AI ASSISTED CODING

LAB ASSIGNMENT – 4.2

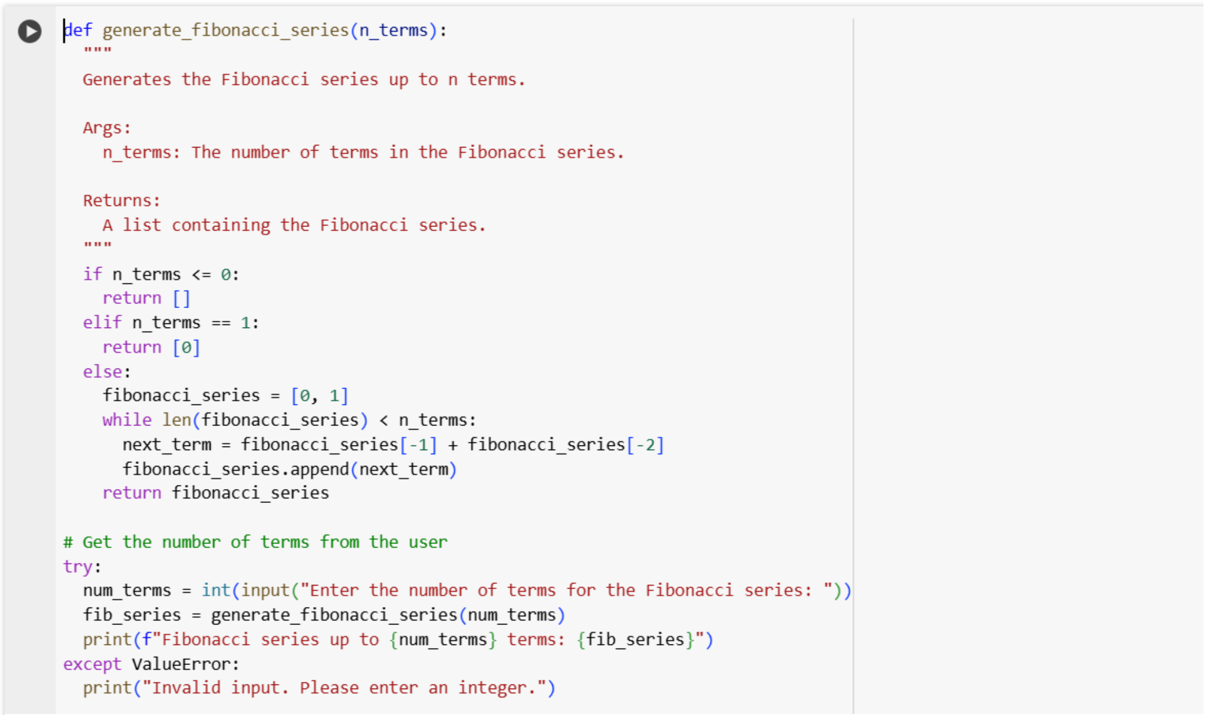
Program	: B-TECH
Specialization	: AIML
Name of Student	: AKULA MANOJ
Enrollment-No	: 2403A52031
Batch No	: 02
Date	: 26-08-2025

TASK DESCRIPTION 1:

ZERO-SHOT: PROMPT AI WITH ONLY THE INSTRUCTION — WRITE A PYTHON FUNCTION TO GENERATE THE FIBONACCI SEQUENCE UP TO N TERMS.

