SR UNIVERSITY

SchoolofComputerScienceandArtificialIntelligence

LabAssignment#2.2

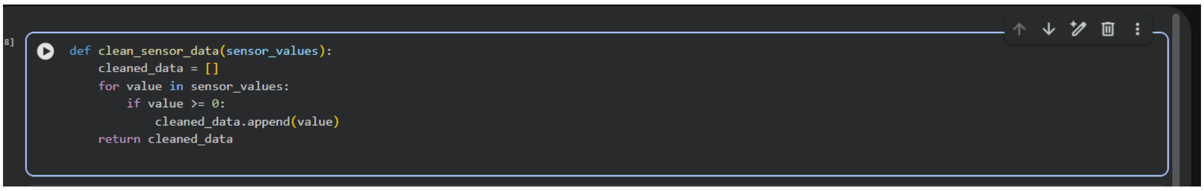
Program	:B.Tech(CSE)
Specialization	:
Course Title	: AI Assisted coding
Course Code	:
Semester	II
Academic Session	:2025-2026
Name of Student	: J Harshith yadav
Enrollment No.	: 2403A51L49
Batch No.	52
Date	:13-01-2026

SubmissionStartshere

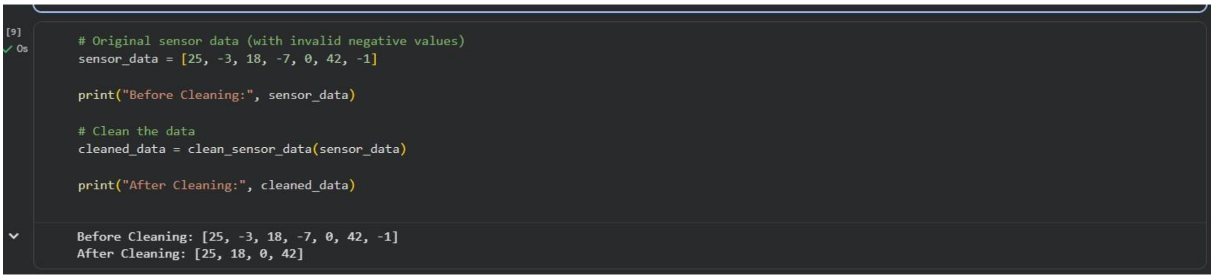
Task-1:CleaningSensorData

- ❖ Scenario:
- ❖ YouarecleaningIoTsensordatawherenegativevaluesareinvalid.
- ❖ Task:

Use Gemini in Colab togenerateafunctionthatfiltersoutallnegative numbers from a list.



```
def clean_sensor_data(sensor_values):
    cleaned_data = []
    for value in sensor_values:
        if value >= 0:
            cleaned_data.append(value)
    return cleaned_data
```



```
# Original sensor data (with invalid negative values)
sensor_data = [25, -3, 18, -7, 0, 42, -1]

print("Before Cleaning:", sensor_data)

# Clean the data
cleaned_data = clean_sensor_data(sensor_data)

print("After Cleaning:", cleaned_data)
```

```
Before Cleaning: [25, -3, 18, -7, 0, 42, -1]
After Cleaning: [25, 18, 0, 42]
```

Line-by-Line Explanation

1 def clean_sensor_data(sensor_values):

- Defines a function named **clean_sensor_data**
- sensor_valuesistheinputlistcontainingsensorreadings(mayincludenegative values)

2 cleaned_data = []

- Creates an **empty list**
- This list will store only **valid (non-negative)** sensor values

3 for value in sensor_values:

- Loops through **each value** in the input sensor list

- Processes one sensor reading at a time

4 if value >= 0:

- Checkswhether thesensor valueis **greaterthan orequal to zero**
- Thisconditionensuresthat **negativevaluesare ignored**

5 cleaned_data.append (value)

- Adds the valid sensor value to the cleaned_data list

6 returncleaned_data

- Returns the final list containing **only valid sensor readings**

▶ **ExampleExecutionExplanation**

sensor_data = [25, -3, 18, -7, 0, 42, -1]

