

# AI ASSISTED CODING LAB 4.2

TOPIC: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques.

ENROLLMENT NO: 2403A52006

NAME: MOHAMMED MUSTAFA

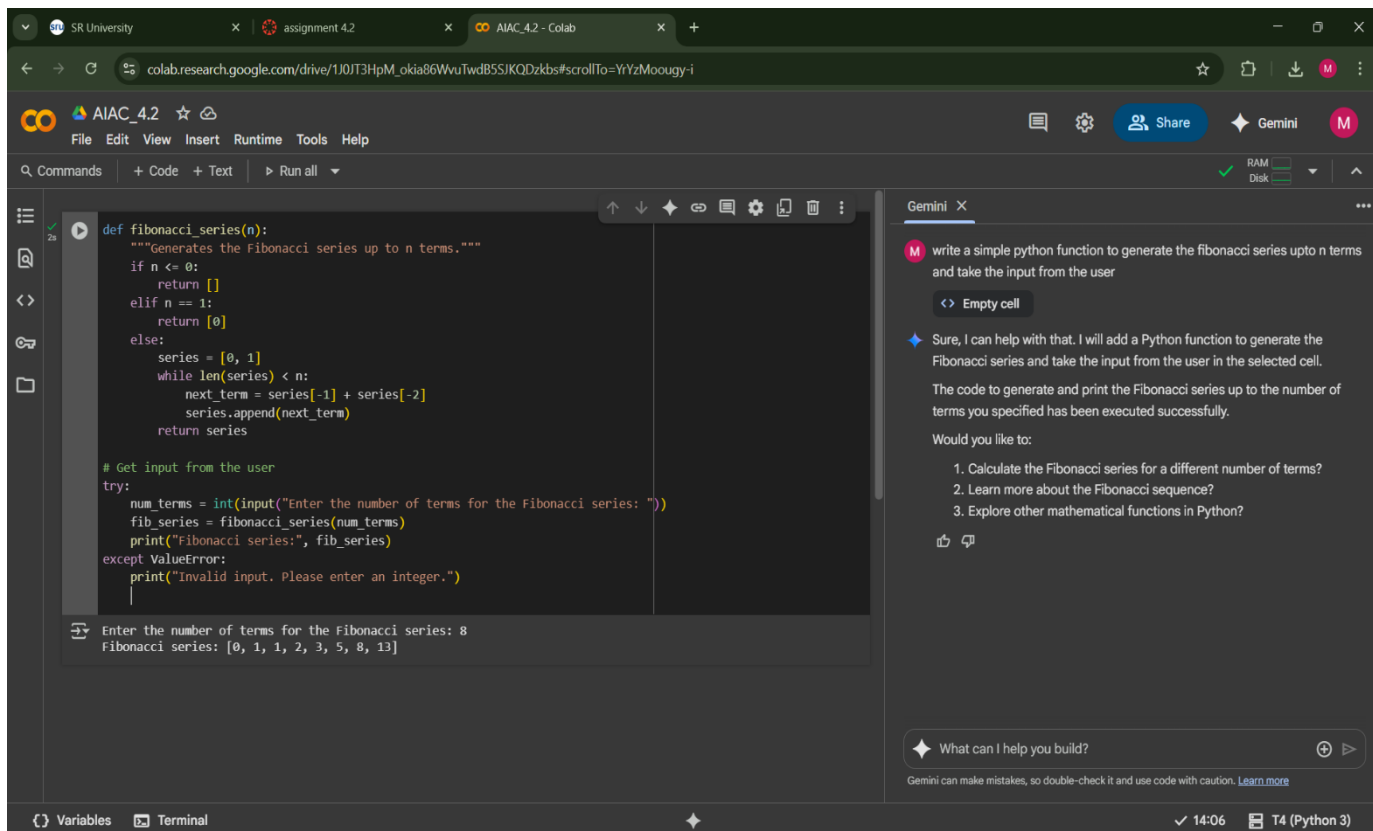
COURSE: CSM(AI/ML)

BATCH: 01

DATE: 26-08-2025

TASK 1: Write a Python function to generate the Fibonacci sequence up to n terms

USED PROMPT: Write a simple python function to generate the Fibonacci series up to n terms and take the input from the use