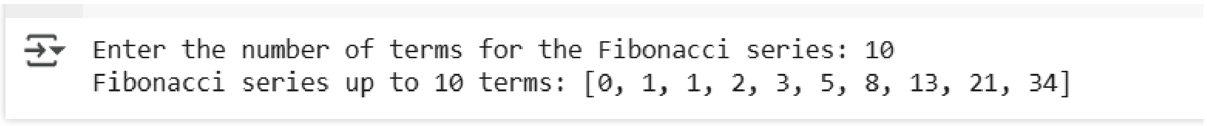
PROMPT: WRITE A PYTHON FUNCTION TO GENERATE THE FIBONACCI SERIES UP TO N TERMS INPUT SHOULD TAKE FROM USER. GIVE ME NEAT AND CLEAR CODE

GENERATED-CODE:



```python
def generate_fibonacci_series(n_terms):
  """
  Generates the Fibonacci series up to n terms.

  Args:
    n_terms: The number of terms in the Fibonacci series.

  Returns:
    A list containing the Fibonacci series.
  """
  if n_terms <= 0:
    return []
  elif n_terms == 1:
    return [0]
  else:
    fibonacci_series = [0, 1]
    while len(fibonacci_series) < n_terms:
      next_term = fibonacci_series[-1] + fibonacci_series[-2]
      fibonacci_series.append(next_term)
    return fibonacci_series

# Get the number of terms from the user
try:
  num_terms = int(input("Enter the number of terms for the Fibonacci series: "))
  fib_series = generate_fibonacci_series(num_terms)
  print(f"Fibonacci series up to {num_terms} terms: {fib_series}")
except ValueError:
  print("Invalid input. Please enter an integer.")
```

OUTPUT:



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

EXPLANATION:

This Python program generates the Fibonacci series up to a specified number of terms entered by the user. It defines a function called generate_fibonacci_series(n_terms) that returns a list containing the Fibonacci sequence. The function first checks if the number of terms is less than or equal to zero, returning an empty list. If the number of terms is one, it returns a list with only the first Fibonacci number [0]. Otherwise, it starts with the initial two Fibonacci numbers [0, 1] and uses a loop to calculate and append the next terms by summing the last two numbers in the sequence until the desired number of terms is reached. The main part of the program asks the user to input the number of terms, calls the function with that input, and prints the resulting Fibonacci series.

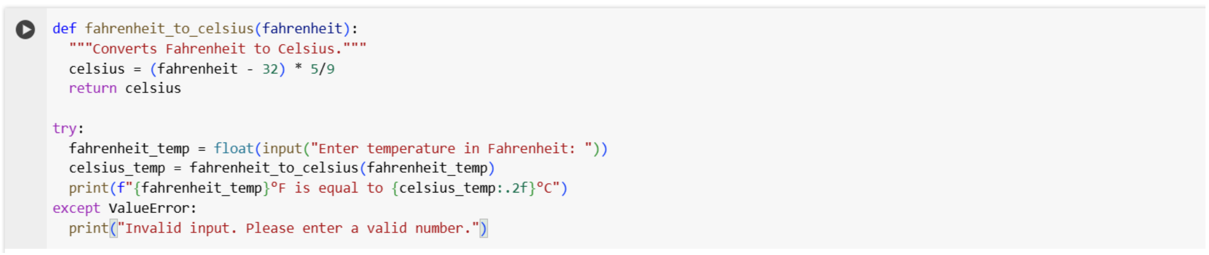
TASK DESCRIPTION 2:

ONE-SHOT: PROVIDE ONE EXAMPLE: INPUT: 100, OUTPUT: 37.78 TO HELP AI GENERATE A FUNCTION THAT CONVERTS FAHRENHEIT TO CELSIUS.

PROMPT: Generate a Python code that converts Fahrenheit to Celsius. Example input:100 and

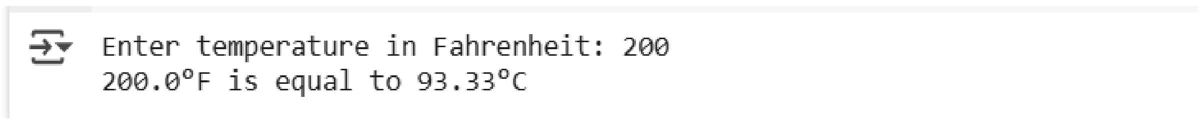
Output:37.78 and it should take input from the user.