- Input list contains both valid and invalid sensor values
-

cleaned_data = clean_sensor_data (sensor_data)

- Function removes all negative values

✅ **Output:**

[25, 18, 0, 42]

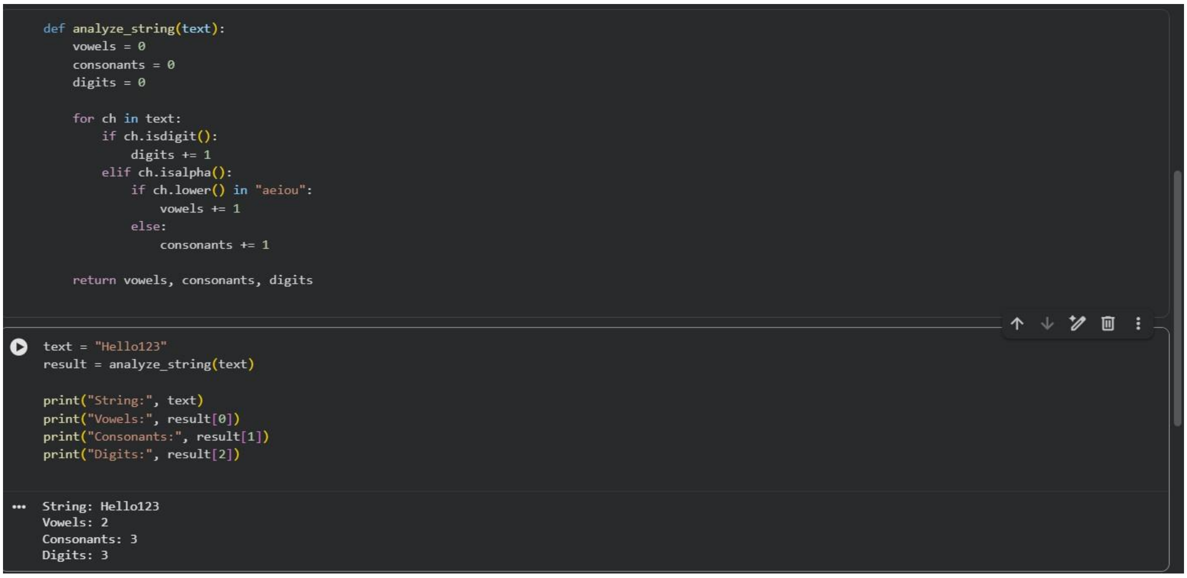
#Task2: StringCharacterAnalysis

❖ Scenario:

Youarebuildingatext-analysisfeature.

❖ Task:

UseGeminitogenerateaPythonfunctionthatcountsvowels,consonants, and digits in a string.



```
def analyze_string(text):
    vowels = 0
    consonants = 0
    digits = 0

    for ch in text:
        if ch.isdigit():
            digits += 1
        elif ch.isalpha():
            if ch.lower() in "aeiou":
                vowels += 1
            else:
                consonants += 1

    return vowels, consonants, digits
```

```
text = "Hello123"
result = analyze_string(text)

print("String:", text)
print("Vowels:", result[0])
print("Consonants:", result[1])
print("Digits:", result[2])
```

```
String: Hello123
Vowels: 2
Consonants: 3
Digits: 3
```

Explaination:

1️⃣ def analyze_string(text):

- Defines a function named analyze_string
- text is the input string that will be analyzed

2️⃣ vowels = 0, consonants = 0, digits = 0

- Initializes three counters
- These variables store the count of vowels, consonants, and digits

3️⃣ for ch in text:

- Loops through each character in the input string
- Processes one character at a time

4️⃣ if ch.isdigit():

- Checks if the character is a number (0–9)
- Iftrue,thedigitcounterisincreased

5️⃣digits += 1:

- Increments the digit count by 1

6️⃣ elif ch.isalpha():

- Checks if the character is an alphabet
- Ignores spaces and special characters

7️⃣ if ch.lower() in "aeiou":