The screenshot displays a Google Colab notebook environment. The main code cell contains a Python function `fibonacci_series(n)` that generates the Fibonacci series up to `n` terms. The function includes a docstring, handles edge cases for `n <= 0` and `n == 1`, and uses a `while` loop to calculate the series. It also includes a user input section with a `try` block to prompt the user for the number of terms and a `ValueError` exception handler for invalid input. The output of the code cell shows the user entering 8 and the resulting Fibonacci series: `[0, 1, 1, 2, 3, 5, 8, 13]`. On the right side, the Gemini chat interface is visible, showing a prompt: "write a simple python function to generate the fibonacci series upto n terms and take the input from the user". The chat response confirms the function's execution and provides suggestions for further exploration.

```
def fibonacci_series(n):  
    """Generates the Fibonacci series up to n terms."""  
    if n <= 0:  
        return []  
    elif n == 1:  
        return [0]  
    else:  
        series = [0, 1]  
        while len(series) < n:  
            next_term = series[-1] + series[-2]  
            series.append(next_term)  
        return series  
  
    # Get input from the user  
    try:  
        num_terms = int(input("Enter the number of terms for the Fibonacci series: "))  
        fib_series = fibonacci_series(num_terms)  
        print("Fibonacci series:", fib_series)  
    except ValueError:  
        print("Invalid input. Please enter an integer.")
```

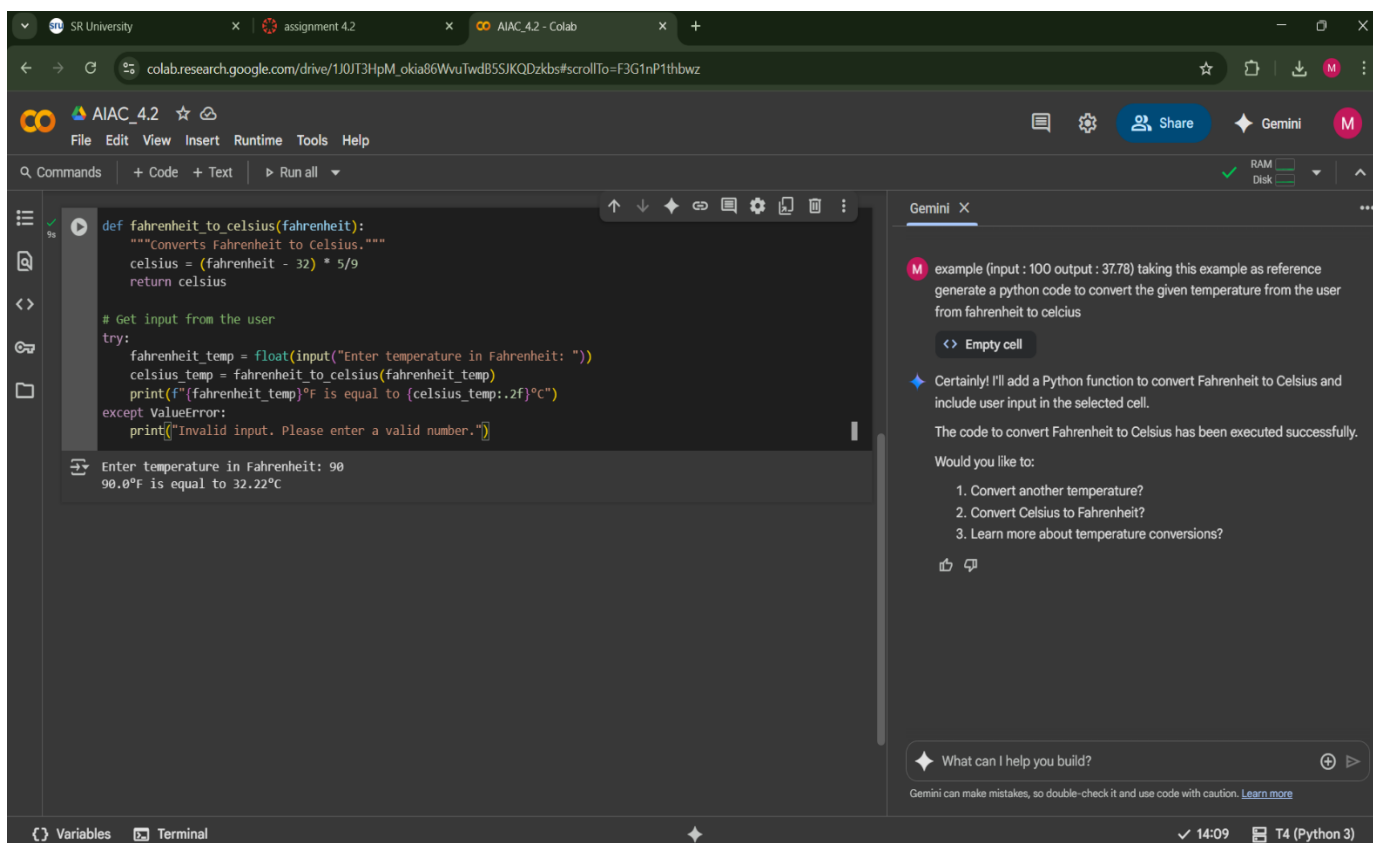
Enter the number of terms for the Fibonacci series: 8  
Fibonacci series: [0, 1, 1, 2, 3, 5, 8, 13]