CODE-GENERATED:



```python
def fahrenheit_to_celsius(fahrenheit):
  """Converts Fahrenheit to Celsius."""
  celsius = (fahrenheit - 32) * 5/9
  return celsius

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.")
```

OUTPUT:



```
Enter temperature in Fahrenheit: 200
200.0°F is equal to 93.33°C
```

EXPLANATION:

This Python program converts a temperature from Fahrenheit to Celsius. It defines a function called fahrenheit_to_celsius(fahrenheit) that uses the standard conversion formula: (fahrenheit - 32) * 5/9, and returns the result in Celsius. The main part of the program prompts the user to enter a temperature in Fahrenheit. The input is converted to a floating-point number and passed to the conversion function. The result is then printed, formatted to two decimal places, showing both the Fahrenheit and the equivalent

Celsius value. The program also includes error handling using a try-except block: if the user enters invalid input (e.g., text instead of a number), a message is displayed asking for a valid number.

TASK DESCRIPTION 3:

FEW-SHOT: GIVE 2–3 EXAMPLES TO CREATE A FUNCTION THAT EXTRACTS THE DOMAIN NAME FROM
AN EMAIL ADDRESS.

PROMPT: GENERATE A PYTHON CODE THAT EXTRACTS THE DOMAIN NAME FROM AN EMAIL ADDRESS.

EXAMPLES:

MANOJ@GMAIL.COM, THE DOMAIN NAME IS GMAIL.COM

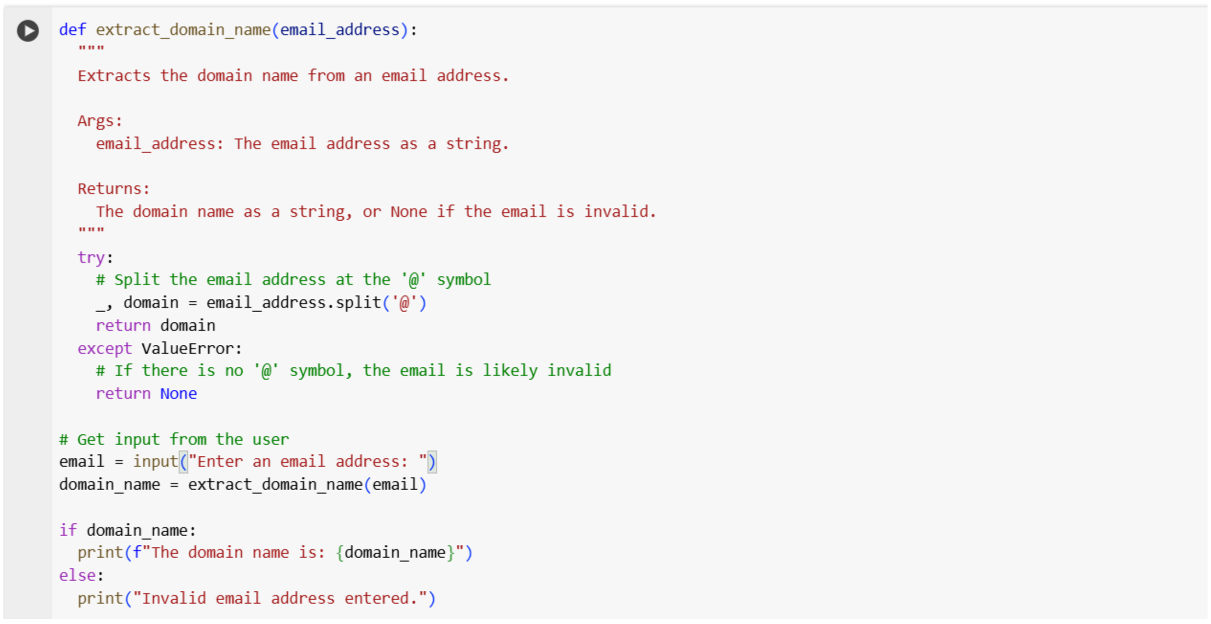
2403A52031@SRU.EDU.IN, THE DOMAIN NAME IS SRU.EDU.IN

LAXMAN@GMAIL.COM, THE DOMAIN NAME IS GMAIL.COM

CREATE CLEAN PYTHON CODE. TAKE USER INPUT FROM THE USER.