- Converts the character to lowercase
- Checks if it is a vowel (a, e, i, o, u)

8️⃣ vowels += 1

- Increments the vowel count if the condition is true

9️⃣ **else:**

- If the alphabet is **not a vowel**, it must be a consonant

⍰**consonants+=1**

- Increments the consonant count

1️⃣1️⃣**returnvowels,consonants,digits**

- Returns all three counts as a tuple

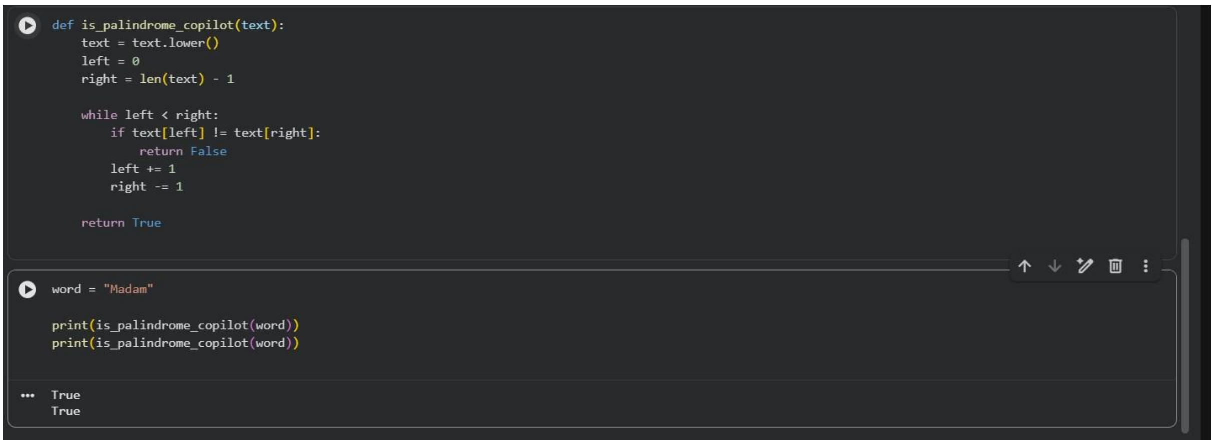
#Task3: PalindromeCheck–ToolComparison

❖ Scenario:

YoumustdecidewhichAItoolisclearerforstringlogic.

❖ Task:

Generateapalindrome-checkingfunctionusingGeminiandCopilot,then compare the results.



```
def is_palindrome_copilot(text):
    text = text.lower()
    left = 0
    right = len(text) - 1

    while left < right:
        if text[left] != text[right]:
            return False
        left += 1
        right -= 1

    return True
```

```
word = "Madam"

print(is_palindrome_copilot(word))
print(is_palindrome_copilot(word))
```

```
True
True
```

Explaination:

1 def is_palindrome_gemini(text):

- Defines a function to check whether a string is a palindrome
- textistheinputstring 2

text = text.lower()

- Converts all characters to lowercase
- Thisavoidscasemismatch(e.g.,Madamvsmadam) 3

text[::-1]

- Reverses the string using Python slicing
- [::-1]meansreadthestringfromendtostart 4

return text == text[::-1]

- Compares the original string with its reversed version
- ReturnsTrueifbotharethesame,otherwiseFalse def

is_palindrome_copilot(text):

- Definesafunctiontocheckifastringisapalindrome 2 text

= text.lower()

- Convertsthestringtolowercaseforcase-insensitivecomparison 3

left = 0

- Pointstothe**firstcharacter**ofthestring 4

right = len(text) - 1

- Pointstothe**lastcharacter**ofthestring 5

while left < right:

- Looprunsuntilbothpointersmeetinthemiddle 6 if

text[left] != text[right]:

- Compares characters from both ends
- If they are not equal, the string is **not a palindrome**

