

AI ASSTD CODING - ASSIGNMNT-2

NAME: R.RONITH REDDY
HTNO: 2303A52280

TASK- 1 : CLEARING SENSOR DATA

PROMPT USED :

#WRITE A PYTHON CODE TO SEPARATE ALL THE NEGATIVE NUMBERS FROM THE SER GIVEN LIST OF NUMBERS AND SHOW NEW LIST WHICH CONTAINS NO EGATIVE NUMBER START THE CODE FROM SCRAP AND IF THE LIST IS EMPTY GIVE STATEMENT OR TEXT SAYING THE DATA IS CLEAN IF THE NO SAY INVALID DATA FOUND"

The screenshot shows a Google Colab notebook titled "AI&AC_ASSIGNMENT_2". The code cell contains the following Python script:

```
print("invalid data found")
raise ValueError("Invalid number in input") # Raise to trigger outer exception

# Check if the evaluated data is actually a list
if not isinstance(data_list, list):
    print('invalid data found')
else:
    is_valid_list = True
    for item in data_list:
        # Check if all items in the list are numbers
        if not isinstance(item, (int, float)):
            print('invalid data found')
            is_valid_list = False
            break

    if is_valid_list:
        # Filter out negative numbers
        for num in data_list:
            if num >= 0:
                filtered_list.append(num)

        if not filtered_list:
            print('the data is clean')
        else:
            print('Filtered list:', filtered_list)

    except (ValueError, SyntaxError):
        # Catch errors if input is not a valid Python literal or if parsing comma-separated f
        if 'invalid data found' not in locals() and 'invalid data found' not in globals():
            print('invalid data found')

... Enter a list of numbers (e.g., 1, -2, 3 or [1, -2, 3]): 9,80,76,56,-87,-65,-23
Filtered list: [9, 80, 76, 56]
```

The code is annotated with several comments and annotations from the AI system on the right side of the interface.

The screenshot shows a Google Colab notebook titled "AI&AC_ASSIGNMENT_2". The code cell contains the following Python script:

```
import ast

# Get user input for the list as a string
user_input = input("Enter a list of numbers (e.g., 1, -2, 3 or [1, -2, 3]): ")

filtered_list = []

try:
    # Attempt to parse as a literal first (e.g., [1,2,3] or (1,2,3))
    # If it's a simple comma-separated string, ast.literal_eval might fail or interpret a tuple
    try:
        data_list = ast.literal_eval(user_input)
        # If it was a tuple, convert to list for consistent processing
        if isinstance(data_list, tuple):
            data_list = list(data_list)
    except (ValueError, SyntaxError):
        # If ast.literal_eval fails, try to parse as comma-separated numbers
        data_list = []
        parts = user_input.split(',')
        for part in parts:
            stripped_part = part.strip()
            if not stripped_part: # Handle empty strings from consecutive commas or leading/trailing spaces
                continue
            try:
                # Try converting to int first, then float if int fails
                if '.' in stripped_part or 'e' in stripped_part.lower():
                    data_list.append(float(stripped_part))
                else:
                    data_list.append(int(stripped_part))
            except ValueError:
                print('invalid data found')
                raise ValueError("Invalid number in input") # Raise to trigger outer exception

Automatic saving failed. This file was updated remotely or in another tab. Show diff
```

The code is annotated with several comments and annotations from the AI system on the right side of the interface.

EXPLANATION :

GEMINI GENERATED THE CODE USING FUNCTIONS THAT FILTERS NEGATIVE NUMBER FROM THE LIST OF GIVEN INPUT BY FOLLOWING CONDITIONS . THE FUNCTION SHOWS THE OUTPUT CLEARLY , IF DATA IS CLEAN OR INVALID .

OUTPUT :

IT CLEARLY FILTERS THE NEGATIVE NUMBER FROM THE INPUT

OBSERVATION :

GEMINI GENERATED THE CODE ACCORDING TO THE PROMPT .
THE SOLUTION WAS VALID AND SUITS THE IOT DATA CLEARENCE

TASK-2 : STRING CHARACTER ANALYSIS

PROMPT USED :

#CREATE A PYTHON FUNCTION THAT ANALYZES A GIVEN STRING TO DETERMINE THE NUMBER OF VOWELS, CONSONANTS, AND DIGITS IT CONTAINS, THEN PRINT THE RESULTS USING AN EXAMPLE STRING.