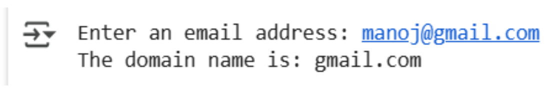
DON'T PROVIDE EXAMPLES IN THE OUTPUT

GENERATED CODE:



```python
def extract_domain_name(email_address):
  """
  Extracts the domain name from an email address.

  Args:
    email_address: The email address as a string.

  Returns:
    The domain name as a string, or None if the email is invalid.
  """
  try:
    # Split the email address at the '@' symbol
    _, domain = email_address.split('@')
    return domain
  except ValueError:
    # If there is no '@' symbol, the email is likely invalid
    return None

# Get input from the user
email = input("Enter an email address: ")
domain_name = extract_domain_name(email)

if domain_name:
  print(f"The domain name is: {domain_name}")
else:
  print("Invalid email address entered.")
```

OUTPUT:



```
Enter an email address: manoj@gmail.com
The domain name is: gmail.com
```

EXPLANATION:

This Python program extracts the domain name from an email address entered by the user. It defines a function called extract_domain_name(email_address) that attempts to split the input email at the '@' symbol. If the email is valid and contains the '@' character, the part after the '@' (i.e., the domain) is returned. If not, the function catches a ValueError and returns None, indicating an invalid email. In the main part of the program, the user is prompted to enter an email address.

TASK DESCRIPTION 4:

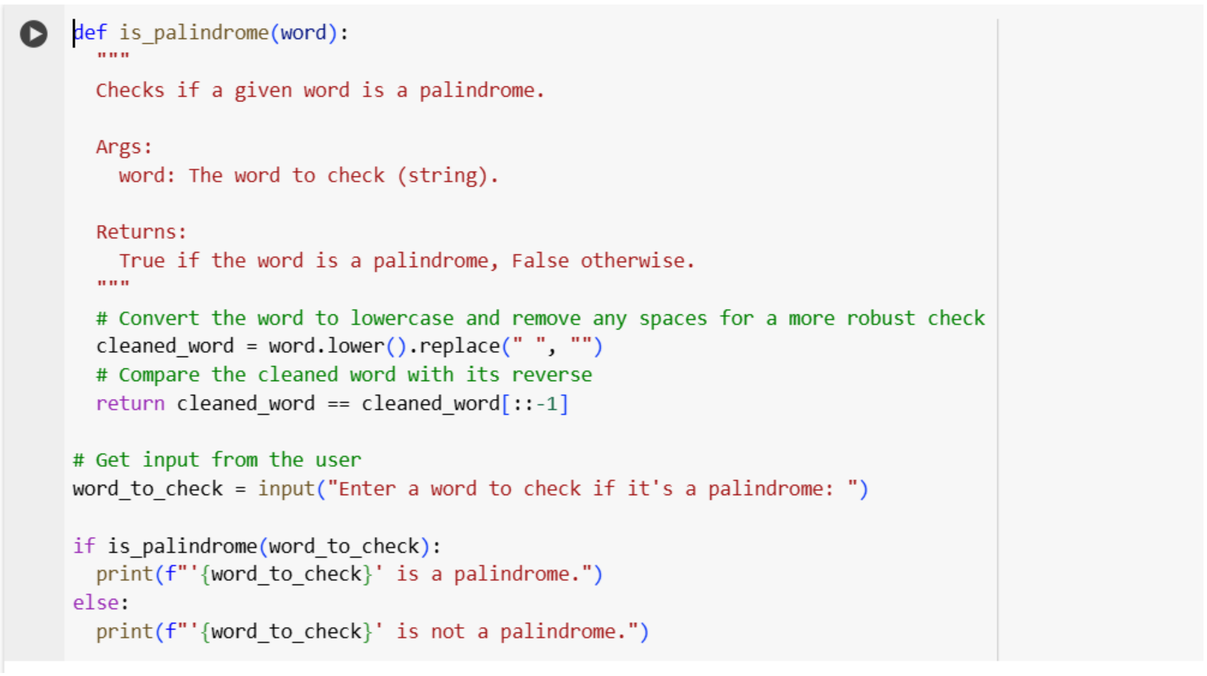
COMPARE ZERO-SHOT VS FEW-SHOT PROMPTING FOR GENERATING A FUNCTION THAT CHECKS

WHETHER A WORD IS A PALINDROME, IGNORING PUNCTUATION AND CASE.

ZERO SHOT:

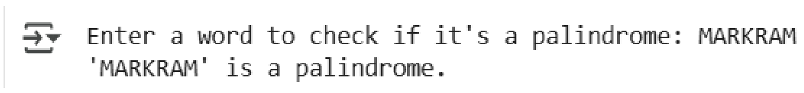
PROMPT: Generate a Python code that checks whether the given word is a palindrome or not. It should take the user

GENERATED CODE:

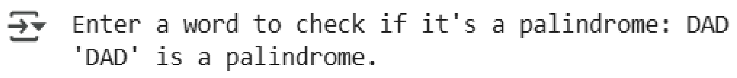


```python
def is_palindrome(word):
  """
  Checks if a given word is a palindrome.

  Args:
    word: The word to check (string).

  Returns:
    True if the word is a palindrome, False otherwise.
  """
  # Convert the word to lowercase and remove any spaces for a more robust check
  cleaned_word = word.lower().replace(" ", "")
  # Compare the cleaned word with its reverse
  return cleaned_word == cleaned_word[::-1]

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

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

OUTPUT:



```
Enter a word to check if it's a palindrome: MARKRAM
'MARKRAM' is a palindrome.
```

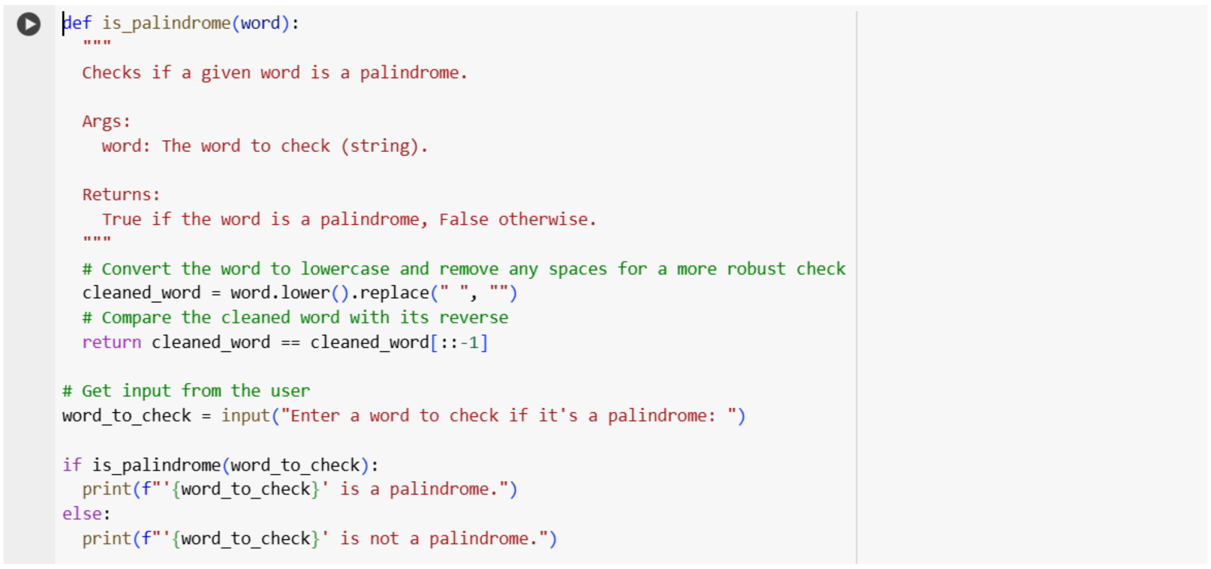


