DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

Course Title – AI Assisted Coding

Course Code : 24CS002PC215

Assignment Type : Lab

Academic Year : 2025–2026

Name : Soumith Sripathi

Enroll Number : 2403A51340

Batch Number : 24BTCAICSB14

Week – 4 Date : 18 AUGUST

Task Description –1

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

PROGRAM AND OUTPUT :

```
[1] def generate_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

    # Example usage:
    n_terms = 10
    fibonacci_sequence = generate_fibonacci_sequence(n_terms)
    print(fibonacci_sequence)

    [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

Explanation :

~ This defines a function named generate_fibonacci_sequence.

~ It takes one argument: n, which represents how many terms of the Fibonacci sequence to generate.

~ sequence = [] Initializes an empty list to store the Fibonacci numbers.

a, b = 0, 1
These are the starting values of the sequence:
a represents the current Fibonacci number.
b represents the next Fibonacci number.
~ for _ in range(n):
Loops n times, once for each term to be generated.
sequence.append(a)
Appends the current value of a to the list
a, b = b, a + b
~ Updates the values:
a becomes the previous b
b becomes the sum of previous a and b
~ This is how the next Fibonacci number is computed in each step.
~ After the loop finishes, the full sequence is returned.

Task Description –2

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

PROGRAM AND OUTPUT :

Explanation :

~ This line defines a function named fahrenheit_to_celsius.

~ It takes one argument: fahrenheit (the temperature in Fahrenheit).

~ This line contains the formula to convert Fahrenheit to Celsius:

Celsius=(Fahrenheit−32)×59

Celsius=(Fahrenheit−32)×95

~ The result is stored in the variable celsius.

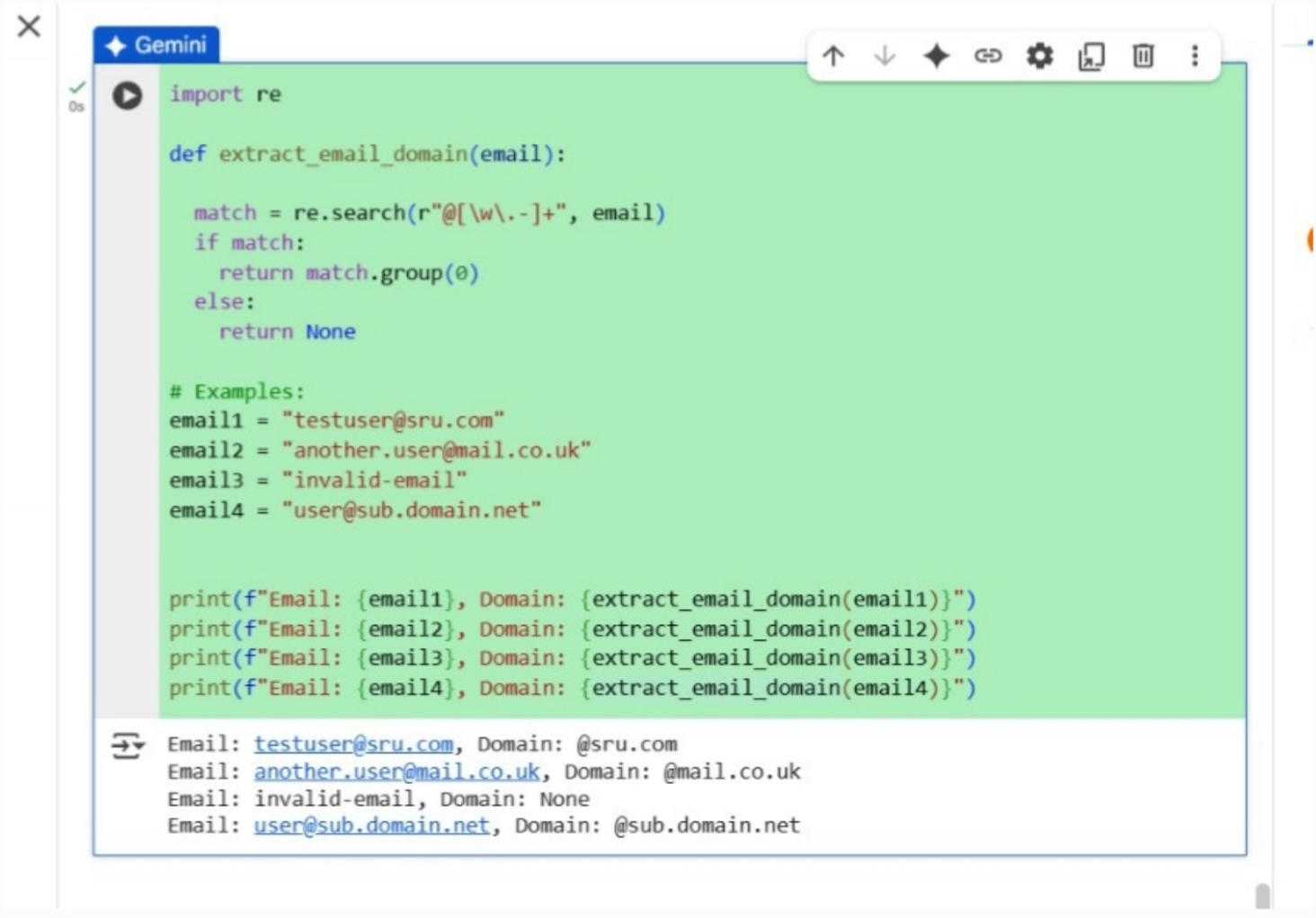
~ This line returns the converted Celsius value from

the function.
~ Here, we set the Fahrenheit input value to 100 as an example.
~ This line calls the function fahrenheit_to_celsius with fahrenheit_input (100).
~It stores the returned Celsius value in celsius_output.
~ This line prints both the input and output values. {celsius_output:.2f} formats the Celsius output to 2 decimal places.
Input: 100, Output: 37.78

Task Description –3

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

PROGRAM AND OUTPUT



Explanation :

~ The re module in Python allows you to work with regular expressions.

~ It's used for pattern matching within strings (e.g., to search for an email structure).

~ This defines a function named extract_email_domain that takes an email string as input.

~ re.search() looks for the first match of the pattern in the input string.

The pattern @[\w\.–]+ breaks down as:

@: Matches the "@" character literally.

[\w\.−]+: Matches one or more characters that are either\w: Alphanumeric (a−z, A−Z, 0−9, or underscore),.: Dot (.)−: Hyphen (−)
~ If a match is found, match.group(0) returns the matched part of the string (e.g., @domain.com).
If no match (invalid email), it returns None.
~ These are sample email addresses used to test the function.
~ Each line prints the email and the extracted domain using an f−string.
The function extract_email_domain() is called for each email.
~ First, second, and fourth emails are valid domains extracted correctly.
~ Third email is invalid (no @) → returns None.

Task Description −4

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

PROGRAM AND OUTPUT :