Gemini X  
write a simple python function to generate the fibonacci series upto n terms and take the input from the user  
Empty cell  
Sure, I can help with that. I will add a Python function to generate the Fibonacci series and take the input from the user in the selected cell. The code to generate and print the Fibonacci series up to the number of terms you specified has been executed successfully. Would you like to:  
1. Calculate the Fibonacci series for a different number of terms?  
2. Learn more about the Fibonacci sequence?  
3. Explore other mathematical functions in Python?  
What can I help you build?  
Gemini can make mistakes, so double-check it and use code with caution. [Learn more](#)

EXPLANATION: This is a zero-prompt technique without giving any detailed information to AI and letting it print the code based on the prompt only

TASK 2: Provide one example: Input: 100,  
Output: 37.78 to help AI generate a function that converts Fahrenheit to Celsius

USED PROMPT: example (input: 100 , output : 37.78) taking this example as reference generate a python code to convert the given temperature from the user from Fahrenheit to Celsius



The screenshot displays a Google Colab notebook environment. The main code cell contains a Python function `fahrenheit_to_celsius` that takes a Fahrenheit temperature as input and returns the equivalent Celsius temperature. The function uses the formula  $C = (F - 32) \times \frac{5}{9}$ . Below the function definition, there is a try-except block that prompts the user to enter a temperature in Fahrenheit, calls the function, and prints the result. The output shows that 90°F is equal to 32.22°C.

```
def fahrenheit_to_celsius(fahrenheit):  
    """Converts Fahrenheit to Celsius."""  
    celsius = (fahrenheit - 32) * 5/9  
    return celsius  
  
# Get input from the user  
try:  
    fahrenheit_temp = float(input("Enter temperature in Fahrenheit: "))  
    celsius_temp = fahrenheit_to_celsius(fahrenheit_temp)  
    print(f"{fahrenheit_temp}°F is equal to {celsius_temp:.2f}°C")  
except ValueError:  
    print("Invalid input. Please enter a valid number.")
```

The output of the code execution is:

```
Enter temperature in Fahrenheit: 90  
90.0°F is equal to 32.22°C
```

On the right side of the notebook, the Gemini chat interface is visible. It shows a prompt: "example (input : 100 output : 37.78) taking this example as reference generate a python code to convert the given temperature from the user from fahrenheit to celcius". The Gemini response confirms that it will add a Python function to convert Fahrenheit to Celsius and include user input in the selected cell. It also states that the code to convert Fahrenheit to Celsius has been executed successfully and offers suggestions for further actions: "1. Convert another temperature?", "2. Convert Celsius to Fahrenheit?", and "3. Learn more about temperature conversions?".

EXPLANATION: This is a one shot-prompt technique. here we give the AI one example to explain it in what format we want the code to be printed

TASK 3: Give 2–3 examples to create a function that extracts the domain name from an email address.