```
Enter a word to check if it's a palindrome: DAD
'DAD' is a palindrome.
```

FEW SHOT:

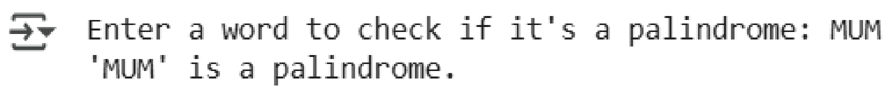
PROMPT:

generate a python code that checks whether the word is a palindrome or not, for example: markram, it is a palindrome, and malayalam is a palindrome, and it should take input from the user, and it should not display examples on the output. generate a clear and neat code.

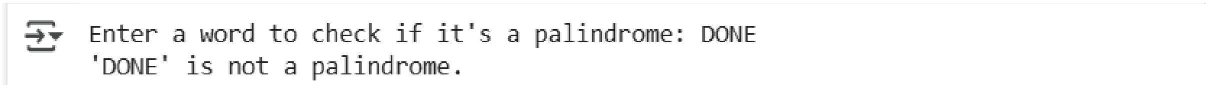


```
def is_palindrome(word):
  """
  Checks if a given word is a palindrome.

  Args:
    word: The word to check (string).

  Returns:
    True if the word is a palindrome, False otherwise.
  """
  # Convert the word to lowercase and remove any spaces for a more robust check
  cleaned_word = word.lower().replace(" ", "")
  # Compare the cleaned word with its reverse
  return cleaned_word == cleaned_word[::-1]

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

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

OUTPUT:



```
Enter a word to check if it's a palindrome: MUM
'MUM' is a palindrome.
```



```
Enter a word to check if it's a palindrome: DONE
'DONE' is not a palindrome.
```

Output-comparison:

The outputs of both zero-shot and one-shot have a slight difference. The code generated is more particular in the one-shot prompt, because we are giving and prompting with examples.

In Output, the is quietly similar because the prompt is slightly changed and added examples in one shot, but the output is similar and the accuracy is perfect. and without mentioning to ignore the CASES, it is already done with ignoring whether it is uppercase or lowercase, and the code is very good in one shot.