7 return False

- ImmediatelystopsandreturnsFalseifmismatchisfound 8

left += 1

- Movestheleftpointerforward 9

right -= 1

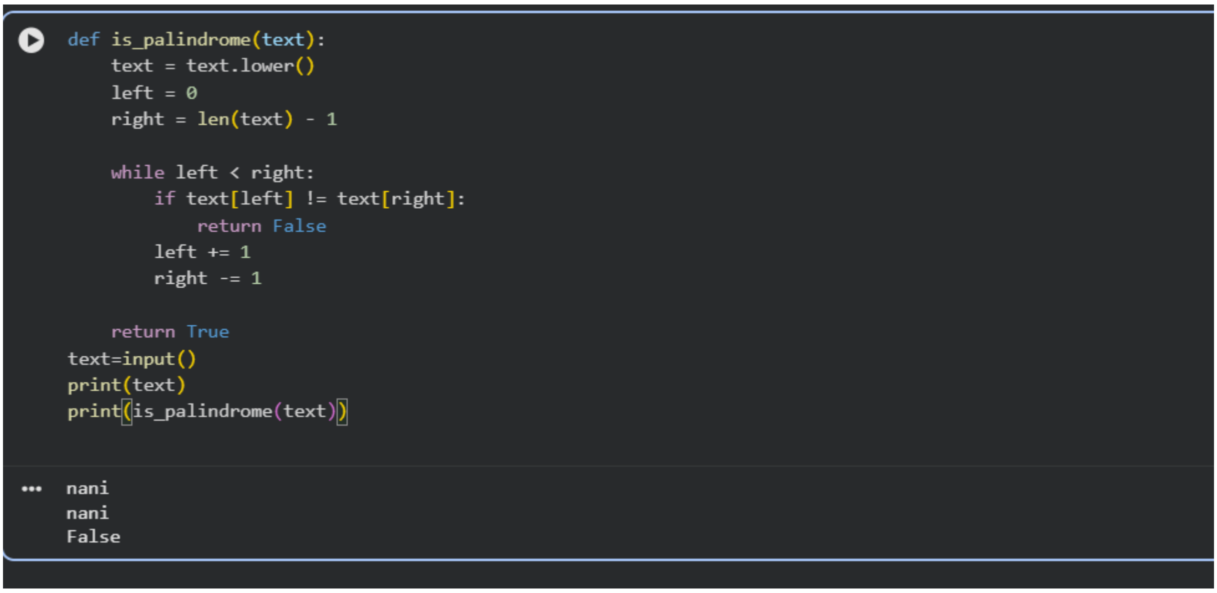
- Moves the right pointer backward

⍰return True

- If all characters match, the string is a palindrome

#Task4: CodeExplanationUsingAI

❖ Scenario:
Youarereviewingunfamiliarcodewrittenbyanotherdeveloper.
❖ Task:
AskGeminitoexplainaPythonfunction(primecheckORpalindrome check) line by line.



```
def is_palindrome(text):
    text = text.lower()
    left = 0
    right = len(text) - 1

    while left < right:
        if text[left] != text[right]:
            return False
        left += 1
        right -= 1

    return True
text=input()
print(text)
print(is_palindrome(text))
```

```
nani
nani
False
```

Explaination:

1️⃣ def is_palindrome(text):

- Defines a function named is_palindrome
- Takesastringtextasinput

2️⃣ text = text.lower()

- Converts all characters in the string to lowercase
- Ensures case-insensitive comparison (e.g., Madam = madam)

3️⃣ left = 0

- Initializes a pointer to the **first character** of the string

4️⃣ right = len(text) - 1

- Initializes a pointer to the **last character** of the string

5️⃣ while left <right:

- Starts a loop that runs until both pointers meet in the middle

6️⃣ if text[left] != text[right]:

- Compares characters at the left and right pointers
- If they are not equal, the string is **not a palindrome**

7️⃣ returnFalse

- Immediately exits the function if a mismatch is found

8️⃣ left += 1

- Moves the left pointer one step forward

9️⃣ right -= 1

- Moves the right pointer one step backward

? returnTrue

- If all character pairs match, the string is a palindrome