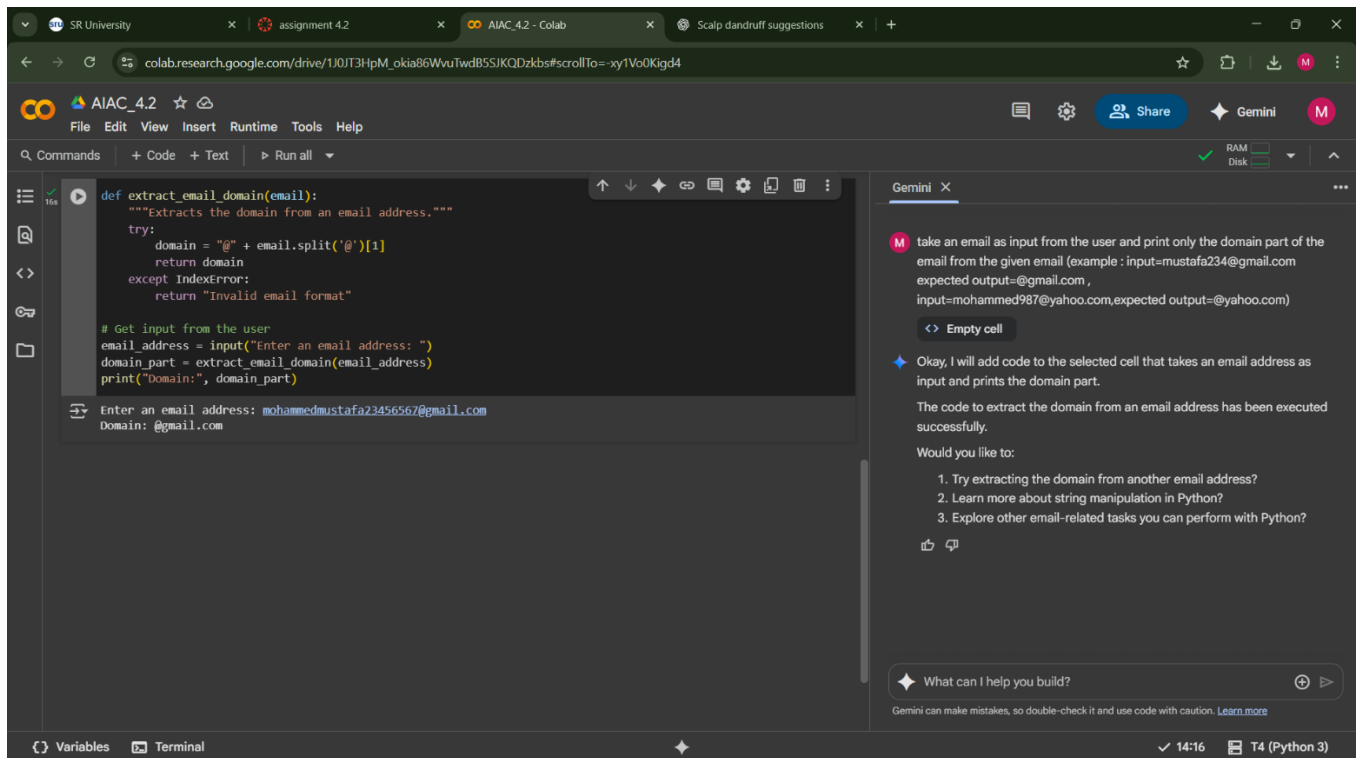
USED PROMPT: Take an email as input from the user and print only the domain part of the email from the given email (example :

[input=mustafa234@gmail.com](#) , expected

[output=@gmail.com](#) ,

[input=mohammed987@yahoo.com](#) , expected

output=@yahoo.com)