The screenshot shows a Google Colab notebook titled "AI&AC_ASSIGNMENT_2 - Colab Notebook". The code cell contains the following Python function:

```
def analyze_string(input_string):
    vowel_count = 0
    consonant_count = 0
    digit_count = 0

    print(f"Original String: '{input_string}'")
    print(f"Number of Vowels: {vowel_count}")
    print(f"Number of Consonants: {consonant_count}")
    print(f"Number of Digits: {digit_count}")

    # Example usage:
    example_string = "Ronith @ 2205"
    analyze_string(example_string)

    print("\n--- Another Example ---")
    analyze_string("Data Centres 2026")
```

The output of the code cell is:

```
Original String: 'Ronith @ 2205'
Number of Vowels: 2
Number of Consonants: 4
Number of Digits: 4

--- Another Example ---
Original String: 'Data Centres 2026'
Number of Vowels: 4
Number of Consonants: 7
Number of Digits: 4
```

To the right of the code cell, there is a Gemini AI interface window. It shows a prompt from the user asking for a Python function to analyze a string, and a response from Gemini providing the code and some example outputs.

The screenshot shows a Google Colab notebook titled "AI&AC_ASSIGNMENT_2". A code cell in the left panel contains a Python function named `analyze_string` that counts vowels, consonants, and digits in a given string. The right panel shows the Gemini AI interface with a prompt asking to create a similar function, followed by a response from Gemini providing the code.

```

def analyze_string(input_string):
    vowels = "aeiou"
    consonants = "bcdfghjklmnpqrstvwxyz"

    vowel_count = 0
    consonant_count = 0
    digit_count = 0

    for char in input_string:
        char_lower = char.lower()
        if char_lower.isalpha():
            if char_lower in vowels:
                vowel_count += 1
            elif char_lower in consonants:
                consonant_count += 1
            elif char_lower.isdigit():
                digit_count += 1

    print(f"Original String: '{input_string}'")
    print(f"Number of Vowels: {vowel_count}")
    print(f"Number of Consonants: {consonant_count}")
    print(f"Number of Digits: {digit_count}")

# Example usage:
example_string = "Ronith @ 2205"
analyze_string(example_string)

print("\n--- Another Example ---")
analyze_string("Data Centres 2024")

```

Gemini

R Create a Python function that analyzes a given string to determine the number of vowels, consonants, and digits it contains, then print the results using an example string.

◆ Certainly! Here's a Python function that analyzes a string and counts the number of vowels, consonants, and digits it contains. I've also included an example to show how it works.

Accept & Run Accept Cancel

What can I help you build?

Gemini 2.5 Flash

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)

EXPLANATION :

GEMINI GENERATED A FUNCTION THAT CHECKS THE EACH CHARACTER BY USING LOOPS AND CONDITIONS , WHERE IT SEPARATED VOWELS , CONSONANTS AND DIGITS .

OUTPUT :

ORIGINAL STRING: 'RONITH @ 2205'
NUMBER OF VOWELS: 2
NUMBER OF CONSONANTS: 4
NUMBER OF DIGITS: 4

OBSERVATION :

GEMINI GIVEN THE CODE WHERE ITS SEPARATES VOWELS , CONSONANTS AND DIGITS IN A SIMPLER WAY , THE BUILT LOGIC IS EASILY UNDERSTANDABLE AND ACCURATE TO THE PROBLEM

TASK 3 : PALINDROME CHECK IN TWO DIFFERENT TOOLS

PROMPT :

