

LAB ASSIGNMENT

NAME: B.SRINITHI

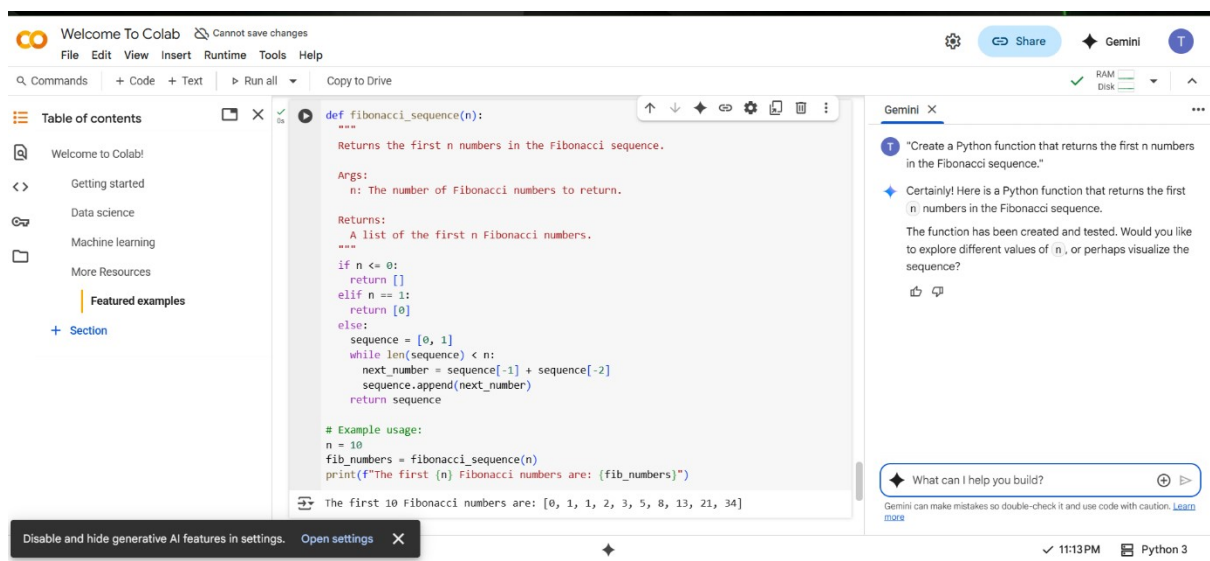
ROLL NO:2403A51413

BRANCH:CSE AIML

TASK 1

PROMPT

Create a python function that return the first n numbers in the fibunacci series



The screenshot displays a Google Colab notebook interface. On the left, a 'Table of contents' sidebar lists sections like 'Welcome to Colab!', 'Getting started', 'Data science', 'Machine learning', and 'More Resources'. The main code editor contains a Python function named `fibonacci_sequence(n)` with a docstring, arguments, and returns. The function logic uses a list to build the sequence iteratively. Below the code, an example usage is shown, and the output displays the first 10 Fibonacci numbers: `[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]`. On the right, a Gemini AI chat window is open, showing a prompt to create a Python function for the Fibonacci sequence, followed by a response that provides the function code and offers further assistance.

```
def fibonacci_sequence(n):  
    """  
    Returns the first n numbers in the Fibonacci sequence.  
  
    Args:  
        n: The number of Fibonacci numbers to return.  
  
    Returns:  
        A list of the first n Fibonacci numbers.  
    """  
    if n <= 0:  
        return []  
    elif n == 1:  
        return [0]  
    else:  
        sequence = [0, 1]  
        while len(sequence) < n:  
            next_number = sequence[-1] + sequence[-2]  
            sequence.append(next_number)  
        return sequence  
  
# Example usage:  
n = 10  
fib_numbers = fibonacci_sequence(n)  
print(f"The first {n} Fibonacci numbers are: {fib_numbers}")
```

The first 10 Fibonacci numbers are: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]

Gemini chat history:
User: "Create a Python function that returns the first n numbers in the Fibonacci sequence."
Gemini: "Certainly! Here is a Python function that returns the first n numbers in the Fibonacci sequence. The function has been created and tested. Would you like to explore different values of n, or perhaps visualize the sequence?"