```
[12] # Palindrome Checker

     import re

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

       Args:
         word: The input word string.

       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]

     # Example usage:
     word1 = "Racecar"
     word2 = "Hello, world!"
     word3 = "A man, a plan, a canal: Panama"

     print(f"'{word1}' is a palindrome: {is_palindrome(word1)}")
     print(f"'{word2}' is a palindrome: {is_palindrome(word2)}")
     print(f"'{word3}' is a palindrome: {is_palindrome(word3)}")

     'Racecar' is a palindrome: True
     'Hello, world!' is a palindrome: False
     'A man, a plan, a canal: Panama' is a palindrome: True
```

Explanation :

~ Imports Python's regular expression module. Needed to remove punctuation and non alphanumeric characters.

~ Defines a function is_palindrome that takes a string word as input.

~ re.sub(r'[^\w]', '', word) removes all characters except letters, numbers, and underscores.

^\w means anything not a word character.

.lower() converts everything to lowercase.

~ Compares the cleaned word to its reverse:

[::–1] reverses the string. Returns True if they're the

same (palindrome), else False.
~ These strings are used to demonstrate the function.
~ Uses formatted string literals (f"") to print the result. {word1!r} prints the string with quotes, for clarity.

Task Description –5

Question : 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.

PROGRAM AND OUTPUT :

Gemini

```
[13] # Function to find the maximum of three numbers without using max()

     # Few-shot examples:
     # Input: 5, 10, 3
     # Output: 10

     # Input: 1, 1, 1
     # Output: 1

     # Input: -5, -2, -8
     # Output: -2

     def find_maximum_of_three(num1, num2, num3):
       """
       Determines the maximum of three numbers without using the built-in max() function.

       Args:
         num1, num2, num3: The three numbers to compare.

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

     # Testing with the few-shot examples:
     print(f"Input: 5, 10, 3, Output: {find_maximum_of_three(5, 10, 3)}")
     print(f"Input: 1, 1, 1, Output: {find_maximum_of_three(1, 1, 1)}")
     print(f"Input: -5, -2, -8, Output: {find_maximum_of_three(-5, -2, -8)}")

     Input: 5, 10, 3, Output: 10
     Input: 1, 1, 1, Output: 1
     Input: -5, -2, -8, Output: -2
```

Explanation ：

~This defines a function named find_maximum_of_three that takes three numbers as input: num1, num2, and num3.

~ Assume num1 is the largest to begin with.

~ If num2 is greater than the current maximum, update maximum to num2.

~ If num3 is greater than the current maximum, update maximum to num3.

~ After the comparisons, the maximum variable

holds the highest value among the three, which is returned.
~ This part tests the function with different input values and prints the result.
~ The function correctly identifies the maximum in each case.
~ Input: 5, 10, 3, Output: 10
Input: 1, 1, 1, Output: 1
Input: −5, −2, −8, Output: −2