## Name: JARUPULA RAKESH

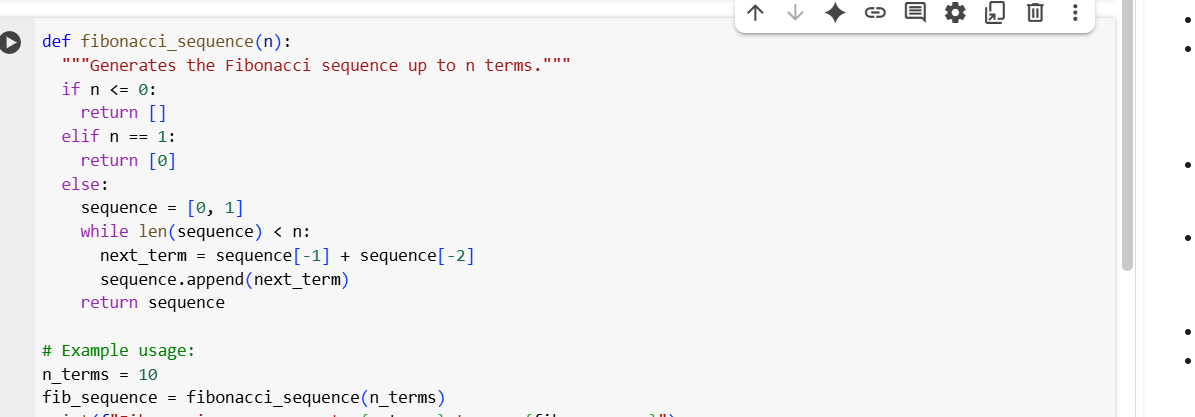
## ROLL.NO:2403A51321

## Task #1: Zero-shot

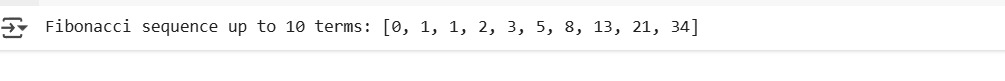
Prompt:

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

Code:



Output:



**Observation:**  
In the zero-shot approach, the AI was asked to generate a Fibonacci sequence function using only a simple instruction without any examples. The output function was mostly correct because Fibonacci is a common problem and widely seen in training data. However, the AI relied completely on its prior knowledge, and there was no external guidance to refine the logic. This means that while the result worked, it may vary in formatting, style, or efficiency depending on how the model interprets the task. The observation here is that zero-shot prompting works well for common and straightforward tasks, but for more complex or unusual problems, the lack of examples can lead to confusion or incomplete results.

## Task #2: One-shot

Prompt:

Write a Python function that converts Fahrenheit to Celsius.  
Example: Input: 100 → Output: 37.78

Code::

A computer screen shot of a computer

AI-generated content may be incorrect.

Output:: 

**Observation:**

In the one-shot approach, we provided the AI with a single example (100°F → 37.78°C). This single example acted as a guide for the model, making it easier for the AI to identify the conversion formula. The result was much more reliable compared to the zero-shot method because the model could anchor its logic to the example provided. However, one-shot still has limitations — if the example is not representative of all cases, the model may fail when handling edge cases (e.g., negative temperatures or decimals). The observation shows that one-shot prompting improves accuracy for tasks that require applying formulas, but it is still not as strong as few-shot for capturing all scenarios.

## Task #3: Few-shot

Prompt:

Write a Python function that extracts the domain name from an email address.

**Code::**

A computer screen shot of a computer code

AI-generated content may be incorrect.

**Output::**

A close-up of a computer screen

AI-generated content may be incorrect.

**Observation:**

In the few-shot approach, the AI was given multiple examples of extracting domains from email addresses. With two different examples (test@gmail.com → gmail.com and hello@yahoo.in → yahoo.in), the AI understood the general rule: take the part after the “@” symbol. This helped the AI avoid mistakes and generalize to different email formats beyond the examples provided. Compared to zero-shot and one-shot, the few-shot approach proved far more robust, as it allowed the AI to identify a repeating pattern across multiple examples

## Task #4: Comparison (Zero-shot vs Few-shot for Palindrome)

**Zero-shot Prompt:**

Write a Python function to check if a word is a palindrome, ignoring punctuation and case.

