LAB ASSIGNMENT -4.2

NAME: **J. Nithin Patil**

HT.NO: **2403A5029**

BATCH NO: **24BTCAIAIB02**

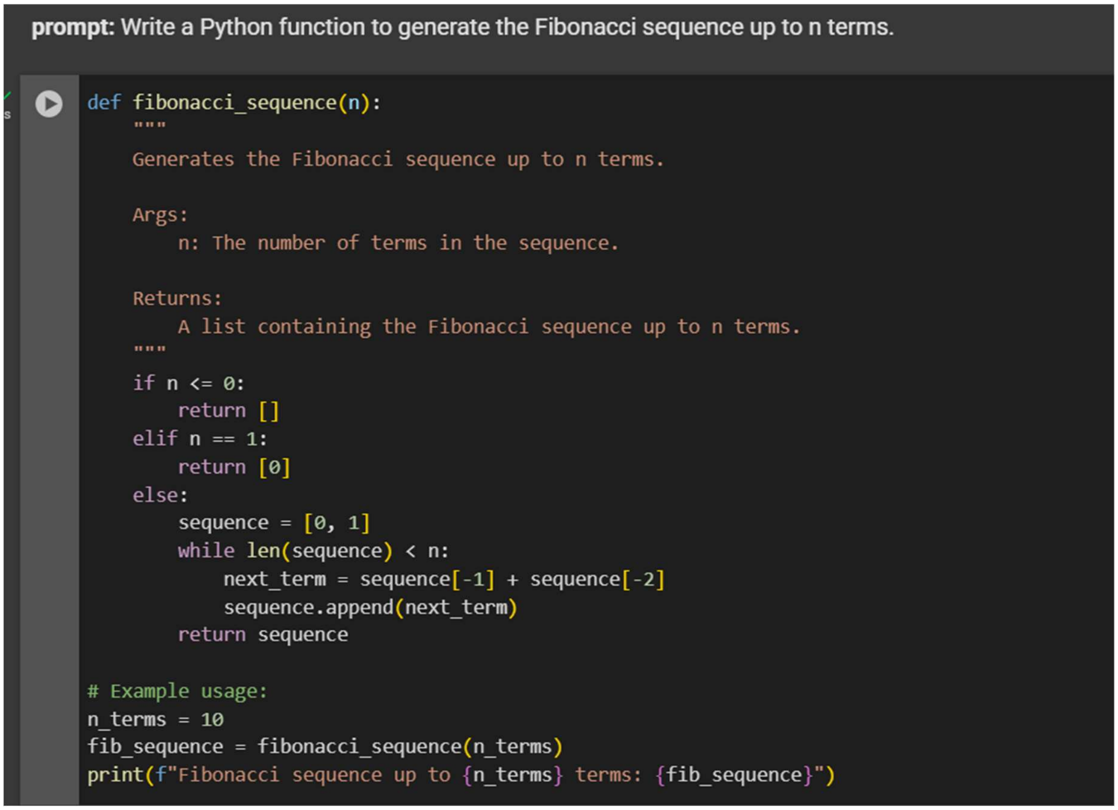
COURSE: **AI Assisted Coding**

Task Description#1

• Zero-shot: Prompt AI with only the instruction — Write a Python function to generate the Fibonacci sequence up to n terms

Expected Output#1

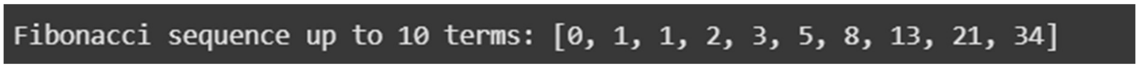
• A working function without using any sample inputs/outputs.



prompt: Write a Python function to generate the Fibonacci sequence up to n terms.

```python
def fibonacci_sequence(n):
    """
    Generates the Fibonacci sequence up to n terms.

    Args:
        n: The number of terms in the sequence.

    Returns:
        A list containing the Fibonacci sequence up to n terms.
    """
    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

# Example usage:
n_terms = 10
fib_sequence = fibonacci_sequence(n_terms)
print(f"Fibonacci sequence up to {n_terms} terms: {fib_sequence}")
```

Output:



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

Explanation:

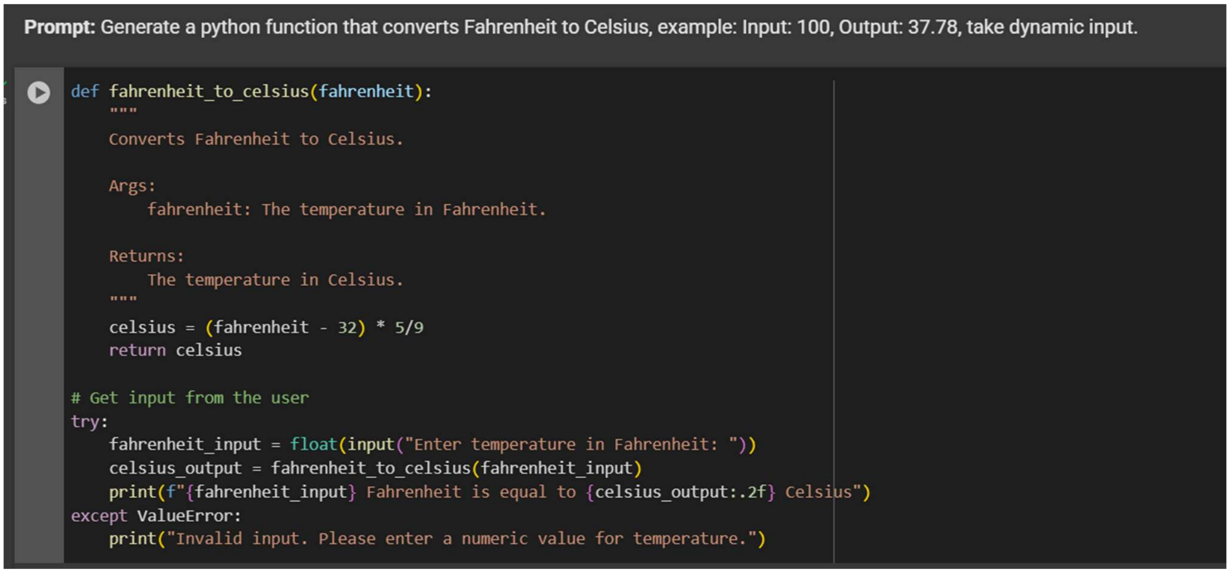
- No examples were given, only the instruction.
- The AI correctly generated a Fibonacci function.
- Uses a loop to generate the sequence.

Task Description#2

• One-shot: Provide one example: Input: 100, Output: 37.78 to help AI generate a function that converts Fahrenheit to Celsius.

Expected Output#2

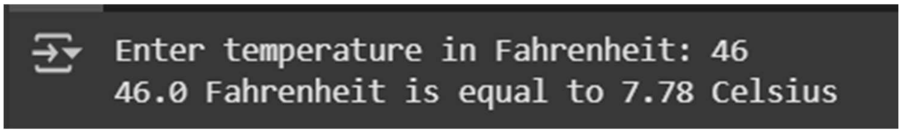
• A correct conversion function guided by the single example.



Prompt: Generate a python function that converts Fahrenheit to Celsius, example: Input: 100, Output: 37.78, take dynamic input.

```python
def fahrenheit_to_celsius(fahrenheit):
    """
    Converts Fahrenheit to Celsius.

    Args:
        fahrenheit: The temperature in Fahrenheit.

    Returns:
        The temperature in Celsius.
    """
    celsius = (fahrenheit - 32) * 5/9
    return celsius

# Get input from the user
try:
    fahrenheit_input = float(input("Enter temperature in Fahrenheit: "))
    celsius_output = fahrenheit_to_celsius(fahrenheit_input)
    print(f"{fahrenheit_input} Fahrenheit is equal to {celsius_output:.2f} Celsius")
except ValueError:
    print("Invalid input. Please enter a numeric value for temperature.")
```

Output:



```
Enter temperature in Fahrenheit: 46
46.0 Fahrenheit is equal to 7.78 Celsius
```

Explanation:

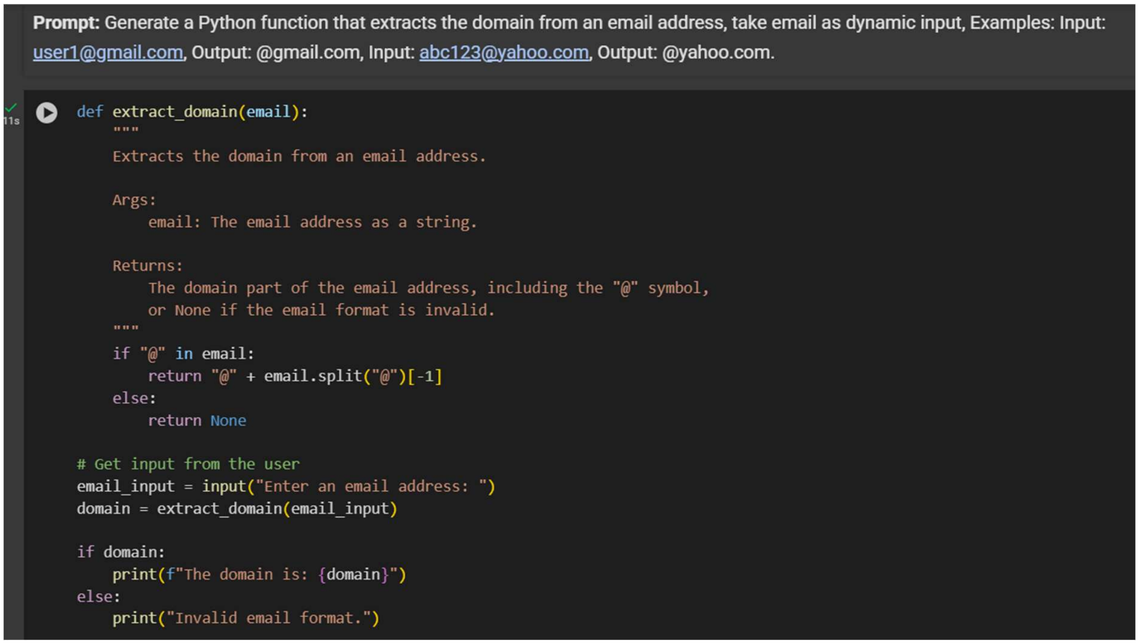
- **A single example helped the AI infer the formula.**
- **The function returns temperature in Celsius with 2 decimal points.**

Task Description#3

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

Expected Output#3

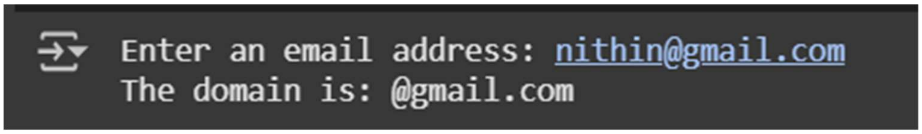
• Accurate function that returns only the domain portion of an email (e.g., @gmail.com)



Prompt: Generate a Python function that extracts the domain from an email address, take email as dynamic input, Examples: Input: user1@gmail.com, Output: @gmail.com, Input: abc123@yahoo.com, Output: @yahoo.com.

```
def extract_domain(email):
    """
    Extracts the domain from an email address.

    Args:
        email: The email address as a string.

    Returns:
        The domain part of the email address, including the "@" symbol,
        or None if the email format is invalid.
    """
    if "@" in email:
        return "@" + email.split("@")[-1]
    else:
        return None

# Get input from the user
email_input = input("Enter an email address: ")
domain = extract_domain(email_input)

if domain:
    print(f"The domain is: {domain}")
else:
    print("Invalid email format.")
```

Output:



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

Explanation:

- **Two examples help the AI identify the pattern.**
- **Function extracts substring from "@" to the end.**