TASK 2

PROMPT

Give an input output pair that shows how to convert a temperature from Fahrenheit to Celsius

Using examples like

77,100,200

The image displays two screenshots of a Google Colab notebook interface. The top screenshot shows a notebook with a single code cell [2] containing a function `fahrenheit_to_celsius` that converts Fahrenheit to Celsius. The output of the cell is "77 degrees Fahrenheit is equal to 25.0 degrees Celsius." The bottom screenshot shows the same notebook with the code cell [5] updated to include two one-shot examples: `temp_f_oneshot = 200` and `temp_f_oneshot_2 = 100`. The output of the cell now shows the conversion for these values: "200 degrees Fahrenheit is equal to 93.33 degrees Celsius." and "100 degrees Fahrenheit is equal to 37.78 degrees Celsius." The Gemini chat interface on the right side of the bottom screenshot shows a conversation where the user asks for a function to convert Fahrenheit to Celsius, and the Gemini model provides the function and explains its usage.

Table of contents

- Welcome to Colab!
- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- + Section

```
[2] def fahrenheit_to_celsius(fahrenheit):  
    """Converts Fahrenheit to Celsius."""  
    celsius = (fahrenheit - 32) * 5/9  
    return celsius  
  
# Example usage:  
temp_f = 77  
temp_c = fahrenheit_to_celsius(temp_f)  
print(f"{temp_f} degrees Fahrenheit is equal to {temp_c} degrees Celsius.")  
  
77 degrees Fahrenheit is equal to 25.0 degrees Celsius.
```

Gemini X

Using only the above example, create a function that converts a temperature value from Fahrenheit to Celsius.

Okay, I can create a function based on the examples provided in the notebook that converts Fahrenheit to Celsius.

Great! The function has been created and tested with example.

What would you like to do next? I could:

- Explain how the function works.
- Show you how to use the function with a different temperature.
- Introduce you to other features of Colab, like working with data or machine learning.

What can I help you build?

Gemini can make mistakes so double-check it and use code with caution.

Table of contents

- Welcome to Colab!
- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- + Section

```
[5] def fahrenheit_to_celsius(fahrenheit):  
    """Converts Fahrenheit to Celsius."""  
    celsius = (fahrenheit - 32) * 5/9  
    return celsius  
  
# Example usage:  
temp_f = 77  
temp_c = fahrenheit_to_celsius(temp_f)  
print(f"{temp_f} degrees Fahrenheit is equal to {temp_c} degrees Celsius.")  
  
# One-shot example:  
temp_f_oneshot = 200  
temp_c_oneshot = fahrenheit_to_celsius(temp_f_oneshot)  
print(f"{temp_f_oneshot} degrees Fahrenheit is equal to {temp_c_oneshot:.2f} degrees Celsius.")  
  
# Another one-shot example:  
temp_f_oneshot_2 = 100  
temp_c_oneshot_2 = fahrenheit_to_celsius(temp_f_oneshot_2)  
print(f"{temp_f_oneshot_2} degrees Fahrenheit is equal to {temp_c_oneshot_2:.2f} degrees Celsius.")  
  
77 degrees Fahrenheit is equal to 25.0 degrees Celsius.  
200 degrees Fahrenheit is equal to 93.33 degrees Celsius.  
100 degrees Fahrenheit is equal to 37.78 degrees Celsius.
```

Gemini X

ne-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.

I can add that example to the existing Fahrenheit to Celsius conversion function as another test case.

I've added the second one-shot example to the Fahrenheit to Celsius conversion function.

Would you like to add more examples, or is there anything else I can help you with in this notebook? I could explain other sections, like data science or machine learning.

TASK 3

PROMPT

Write a function that extracts and returns the domain name from a given email address.

Provide **2–3 examples** to demonstrate the function.