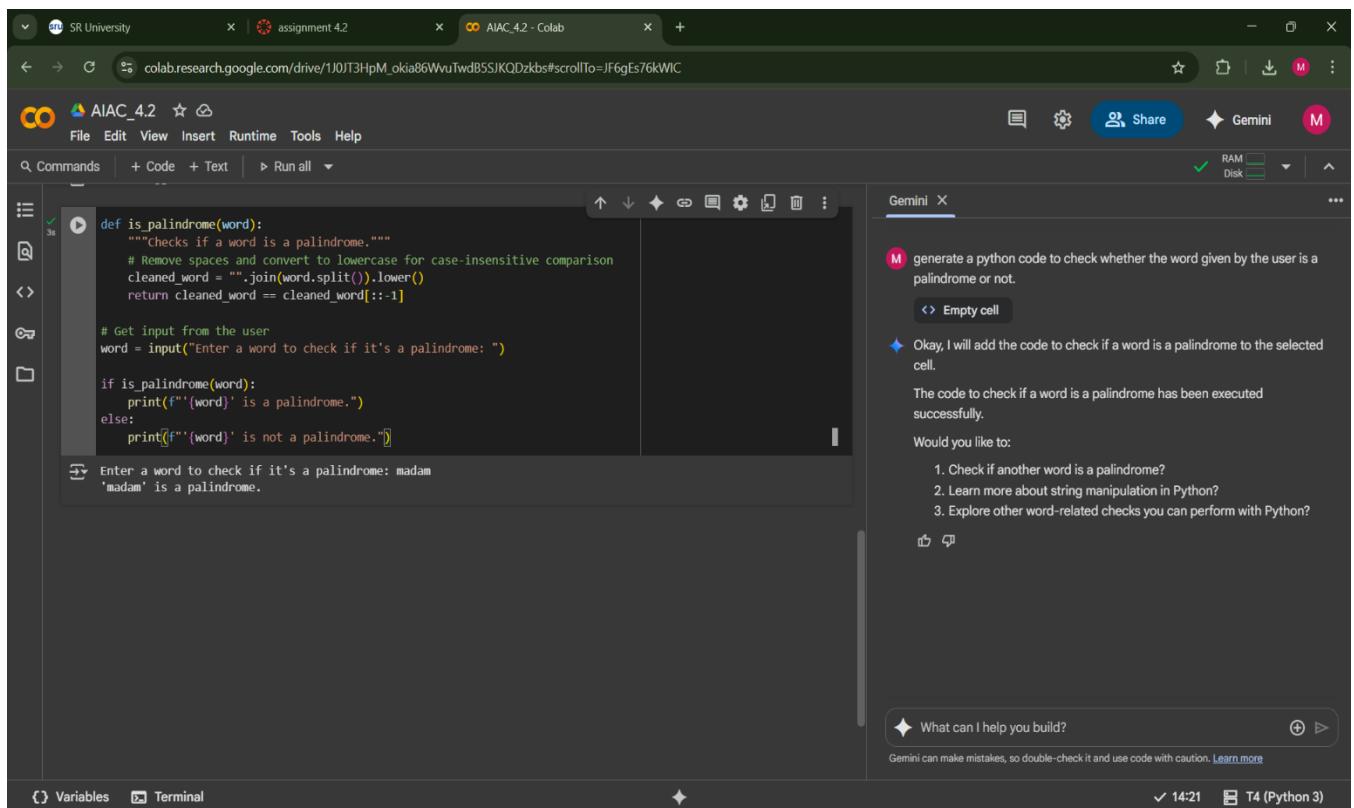
EXPLANATION: this is a few-shot technique.

Here we give AI 2-3 examples of how we expect the code to be and give various conditions to be implemented in the code.

TASK 4: Compare zero-shot vs few-shot prompting for generating a function that checks whether a word is a palindrome, ignoring punctuation and case.

USED PROMPT: **ZERO-SHOT TECHNIQUE:**

GENERATE A PYTHON CODE TO CHECK  
WHETHER THE WORD GIVEN BY THE USER IS A  
PALINDROME OR NOT



The screenshot displays a Google Colab notebook environment. The notebook contains a Python function `is_palindrome` that takes a word as input, cleans it by removing spaces and converting to lowercase, and then checks if it is a palindrome by comparing the word with its reverse. The function is called with the input "madam", and the output is "madam is a palindrome.".

```
def is_palindrome(word):  
    """Checks if a word is a palindrome."""  
    # Remove spaces and convert to lowercase for case-insensitive comparison  
    cleaned_word = "".join(word.split()).lower()  
    return cleaned_word == cleaned_word[::-1]  
  
# Get input from the user  
word = input("Enter a word to check if it's a palindrome: ")  
  
if is_palindrome(word):  
    print(f'{word} is a palindrome.')  
else:  
    print(f'{word} is not a palindrome.')
```

The Gemini AI chat interface on the right shows the prompt: "generate a python code to check whether the word given by the user is a palindrome or not." The AI responds with a confirmation and a list of suggestions for further exploration.

generate a python code to check whether the word given by the user is a palindrome or not.

Empty cell

Okay, I will add the code to check if a word is a palindrome to the selected cell.

The code to check if a word is a palindrome has been executed successfully.

Would you like to:

1. Check if another word is a palindrome?
2. Learn more about string manipulation in Python?
3. Explore other word-related checks you can perform with Python?