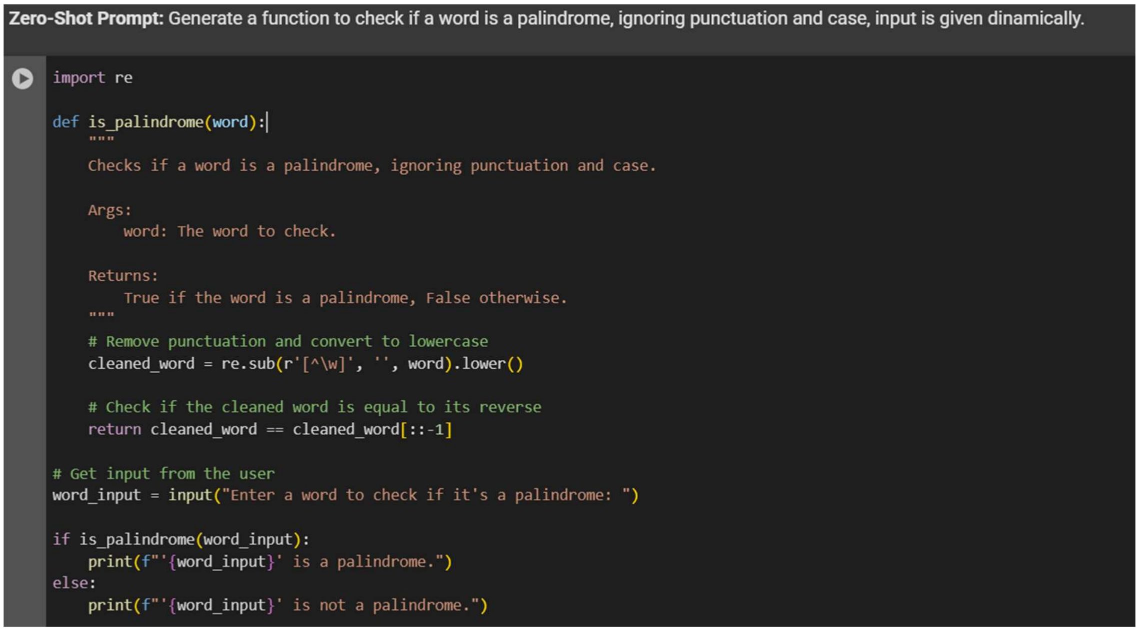
Task Description#4

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

Expected Output#4

- Output comparison + student explanation on how examples helped the model.

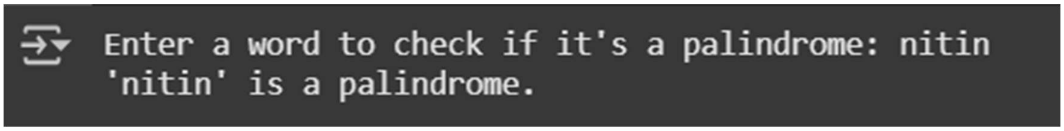
Zero-Shot Prompt:



Zero-Shot Prompt: Generate a function to check if a word is a palindrome, ignoring punctuation and case, input is given dinamically.

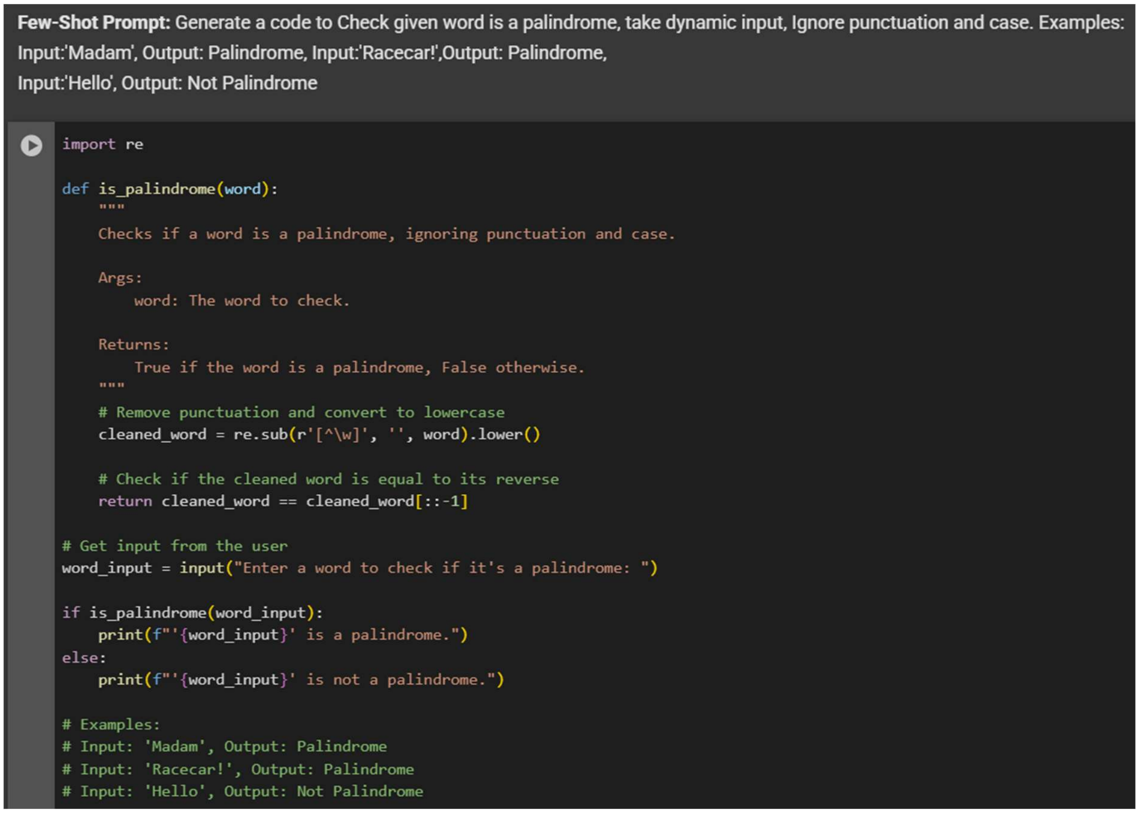
```python
import re

def is_palindrome(word):
    """
    Checks if a word is a palindrome, ignoring punctuation and case.

    Args:
        word: The word to check.

    Returns:
        True if the word is a palindrome, False otherwise.
    """
    # Remove punctuation and convert to lowercase
    cleaned_word = re.sub(r'[^\w]', '', word).lower()

    # Check if the cleaned word is equal to its reverse
    return cleaned_word == cleaned_word[::-1]

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

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

Output:



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

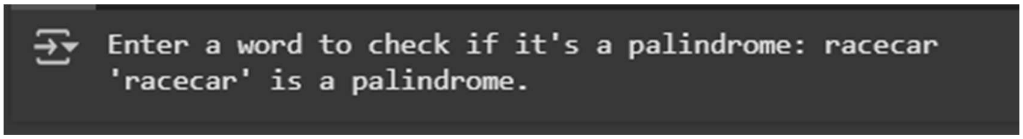
Few-Shot Prompt:



Few-Shot Prompt: Generate a code to Check given word is a palindrome, take dynamic input, Ignore punctuation and case. Examples: Input:'Madam', Output: Palindrome, Input:'Racecar!',Output: Palindrome,
Input:'Hello', Output: Not Palindrome