Welcome To Colab

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all Copy to Drive

Table of contents

Welcome To Colab

- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- + Section

```
[3] def extract_domain(email_address):
    """Extracts the domain name from an email address."""
    try:
        domain = email_address.split('@')[1]
        return domain
    except IndexError:
        return "Invalid email address format"

# Example usage:
email1 = "test@example.com"
email2 = "user.name@sub.domain.co.uk"
email3 = "invalid-email"

print(f"The domain of {email1} is: {extract_domain(email1)}")
print(f"The domain of {email2} is: {extract_domain(email2)}")
print(f"The domain of {email3} is: {extract_domain(email3)}")
```

The domain of test@example.com is: example.com
The domain of user.name@sub.domain.co.uk is: sub.domain.co.uk
The domain of invalid-email is: Invalid email address format

Gemini

Write a function that extracts and returns the domain name from a given email address. Provide 2-3 examples to demonstrate the function.

Sure, I can write a function to extract the domain name from an email address and provide examples.

I've created the function 'extract_domain' and demonstrated it with a few examples.

What would you like to do next? I could:

- Explain the code I just wrote.
- Show you how to apply this function to a list of email addresses.
- Continue exploring other features of Colab.

What can I help you build?

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

Variables Terminal 11:32 PM Python 3

TASK 4

PROMPT

Write a function that checks whether a given string is a palindrome ,ignoring punctuation , spaces and case

Welcome To Colab

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all Copy to Drive

Table of contents

Welcome To Colab

- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- Task

- Generate zero-shot prompt
- Generate code (zero-shot)
- Generate zero-shot prompt (retry 1)
- Generate zero-shot prompt (retry 2)
- Generate code (zero-shot)

```
import re

# Plausible zero-shot function based on the prompt description
def is_palindrome_zero_shot(word):
    """Checks if a word is a palindrome, ignoring case and punctuation."""
    cleaned_word = re.sub(r'[^a-zA-Z]', '', word).lower()
    return cleaned_word == cleaned_word[::-1]

# Plausible few-shot function based on the prompt description and examples
def is_palindrome_few_shot(word):
    """Checks if a word is a palindrome, ignoring case and punctuation."""
    cleaned_word = re.sub(r'[^a-zA-Z]', '', word).lower()
    return cleaned_word == cleaned_word[::-1]

test_cases = [
    "Racecar",
    "Hello, World!",
    "A man, a plan, a canal: Panama",
    "No 'x' in Nixon",
    "madam",
    "python",
    "Able, was I ere I saw Elba!",
    "121",
    "1 2 1",
    "Palindrome?",
    "",
    "A"
]
```

Gemini

function that finds the largest of three numbers without using the built-in max() function.

- Generate code (few-shot):** Use the few-shot prompt with the ai.generate_text function to generate the code.
- Test the generated code:** Test the generated function with various inputs to ensure it works correctly.
- Finish task:** Summarize the results and present the generated code and test results.

Generating...

- Preparing
- Generate few-shot prompt
- Generate code (few-shot)
- Test the generated code
- Finish task

What can I help you build?

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

Variables Terminal 12:03 AM Python 3

[illegible]

TASK 5

PROMPT

Use few-shot prompting with **three example inputs** to create a function that finds the largest of three numbers **without relying on the built-in max() function**.

The screenshot shows the Google Colab interface. On the left is a 'Table of contents' sidebar with a 'Task' section containing several prompts. The main code editor area displays a Python function named `few_shot_largest_number_code` that takes three numbers as input and returns the largest one without using the built-in `max()` function. The function is defined as follows:

```
[19] few_shot_largest_number_code = few_shot_prompt
print(few_shot_largest_number_code)
```

Below the code, there are three example inputs and their corresponding outputs:

```
Input: 1, 5, 2
Output: 5

Input: 10, 3, 7
Output: 10

Input: 4, 9, 9
Output: 9
```

On the right side of the interface is a Gemini chat window. It contains a prompt asking for a function to find the largest of three numbers without using `max()`. The Gemini response provides a plan to use few-shot prompting and lists several tasks to complete the assignment:

- **Generate few-shot prompt:** Create a prompt with instructions and three examples for a Python function that finds the largest of three numbers without using the built-in `max()` function.
- **Generate code (few-shot):** Use the few-shot prompt with the `ai.generate_text` function to generate the code.
- **Test the generated code:** Test the generated function with various inputs to ensure it works correctly.
- **Finish task:** Summarize the results and present the

The bottom of the interface shows a status bar with 'Variables', 'Terminal', and 'Python 3' tabs, along with a timestamp of 11:53 PM.