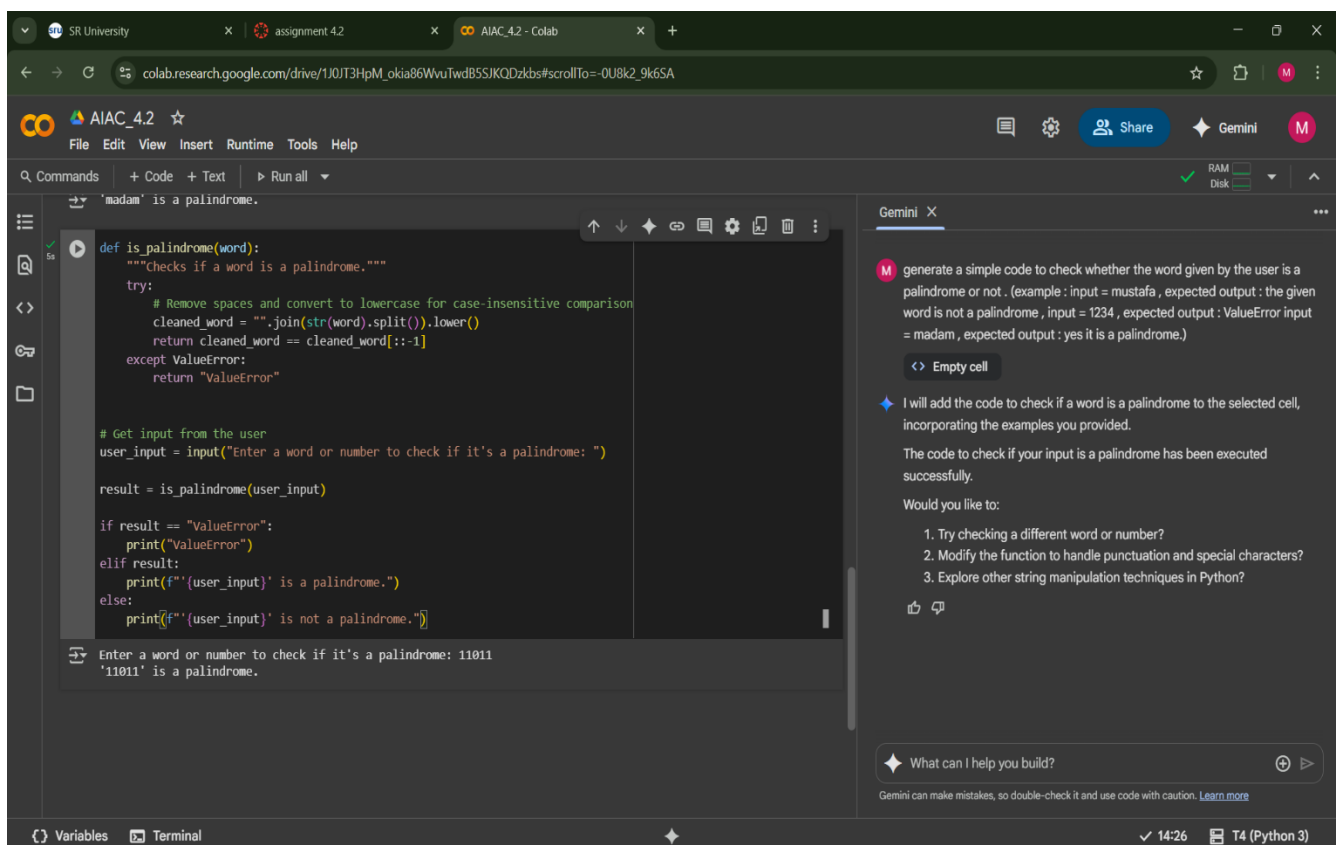
What can I help you build?

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

EXPLANATION: THIS IS JUST A ZERO-SHOT  
TECHNIQUE, ASKING AI TO GENERATE A  
CODE TO CHECK WHETHER A WORD IS A  
PALINDROME OR NOT WITHOUT ANY  
CONDITIONS USED IN THE PROMPT.

