator (add, subtract, multiply, divide). Write the comment: # Simple calculator with 4 operations and let AI complete it.

**Expected Output#4**

- Fully working calculator with input/output and operator selection logic

**Prompt #4:**

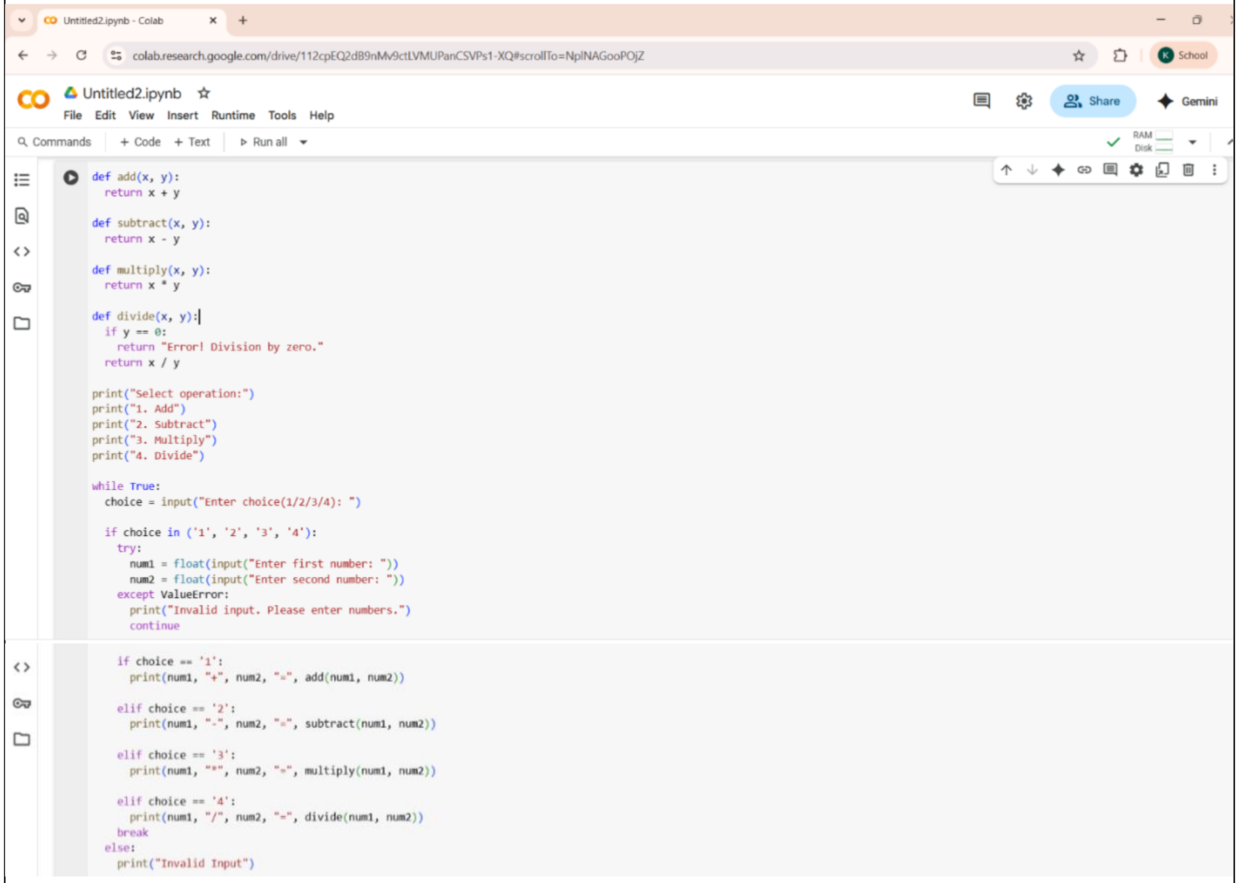
1.Simple calculator with 4 operations
2.Add, subtract, multiply, divide
3.Take user input and show result

**Observations:**

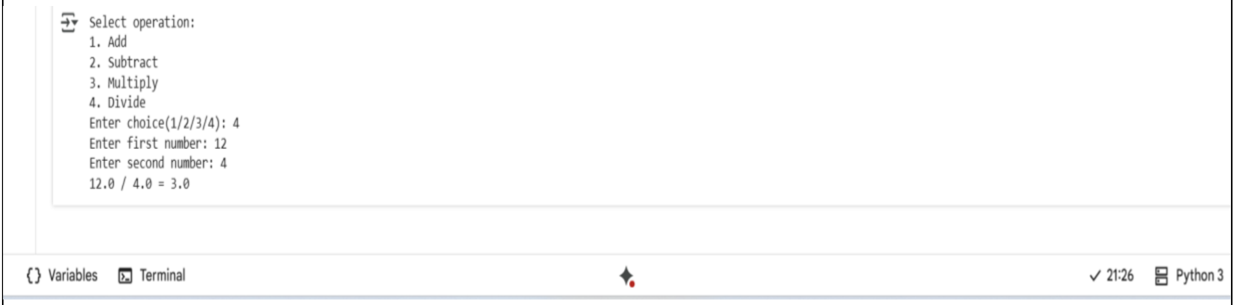
1. This tells Copilot to make a basic calculator.
2. It should support add, subtract, multiply, and divide.
3. Likely will include user input for numbers and operation choice.

