

AI-Assisted Coding

Name: Vangapandla Koushik

Htno: 2403A52004

Task-1:

Prompt: Generate Fibonacci sequence up to n terms with user input.

Code&Output:

The screenshot displays a Google Colab notebook interface. The top bar shows the notebook is titled 'Untitled6.ipynb' and is part of a 'LAB TEST-1: AI Assisted Coding' assignment. The notebook contains a Python script that defines a function `fibonacci_sequence(n)` to generate the Fibonacci sequence up to `n` terms. The script includes input validation and a try-except block to handle user input errors. The output of the script shows the user entering 5 terms, resulting in the sequence [0, 1, 1, 2, 3].

```
def fibonacci_sequence(n):
    """Generates the Fibonacci sequence up to n terms."""
    sequence = []
    a, b = 0, 1
    for _ in range(n):
        sequence.append(a)
        a, b = b, a + b
    return sequence

try:
    n_terms = int(input("Enter the number of terms for the Fibonacci sequence: "))
    if n_terms <= 0:
        print("Please enter a positive integer.")
    else:
        fib_sequence = fibonacci_sequence(n_terms)
        print("Fibonacci sequence:")
        print(fib_sequence)
except ValueError:
    print("Invalid input. Please enter an integer.")
```

The output of the script is:

```
Enter the number of terms for the Fibonacci sequence: 5
Fibonacci sequence:
[0, 1, 1, 2, 3]
```

The Gemini AI assistant's response is shown on the right side of the interface. It acknowledges the prompt and offers to generate the code. It also provides suggestions for further exploration, such as calculating the sum of the sequence, finding a specific term, visualizing the sequence, or exploring other mathematical concepts.

Gemini X

Generate Fibonacci sequence up to n terms with user input.

Sure, I can help you with that! I'll generate the code to generate the Fibonacci sequence.

I've generated the Fibonacci sequence based on your input. What would you like to do next? Perhaps we could:

- Calculate the sum of the sequence?
- Find a specific term in the sequence?
- Visualize the sequence?
- Explore another mathematical concept?

Let me know what you have in mind!

What can I help you build?

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

Observation: The `fibonacci_sequence(n)` function builds the sequence by starting with 0 and 1, then repeatedly adding the last two numbers to get the next.

Task-2:

Prompt: Convert Fahrenheit to Celsius using a function with user input, print result rounded to 2 decimals.

Code&Output:

The screenshot shows a Google Colab notebook interface. The browser tabs at the top include 'LAB TEST-1: AI Assisted Coding', 'assignment 4.2', 'Untitled6.ipynb - Colab', and 'Fibonacci sequence explanation'. The address bar shows the Colab URL. The notebook's menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. The code editor contains the following Python code:

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

The output cell shows the execution result:

```
Enter temperature in Fahrenheit: 250  
250.0°F is equal to 121.11°C
```

On the right side, the Gemini chat interface is open. It contains the following text:

Convert Fahrenheit to Celsius using a function with user input, print result rounded to 2 decimals.

I can certainly help with that! Here's the code to convert Fahrenheit to Celsius with user input and round the result.

I have converted the temperature from Fahrenheit to Celsius for you. What would you like to explore next? We could:

- Convert Celsius back to Fahrenheit?
- Work with other temperature units like Kelvin?
- Calculate the average of a list of temperatures?
- Perform a different unit conversion?

Let me know what you'd like to do!

At the bottom of the Gemini chat, there is a prompt: "What can I help you build?" and a disclaimer: "Gemini can make mistakes so double-check it and use code with caution. [Learn more](#)".

Observation: The gemini generates a function which converts a Fahrenheit to the Celsius , as I give the example , it may checks the example and give the exact suitable function to it.

Task-3:

Prompt: Write a Python program with a function to extract the domain from email addresses. Take multiple emails as dynamic user input and print only the domain part for each.

Code&Output:

The screenshot displays a Google Colab environment with a Python notebook titled 'Untitled6.ipynb'. The code defines a function `extract_domain(email)` that splits an email at the '@' symbol. It includes a try-except block to handle `IndexError` if the '@' is missing, returning 'Invalid email format'. The main program prompts the user to enter email addresses one by one until 'done' is entered. It then prints the extracted domains for each email.