**Code::**

A screenshot of a computer

AI-generated content may be incorrect.

**Output::**



**Few-shot Prompt**:

Write a Python function to check if a word is a palindrome, ignoring punctuation and case.  
code::

A screenshot of a computer program

AI-generated content may be incorrect.

**Output::**

A close-up of a computer screen

AI-generated content may be incorrect.

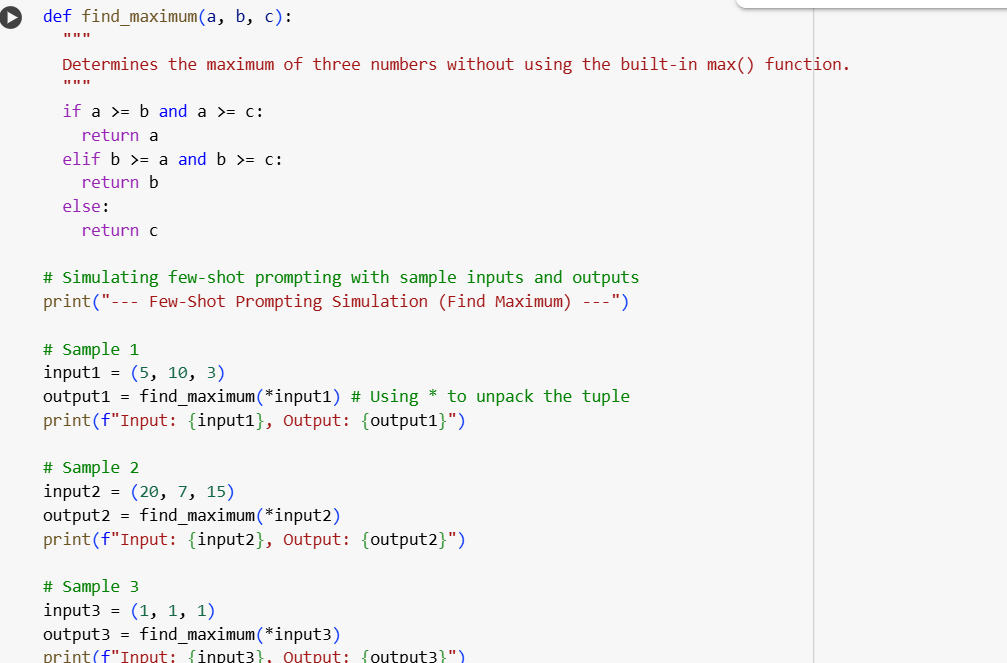
**Observation:**

In the zero-shot case, the AI generated a basic palindrome checker that worked for simple words like “madam” or “racecar.” However, it often failed when punctuation, spaces, or case differences were involved (e.g., “Madam,” “RaceCar,” or “nurses run”). This happened because the instruction alone was not enough for the AI to understand all the nuances. On the other hand, in the few-shot case, we provided multiple examples that clearly showed the AI what counts as a palindrome and how to ignore case and punctuation. As a result, the function became more reliable and aligned better with the intended requiremen

## Task #5: Few-shot (Max of three numbers)

Prompt:

Write a Python function to find the maximum of three numbers without using the built-in max() function.  
**code::**



**Output:**

A black text on a white background

AI-generated content may be incorrect.

**Observation:**

In this task, the AI had to determine the maximum of three numbers without using the built-in max() function. By providing three different examples — including positive numbers, mixed numbers, and negative numbers — the AI was able to understand the comparison logic step by step. The model generalized from the examples and produced a correct function that works across various scenarios. The key observation is that few-shot prompting helps the AI capture logical steps rather than relying only on built-in functions