#GENERATE A CODE FOR CHECKING WHETHER THE INPUT WORD IS PALINDROME BY CHECKING FUNCTION AND DYNAMIC INPUT

AI&AC_ASSIGNMENT_2 - Colab

To exit full screen, press and hold esc

File Edit View Insert Star notebook in Google Drive

Commands + Code + Text | Run all

```
[11] ✓ 6s
● def is_palindrome(word):
    # Convert the word to lowercase and remove spaces for accurate checking
    cleaned_word = ''.join(char.lower() for char in word if char.isalnum())
    # Compare the cleaned word with its reverse
    return cleaned_word == cleaned_word[::-1]

# Get dynamic input from the user
user_word = input("Enter a word or phrase to check if it's a palindrome: ")

# Check if it's a palindrome and print the result
if is_palindrome(user_word):
    print(f"'{user_word}' is a palindrome!")
else:
    print(f"'{user_word}' is not a palindrome.")

print("\n--- More Examples ---")
print("Racecar' is a palindrome: {is_palindrome('Racecar')}")
print("madam' is a palindrome: {is_palindrome('madam')}")
print("A man, a plan, a canal: Panama' is a palindrome: {is_palindrome('A man, a plan, a canal: Panama')}")
print("hello' is a palindrome: {is_palindrome('hello')}")
print("Python' is a palindrome: {is_palindrome('Python')}")


... Enter a word or phrase to check if it's a palindrome: MALAYALAM
'MALAYALAM' is a palindrome!
```

Start coding or generate with AI.

Automatic saving failed. This file was updated remotely or in another tab. Show diff

Gemini

R generate a code for checking whether the input word is palindrome by checking function and dynamic input

◆ Certainly! Here's a Python function to check if a word is a palindrome, along with a prompt for dynamic input:

Both of the last executed cells completed successfully.

For the string analysis, the code correctly identified the vowels, consonants, and digits in 'Ronith @ 2205' and 'Data Centres 2026'. For 'Ronith @ 2205', there were 2 vowels, 4 consonants, and 4 digits. For 'Data Centres 2026', it found 4 vowels, 7 consonants, and 4 digits.

Regarding the palindrome checker, you entered 'atta' which was correctly identified as a palindrome. The code also showed several examples, confirming that 'Racecar', 'madam', and 'A man, a plan, a canal: Panama' are palindromes, while 'hello' and 'Python' are not.

Write a function to check multiple words for palindromes

Generate Python code to sort a list of numbers

How can I create a class in Python

What can I help you build?

+ Gemini 2.5 Flash ▶

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)

✓ 12:16 AM Python 3

ASSIGNMENT-6.py

Al-assisted class generation for stude.py

```
1 # Generate the code to check whether the given string is palindrome
2
3 def is_palindrome(s):
4     # Remove spaces and convert to lowercase
5     cleaned = ''.join(s.split()).lower()
6     # Check if the cleaned string is equal to its reverse
7     return cleaned == cleaned[::-1]
8
9 # Test the function
10 test_string = input("Enter a string: ")
11 if is_palindrome(test_string):
12     print("The string is a palindrome.")
13 else:
14     print("The string is not a palindrome.")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE

```
/usr/local/bin/python3 "/Users/ronithreddyrevuri/Desktop/AI Asstd Coding/ASSIGNMENT-6.py" # AI-assisted class generation for stude.py"
● ronithreddyrevuri@Roniths-MacBook-Air ASSIGNMENT-6.py % /usr/local/bin/python3 "/Users/ronithreddyrevuri/Desktop/AI
Asstd Coding/ASSIGNMENT-6.py" # AI-assisted class generation for stude.py"
Enter a string: MALAYALAM
The string is a palindrome.
↳ ronithreddyrevuri@Roniths-MacBook-Air ASSIGNMENT-6.py %
```

zsh + ...

< OUTLINE > TIMELINE

0 ▲ 0 Live Share

Ln 15, Col 1 Spaces: 4 UTF-8 LF {} Python Python 3.13.9

EXPLANATION :

COPilot created a simple code to check palindrome function which is more easy to be applied .

Where as in Gemini created a code with more descriptive method with more detailed and structured way

OUTPUT :