4. Needs to handle division by zero.
5. Should be tested with different numbers and operators.

**CODE:**



```python
def add(x, y):
  return x + y

def subtract(x, y):
  return x - y

def multiply(x, y):
  return x * y

def divide(x, y):
  if y == 0:
    return "Error! Division by zero."
  return x / y

print("Select operation:")
print("1. Add")
print("2. Subtract")
print("3. Multiply")
print("4. Divide")

while True:
  choice = input("Enter choice(1/2/3/4): ")

  if choice in ('1', '2', '3', '4'):
    try:
      num1 = float(input("Enter first number: "))
      num2 = float(input("Enter second number: "))
    except ValueError:
      print("Invalid input. Please enter numbers.")
      continue

    if choice == '1':
      print(num1, "+", num2, "=", add(num1, num2))

    elif choice == '2':
      print(num1, "-", num2, "=", subtract(num1, num2))

    elif choice == '3':
      print(num1, "*", num2, "=", multiply(num1, num2))

    elif choice == '4':
      print(num1, "/", num2, "=", divide(num1, num2))
    break
  else:
    print("Invalid Input")
```



```
Select operation:
1. Add
2. Subtract
3. Multiply
4. Divide
Enter choice(1/2/3/4): 4
Enter first number: 12
Enter second number: 4
12.0 / 4.0 = 3.0
```

**Task Description#5**

- Use a comment to instruct AI to write a function that reads a file and returns the number of lines..

**Expected Output#5**

- Functional implementation using open() or with open() and readlines()

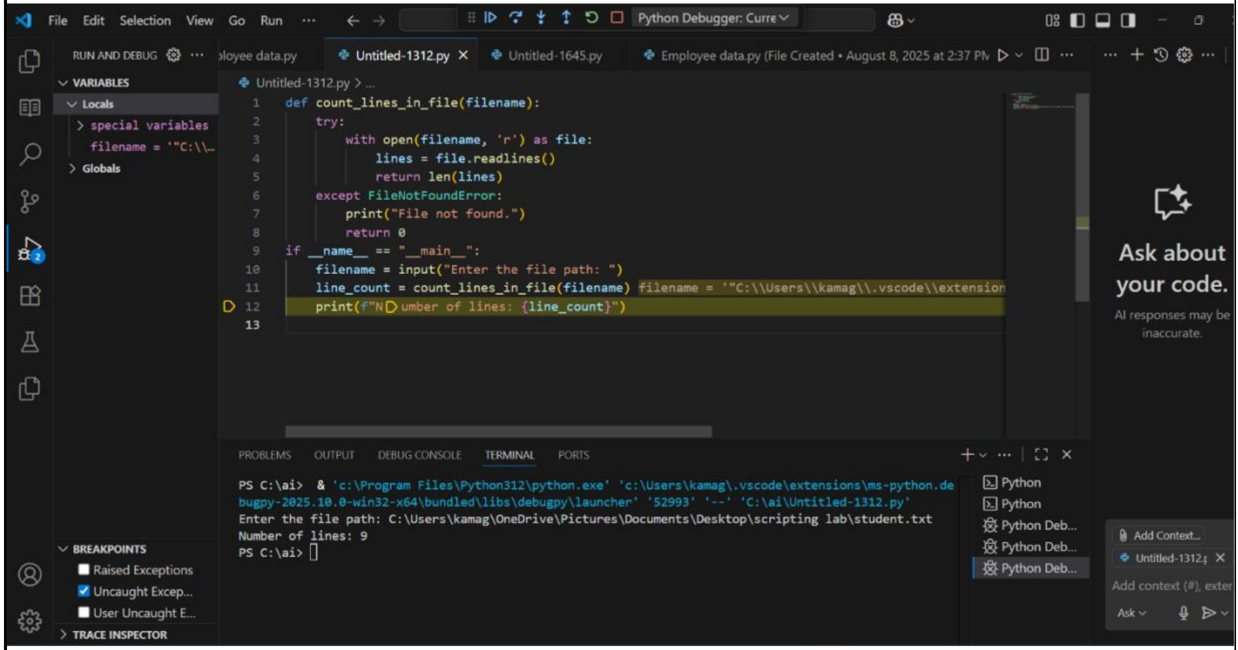
**<u>Prompt #5:</u>**

1.Function to read a file
2. Count the number of lines 3. Return the count

**<u>Observations:</u>**

1. This explains the task.
2. AI will likely use open() or with open() to read the file.
3. Counting can be done with Len (readlines ()) or a loop.
4. Needs testing with empty and multi-line files.

**<u>CODE:</u>**



```python
def count_lines_in_file(filename):
    try:
        with open(filename, 'r') as file:
            lines = file.readlines()
            return len(lines)
    except FileNotFoundError:
        print("File not found.")
        return 0
if __name__ == "__main__":
    filename = input("Enter the file path: ")
    line_count = count_lines_in_file(filename) filename = '"C:\\Users\\kamag\\.vscode\\extension
    print(f"Number of lines: {line_count}")
```

```
PS C:\ai>  & 'c:\Program Files\Python312\python.exe' 'c:\Users\kamag\.vscode\extensions\ms-python.de
bugpy-2025.10.0-win32-x64\bundled\libs\debugpy\launcher' '52993' '--' 'C:\ai\Untitled-1312.py'
Enter the file path: C:\Users\kamag\OneDrive\Pictures\Documents\Desktop\scripting lab\student.txt
Number of lines: 9
PS C:\ai> 
```

## Code Explanation:

This program opens a file, counts how many lines it has, and shows the result. If the file doesn't exist, it tells you.

**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 |
|---|---|
| Task #1 | 0.5 |
| Task #2 | 0.5 |
| Task #3 | 0.5 |
| Task #4 | 0.5 |
| Task #5 | 0.5 |
| **Total** | **2.5 Marks** |