```python
import re

def is_palindrome(word):
    """
    Checks if a word is a palindrome, ignoring punctuation and case.

    Args:
        word: The word to check.

    Returns:
        True if the word is a palindrome, False otherwise.
    """
    # Remove punctuation and convert to lowercase
    cleaned_word = re.sub(r'[^\w]', '', word).lower()

    # Check if the cleaned word is equal to its reverse
    return cleaned_word == cleaned_word[::-1]

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

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

# Examples:
# Input: 'Madam', Output: Palindrome
# Input: 'Racecar!', Output: Palindrome
# Input: 'Hello', Output: Not Palindrome
```

Output:



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

Comparison Explanation:

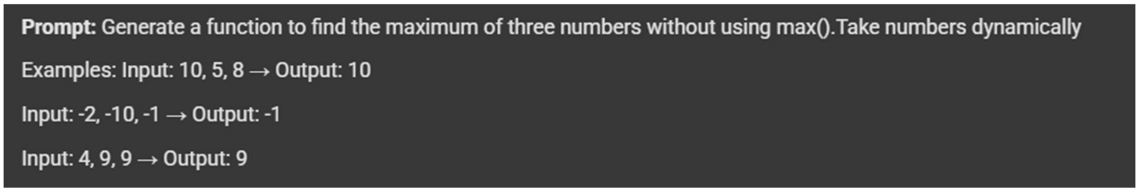
- Both functions are correct, but the few-shot version gives more clarity and consistency in logic.
- With examples, AI better handles edge cases like punctuation (!) and uppercase letters.

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.

Expected Output#5

• A function that handles all cases with correct logic based on example patterns

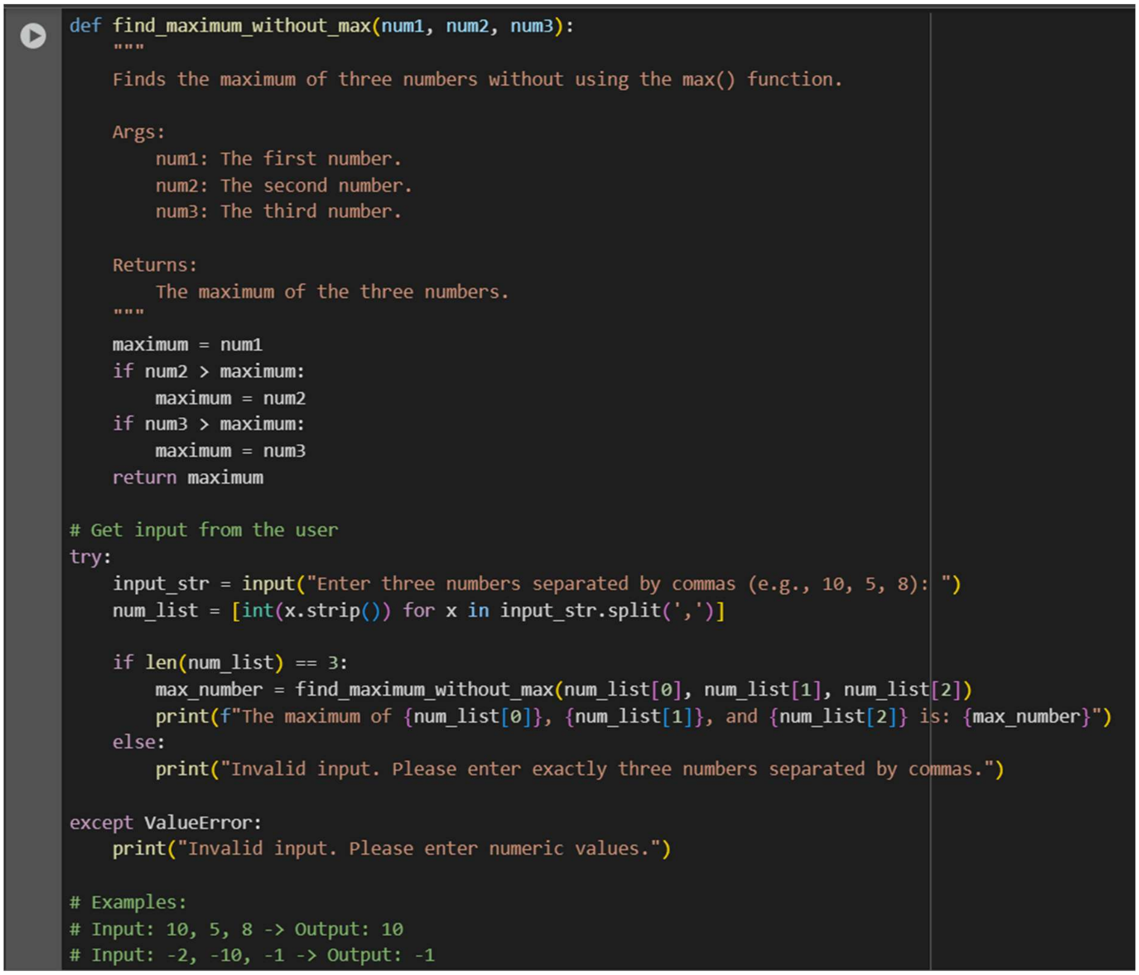


Prompt: Generate a function to find the maximum of three numbers without using max().Take numbers dynamically

Examples: Input: 10, 5, 8 → Output: 10

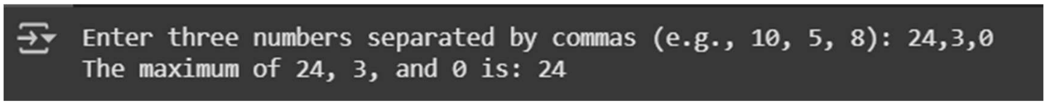
Input: -2, -10, -1 → Output: -1

Input: 4, 9, 9 → Output: 9



```
def find_maximum_without_max(num1, num2, num3):
    """
    Finds the maximum of three numbers without using the max() function.

    Args:
        num1: The first number.
        num2: The second number.
        num3: The third number.

    Returns:
        The maximum of the three numbers.
    """
    maximum = num1
    if num2 > maximum:
        maximum = num2
    if num3 > maximum:
        maximum = num3
    return maximum

# Get input from the user
try:
    input_str = input("Enter three numbers separated by commas (e.g., 10, 5, 8): ")
    num_list = [int(x.strip()) for x in input_str.split(',')]

    if len(num_list) == 3:
        max_number = find_maximum_without_max(num_list[0], num_list[1], num_list[2])
        print(f"The maximum of {num_list[0]}, {num_list[1]}, and {num_list[2]} is: {max_number}")
    else:
        print("Invalid input. Please enter exactly three numbers separated by commas.")

except ValueError:
    print("Invalid input. Please enter numeric values.")

# Examples:
# Input: 10, 5, 8 -> Output: 10
# Input: -2, -10, -1 -> Output: -1
```

Output:



```
Enter three numbers separated by commas (e.g., 10, 5, 8): 24,3,0
The maximum of 24, 3, and 0 is: 24
```

Explanation:

- **AI uses logical comparisons to find the max.**
- **Few-shot prompting helped define the structure without using max().**