```
RONITHREDDYREVURI@RONITHS-MACBOOK-AIR ASSIGNMENT-6.PY % /USR/LOCAL/BIN/  
PYTHON3 "/USERS/RONITHREDDYREVURI/DESKTOP/AI  
ASSTD CODING/ASSIGNMENT-6.PY/# AI-ASSISTED CLASS GENERATION FOR STUDE.PY"  
ENTER A STRING: MALAYALAM  
THE STRING IS A PALINDROME. (VS CODE )
```

ENTER A WORD OR PHRASE TO CHECK IF IT'S A PALINDROME: MALAYALAM
'MALAYALAM' IS A PALINDROME! (GOOGLE COLAB)

OBSERVATION :

COPilot generated code faster suitable for quick usage

GEMINI GENERATED CODE IN DETAILED FORMAT FOR LEARNIING QUICKLY AND UNDERSTANDABLE.

TASK 4 : CODE EXPLANATION USING AI

PROMPT :

#EXPLAIN PALINDROME CHECK FUNCTION EACH LINE

The screenshot shows a Google Colab notebook titled "AI&AC_ASSIGNMENT_2". In cell [11], there is a Python function named `is_palindrome` that takes a string `user_word` and prints whether it is a palindrome or not. The function uses `lower()` to convert the word to lowercase and `join()` to remove spaces and punctuation. The AI-generated explanation for this part states that `char.lower()` converts each alphanumeric character to lowercase, making the check case-insensitive, and `(''.join(...))` joins all processed characters back together into a new string `cleaned_word`. It provides examples like 'A man, a plan, a canal: Panama' becoming 'amanaplanacanalpanama'.

```
if is_palindrome(user_word):  
    print(f'{user_word} is a palindrome')  
else:  
    print(f'{user_word} is not a palindrome.')  
  
print("\n--- More Examples ---")  
print("Racecar" is a palindrome: {is_palindrome('Racecar')}")  
print("madam" is a palindrome: {is_palindrome('madam')}")  
print("A man, a plan, a canal: Panama" is a palindrome: {is_palindrome('A man, a plan, a canal: Panama')}")  
print("hello" is a palindrome: {is_palindrome('hello')}")  
print("Python" is a palindrome: {is_palindrome('Python')}")
```

... Enter a word or phrase to check if it's a palindrome!
'MALAYALAM' is a palindrome!

--- More Examples ---
'Racecar' is a palindrome: True
'madam' is a palindrome: True
'A man, a plan, a canal: Panama' is a palindrome: True
'hello' is a palindrome: False
'Python' is a palindrome: False

Start coding or generate with AI.

Gemini

effectively filters out spaces, punctuation, and symbols.

- `char.lower()`: For each alphanumeric character, it converts it to lowercase. This makes the check case-insensitive (e.g., 'Racecar' and 'racecar' are treated the same).
- `(''.join(...))`: Finally, all the processed (lowercase, alphanumeric) characters are joined back together to form a new string called `cleaned_word`. For example, 'A man, a plan, a canal: Panama' becomes 'amanaplanacanalpanama'.

4. # Compare the cleaned word with its reverse

- Another comment explaining the next logical step.

5. `return cleaned_word == cleaned_word[::-1]`

- This line performs the actual palindrome check and returns the result.
 - `cleaned_word[::-1]`: This is a Python slice that creates a reversed version of the `cleaned_word`. For example, if `cleaned_word` is 'madam', `cleaned_word[::-1]` will be 'madam'. If `cleaned_word` is 'hello', `cleaned_word[::-1]` will be 'olleh'.
 - `cleaned_word == ...`: It compares the original `cleaned_word` with its reversed version. If they are identical, the expression evaluates to `True` (it's a palindrome); otherwise, it's `False`.

▪ `return`: The `True` or `False` result is returned by the function.

Test 'is_palindrome' with a new phrase Suggest an edge case for the palindrome function
Show me how to use the string analysis function again

What can I help you build?

Gemini

Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)

Automatic saving failed. This file was updated remotely or in another tab. [Show diff](#)

12:16 AM Python 3

AI&AC_ASSIGNMENT_2

```

if is_palindrome(user_word):
    print(f'{user_word} is a palindrome.')
else:
    print(f'{user_word} is not a palindrome.')