FEW-SHOT TECHNIQUE:

generate a simple code to check whether the word given by the user is a palindrome or not .  
(example : input = mustafa , expected output : the given word is not a palindrome , input = 1234 , expected output : ValueError input = madam , expected output : yes itis a palindrome)



The screenshot displays a Google Colab notebook environment. The top toolbar includes icons for file operations, a search bar, and a 'Run all' button. The notebook's code cell contains a Python function `is_palindrome` that takes a word as input, cleans it by removing spaces and converting to lowercase, and then checks if it is a palindrome. The function uses `cleaned_word[::-1]` to reverse the string. It includes a `try` block to handle `ValueError` exceptions. Below the function, there is a section for user input and output handling. The output cell shows the result of running the code with the input '11011', which is '11011' is a palindrome.

```
def is_palindrome(word):  
    """Checks if a word is a palindrome."""  
    try:  
        # Remove spaces and convert to lowercase for case-insensitive comparison  
        cleaned_word = "".join(str(word).split()).lower()  
        return cleaned_word == cleaned_word[::-1]  
    except ValueError:  
        return "ValueError"  
  
# Get input from the user  
user_input = input("Enter a word or number to check if it's a palindrome: ")  
  
result = is_palindrome(user_input)  
  
if result == "ValueError":  
    print("ValueError")  
elif result:  
    print(f'{user_input} is a palindrome.')  
else:  
    print(f'{user_input} is not a palindrome.')
```

Enter a word or number to check if it's a palindrome: 11011  
'11011' is a palindrome.

The Gemini chat interface on the right shows a user prompt asking for a simple code to check for palindromes, with examples provided. The Gemini response indicates it will add the code to the selected cell and provides a list of suggestions for further exploration.

generate a simple code to check whether the word given by the user is a palindrome or not . (example : input = mustafa , expected output : the given word is not a palindrome , input = 1234 , expected output : ValueError input = madam , expected output : yes itis a palindrome.)

Empty cell

I will add the code to check if a word is a palindrome to the selected cell, incorporating the examples you provided.

The code to check if your input is a palindrome has been executed successfully.

Would you like to:

1. Try checking a different word or number?
2. Modify the function to handle punctuation and special characters?
3. Explore other string manipulation techniques in Python?

What can I help you build?

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

EXPLANATION: COMPARED TO THE ZERO-SHOT TECHNIQUE, THE FEW SHOT TECHNIQUE CHECKS

FOR MULTIPLE CONDITIONS/INPUTS. THE USER CHECKS ALL THE CONDITIONS HE WANTS TO CHECK IN THE PROMPT AND ASKS AI TO GENERATE THE CODE FOLLOWING THE CONDITIONS INCLUDED IN THE PROMPT.

TASK 5: Use few-shot prompting with 3 sample inputs to generate a function that determines the maximum of three numbers without using the built-in `max()` function.

USED PROMPT: generate a python code. take 3 numbers input from user and print the largest among them use the `max()` in-built keyword . if the user enters a string raise an



error to input integers example = input :  
2,5,6 , expected output : 6 is the largest input  
: 5 , 5 , 7 , expected output : please enter  
three different numbers input : 5 , 5 , 5 ,  
expected output : please enter three different  
numbers

The screenshot shows a Google Colab notebook titled "AIAC\_4.2". The code defines a function `find_largest_of_three()` that takes three numbers as input and prints the largest among them. It includes error handling for non-integer inputs and for cases where two or more numbers are equal. The function is called with inputs 5, 5, and 9, resulting in the output "9 is the largest".

```
def find_largest_of_three():  
    """Takes three numbers as input and prints the largest among them."""  
    try:  
        num1 = int(input("Enter the first number: "))  
        num2 = int(input("Enter the second number: "))  
        num3 = int(input("Enter the third number: "))  
  
        if num1 == num2 or num2 == num3 or num1 == num3:  
            print("Please enter three different numbers.")  
        else:  
            largest = max(num1, num2, num3)  
            print(f"{largest} is the largest")  
    except ValueError:  
        print("Error: Please input integers.")  
  
    # Call the function to get input and find the largest number  
    find_largest_of_three()
```

The output of the code execution is:

```
Enter the first number: 5  
Enter the second number: 5  
Enter the third number: 9  
9 is the largest
```

The Gemini chat interface on the right shows a prompt asking for a Python code to find the largest of three numbers, with examples of expected behavior for different inputs. The chat response provides a solution and suggests further exploration.

**EXPLANATION:** It is a few-shot technique used to give multiple shots/conditions in the prompt. Here I have used 2 different conditions the input shouldn't be a string and no 2 or 3 numbers should be same