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

print("Enter email addresses one by one. Type 'done' when you are finished.")
emails = []
while True:
    email_input = input("Enter an email address: ")
    if email_input.lower() == 'done':
        break
    emails.append(email_input)

print("\nExtracted domains:")
for email in emails:
    domain = extract_domain(email)
    print(f"Email: {email}, Domain: {domain}")
```

The terminal output shows the user entering three email addresses: `koushik5511@gmail.com`, `yamshi1432@outlook.com`, and `pardhu2001@email.com`. The program successfully extracts the domains: `gmail.com`, `outlook.com`, and `email.com`.

On the right, the Gemini AI chat interface is visible. It contains a prompt: "Write a Python program with a function to extract the domain from email addresses. Take multiple emails as dynamic user input and print only the domain part for each." The AI response confirms it will create such a program and lists suggestions for further enhancements like validation, counting domains, and file input. Below the chat is a search bar with the text "What can I help you build?".

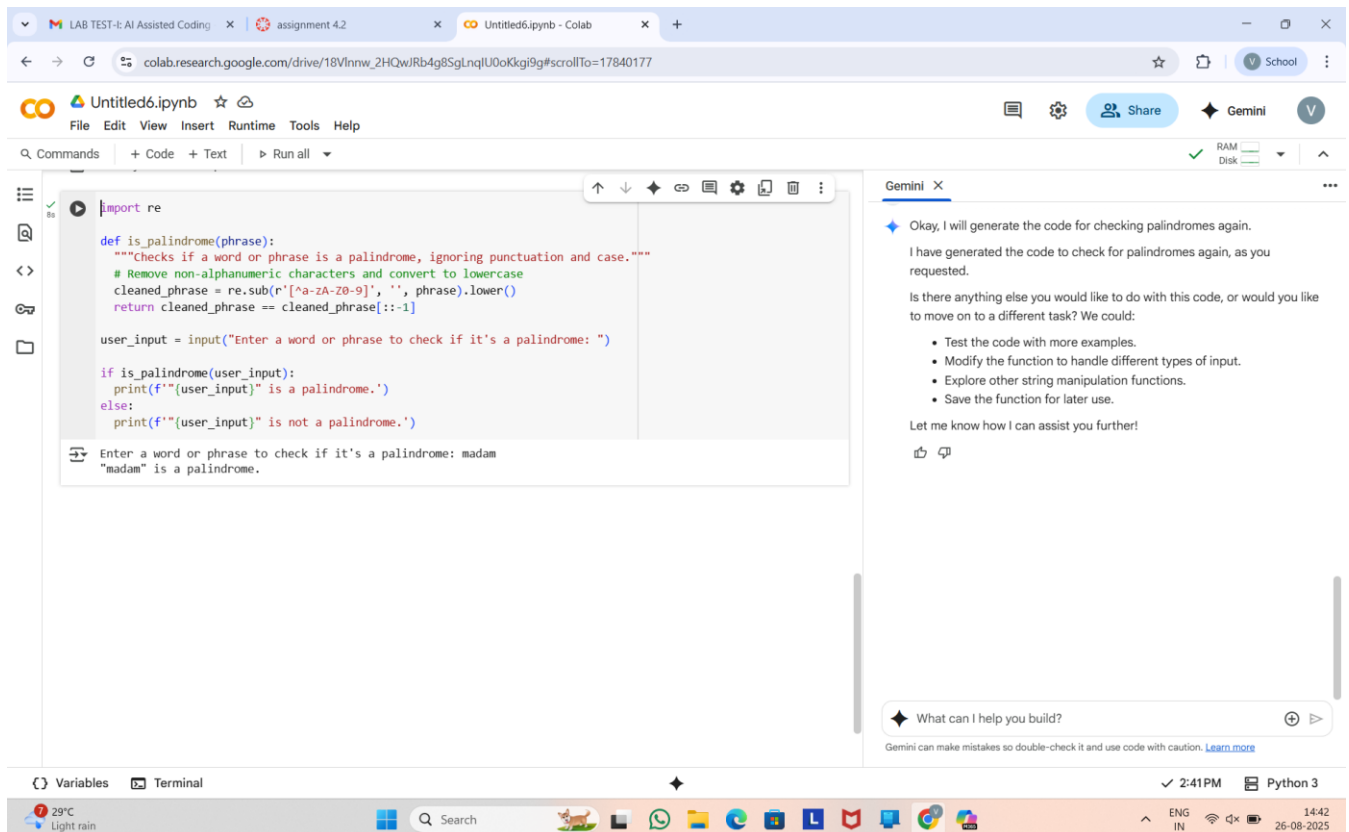
Observation: The code correctly extracts the domain portion of a valid email and gives an appropriate error message if the @ symbol is missing. However, it does not handle other invalid cases (like multiple @ symbols or missing domain part).

Task-4:

Prompt: Write a Python program with a function to check if a word or phrase is a palindrome, ignoring punctuation and case. Take dynamic user input and print whether it is a palindrome or not. for example

input:madam and output is "madam" is a palindrome
and input:vamshi and output is "vamshi" is not a
palindrome

Code&Output:



```
import re

def is_palindrome(phrase):
    """Checks if a word or phrase is a palindrome, ignoring punctuation and case."""
    # Remove non-alphanumeric characters and convert to lowercase
    cleaned_phrase = re.sub(r'[^a-zA-Z0-9]', '', phrase).lower()
    return cleaned_phrase == cleaned_phrase[::-1]

user_input = input("Enter a word or phrase to check if it's a palindrome: ")

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

Enter a word or phrase to check if it's a palindrome: madam
"madam" is a palindrome.

Gemini ✕

Okay, I will generate the code for checking palindromes again.

I have generated the code to check for palindromes again, as you requested.

Is there anything else you would like to do with this code, or would you like to move on to a different task? We could:

- Test the code with more examples.
- Modify the function to handle different types of input.
- Explore other string manipulation functions.
- Save the function for later use.

Let me know how I can assist you further!

What can I help you build?

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

Observation: The code given and also even the output will be the same for the both codes , but the few -shot prompt given examples gives a clarity that they guide both the code logic and the validation of correctness.

Task-5:

Prompt: Generate a python function that the which gives the output the largest number from the three given input numbers , for example input:2,3,4 output : 4 is largest number , if we didn't give three correct numbers it raises an error, the input has to be taken from user and generate the largest number.

Code&Output:

The screenshot displays a Google Colab notebook environment. The top section shows the notebook's title bar and navigation tabs. The main area contains a Python code cell with the following code:

```
def find_largest_number(num1, num2, num3):  
    """Finds the largest of three numbers."""  
    if not all(isinstance(i, (int, float)) for i in [num1, num2, num3]):  
        raise ValueError("All inputs must be numbers.")  
  
    return max(num1, num2, num3)  
  
try:  
    input_str = input("Enter three numbers separated by commas: ")  
    num_list = [float(num.strip()) for num in input_str.split(',')]  
  
    if len(num_list) != 3:  
        raise ValueError("Please enter exactly three numbers.")  
  
    largest = find_largest_number(num_list[0], num_list[1], num_list[2])  
    print(f"{largest} is the largest number.")  
  
except ValueError as e:  
    print(f"Error: {e}")  
except Exception as e:  
    print(f"An unexpected error occurred: {e}")
```

Below the code cell, the output is shown: "Enter three numbers separated by commas: 50,1,5" followed by "50.0 is the largest number." To the right of the code cell is a Gemini chat interface. It contains a prompt that matches the one in the text above, followed by a response from Gemini. The response states that it will create a Python function to find the largest of three numbers and handle invalid input. It then lists several suggestions for further exploration, such as finding the smallest number, finding the largest or smallest number from a list, sorting numbers, and exploring other numerical operations. The chat interface also includes a search bar and a status bar at the bottom.

Observation: The code correctly takes three numbers as input from the user, validates that exactly three values are provided, and finds the largest among them. It also handles errors gracefully. However, it uses

Python's built-in `max()` function, like the prompt asked to raises an error—so the logic works, then it fully satisfy the given requirement.