print("\n--- More Examples ---")
print("Racecar" is a palindrome: {is_palindrome('Racecar')})
print("madam" is a palindrome: {is_palindrome('madam')})
print("A man, a plan, a canal: Panama" is a palindrome: {is_palindrome('A man, a plan, a canal: Panama')})
print("hello" is a palindrome: {is_palindrome('hello')})
print("Python" is a palindrome: {is_palindrome('Python')})

```

... Enter a word or phrase to check if it's a palindrome!
'MALAYALAM' is a palindrome!

--- More Examples ---
'Racecar' is a palindrome: True
'madam' is a palindrome: True
'A man, a plan, a canal: Panama' is a palindrome:
'hello' is a palindrome: False
'Python' is a palindrome: False

Start coding or generate with AI.

Gemini

- def is_palindrome(word):**

- This line defines a function named `is_palindrome` that takes one argument, `word` (which is expected to be a string).

- # Convert the word to lowercase and remove spaces for accurate checking**

- This is a comment explaining the purpose of the next line.

- cleaned_word = ''.join(char.lower() for char in word if char.isalnum())**

- This is the core of the cleaning process.
- for char in word: It iterates through each character in the input `word`.
- if char.isalnum(): It checks if the character is alphanumeric (i.e., a letter or a number). This effectively filters out spaces, punctuation, and symbols.
- char.lower(): For each alphanumeric character, it converts it to lowercase. This makes the check case-insensitive (e.g., 'Racecar' and 'racecar' are treated the same).
- '''.join(...)' : Finally, all the processed (lowercase, alphanumeric) characters are joined back together to form a new string called `cleaned_word`. For example, 'A man, a plan, a canal: Panama' becomes 'amanaplanacanalpanama'.

 - # Compare the cleaned word with its reverse**

Another comment explaining the next logical step

Test 'is_palindrome' with a new phrase | Suggest an edge case for the palindrome function | Show me how to use the string analysis function again

What can I help you build?

+ Gemini 2.5 Flash ▶

Automatic saving failed. This file was updated remotely or in another tab. Show diff

12:16 AM Python 3

AI&AC_ASSIGNMENT_2

```

if is_palindrome(user_word):
    print(f'{user_word} is a palindrome.')
else:
    print(f'{user_word} is not a palindrome.')

print("\n--- More Examples ---")
print("Racecar" is a palindrome: {is_palindrome('Racecar')})
print("madam" is a palindrome: {is_palindrome('madam')})
print("A man, a plan, a canal: Panama" is a palindrome: {is_palindrome('A man, a plan, a canal: Panama')})
print("hello" is a palindrome: {is_palindrome('hello')})
print("Python" is a palindrome: {is_palindrome('Python')})

```

... Enter a word or phrase to check if it's a palindrome!
'MALAYALAM' is a palindrome!

--- More Examples ---
'Racecar' is a palindrome: True
'madam' is a palindrome: True
'A man, a plan, a canal: Panama' is a palindrome:
'hello' is a palindrome: False
'Python' is a palindrome: False

Start coding or generate with AI.

Gemini

R EXPLAIN palindrome check function each line

◆ Certainly! Let's break down the `is_palindrome` function line by line:

R Explain this palindrome checking function line by line

◆ Let's go through the `is_palindrome` function line by line:

R Continue explaining the 'is_palindrome' function line by line

◆ Okay, let's break down the `is_palindrome` function step by step:

```

def is_palindrome(word):
    # Convert the word to lowercase and remove spaces for accurate checking
    cleaned_word = ''.join(char.lower() for char in word if char.isalnum())
    # Compare the cleaned word with its reverse
    return cleaned_word == cleaned_word[::-1]

```

Test 'is_palindrome' with a new phrase | Suggest an edge case for the palindrome function | Show me how to use the string analysis function again

What can I help you build?

+ Gemini 2.5 Flash ▶

Automatic saving failed. This file was updated remotely or in another tab. Show diff

12:16 AM Python 3

EXPLANATION:

GEMINI EXPLAINED THE CODE IN A SIMPLER WAY OF EACH AND EVERY LINE OF THE PALINDROME FUNCTION , SO THAT IT MAKES UNDERSTANDING THE CODE SIMPLER AND EASIER

OUTPUT :

IT HELPS IN UNDERSTANDING THE OUTPUT IS PALINDROME OR NOT

OBSERVATION :

GEMINI HELPED IN UNDERSTANDING THE CODE OF EACH AND EVERY STEP WHERE IT CAN HELP TO QUICK UNDERSTANDING OF CODE .

CONCLUSION :

TOOLS LIKE VS CODE AND CEMINI HELPS IN UNDERSTANDING THE CODE QUICKLY , USING THE CODE QUICKLY .