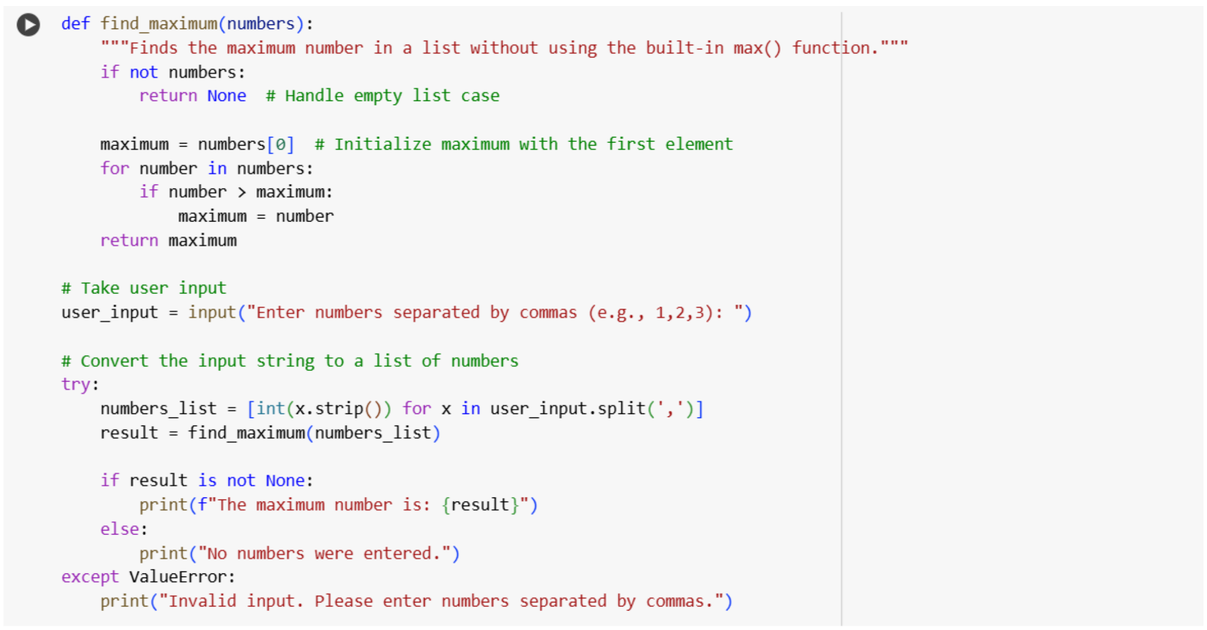
TASK DESCRIPTION 5:

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.

PROMPT:

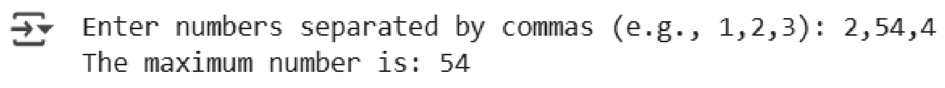
Generates a Python code, a maximum of 3 samples given by the user. Example in 1,2,3 the maximum number is 3, in 32,31,45 the maximum number is 45, in 2,4,3 the maximum number is 4. Generate a simple Python code from taking the given example, without using built-in max functions.without any built-in functions. and generate a code that takes user input from us, and do not show examples in the output after running the code.

GENERATED CODE:



```python
def find_maximum(numbers):
    """Finds the maximum number in a list without using the built-in max() function."""
    if not numbers:
        return None  # Handle empty list case

    maximum = numbers[0]  # Initialize maximum with the first element
    for number in numbers:
        if number > maximum:
            maximum = number
    return maximum

# Take user input
user_input = input("Enter numbers separated by commas (e.g., 1,2,3): ")

# Convert the input string to a list of numbers
try:
    numbers_list = [int(x.strip()) for x in user_input.split(',')]
    result = find_maximum(numbers_list)

    if result is not None:
        print(f"The maximum number is: {result}")
    else:
        print("No numbers were entered.")
except ValueError:
    print("Invalid input. Please enter numbers separated by commas.")
```

OUTPUT:



```
Enter numbers separated by commas (e.g., 1,2,3): 2,54,4
The maximum number is: 54
```

EXPLANATION: This Python program finds the maximum number in a list without using the built-in max() function. It defines a function called find_maximum(numbers) that takes a list of numbers and returns the largest one. If the list is empty, the function returns None. Otherwise, it initializes the maximum value as the first element of the list and then loops through the rest, updating the maximum whenever a larger number is found. The main part of the program asks the user to enter numbers separated by commas (e.g., 1, 2, 3). It then converts this input string into a list of integers and passes it to the function. If valid numbers were entered, it prints the maximum; otherwise, it notifies the user if no numbers were given or if the input was invalid

It was handled without the max() function by taking and analyzing the example patterns.

THANK YOU