AI ASSISTED CODING

Prompt Engineering-Improving Prompts

H.NO:2403a510G7 **Name: N.Prudhvi**

Batch:06

TASK-1:

CODE:

```
# Create a dummy CSV file for demonstration
data = {'col1': [1, 2, 3, 4, 5], 'col2': ['a', 'b', 'c', 'd', 'e'], 'col3': [10.5, 20.1, 5.9, 15.0, 25.3]}
dummy_df = pd.DataFrame(data)
dummy_file_path = 'dummy_data.csv'
dummy_df.to_csv(dummy_file_path, index=False)

# Use the function to analyze a column
analysis_results = analyze_csv_column(dummy_file_path, 'col1')

# Print the results
if analysis_results:
    print("Analysis Results:")
    print(f"Mean: {analysis_results['mean']}")
    print(f"Min: {analysis_results['min']}")
    print(f"Max: {analysis_results['max']}")

# Analyze a non-numeric column to show the warning
analysis_results_non_numeric = analyze_csv_column(dummy_file_path, 'col2')

# Analyze a column with mixed data types
analysis_results_mixed = analyze_csv_column(dummy_file_path, 'col3')

if analysis_results_mixed:
    print("\nAnalysis Results for mixed data column:")
    print(f"Mean: {analysis_results_mixed['mean']}")
    print(f"Min: {analysis_results_mixed['min']}")
    print(f"Max: {analysis_results_mixed['max']}")
```

OUTPUT:

```
Analysis Results:
Mean: 3.0
Min: 1
Max: 5
Warning: Column 'col2' contains no valid numeric data for calculations.

Analysis Results for mixed data column:
Mean: 15.36
Min: 5.9
Max: 25.3
```

TASK-2:

Gemini

```
def is_palindrome_gemini(text):
  """Checks if a string is a palindrome (reads the same forwards and backward),
  ignoring case and non-alphanumeric characters.

  Args:
    text: The input string to check.

  Returns:
    True if the string is a palindrome, False otherwise.
  """
  cleaned_text = ''.join(char for char in text.lower() if char.isalnum())
  return cleaned_text == cleaned_text[::-1]
```

Copilot

```
def is_palindrome_copilot(text):
  """Checks if a string is a palindrome (reads the same forwards and backward),
  ignoring case and non-alphanumeric characters.

  Args:
    text: The input string to check.

  Returns:
    True if the string is a palindrome, False otherwise.
  """
  cleaned_text = ''.join(char for char in text.lower() if char.isalnum())
  return cleaned_text == cleaned_text[::-1]
```

```
Comparison and Observations:
- Both functions are identical in their logic and implementation.
- Both use the same approach to clean the input string: converting to lowercase and removing non-alphanumeric characters.
- Both use slicing [::-1] to reverse the cleaned string and compare it to the original cleaned string.
- Variable names (`text`, `cleaned_text`) are the same in both functions.
- The docstrings are also identical, providing the same information about the function's purpose, arguments, and return value.
- The overall structure of both functions is exactly the same.
```

TASK-3:

CODE:

```
import math

def circle_area(radius):
  """Calculates the area of a circle given its radius."""
  if radius < 0:
    return "Error: Radius cannot be negative"
  else:
    return math.pi * (radius ** 2)

# Example usage
radius1 = 5
area1 = circle_area(radius1)
print(f"The area of a circle with radius {radius1} is: {area1}")

radius2 = -2
area2 = circle_area(radius2)
print(f"The area of a circle with radius {radius2} is: {area2}")
```

OUTPUT:

```
The area of a circle with radius 5 is: 78.53981633974483
The area of a circle with radius -2 is: Error: Radius cannot be negative
```

TASK-4:

PROMPT:

generate a Python function (e.g., sum of squares).

Okay, here is a Python function that calculates the sum of squares of numbers in a list:

I have generated and executed the code for the `sum_of_squares` function.

CODE:

```
def sum_of_squares(numbers):
  """Calculates the sum of squares of numbers in a list."""
  return sum(x**2 for x in numbers)

# Example usage
my_list = [1, 2, 3, 4, 5]
result = sum_of_squares(my_list)
print(f"The sum of squares of {my_list} is: {result}")
```

```
The sum of squares of [1, 2, 3, 4, 5] is: 55
```

TASK-5:

CODE & OUTPUT:

```python
def sum_odd_even(numbers):
  """Calculates the sum of odd and even numbers in a list."""
  sum_even = 0
  sum_odd = 0
  for number in numbers:
    if number % 2 == 0:
      sum_even += number
    else:
      sum_odd += number
  return sum_even, sum_odd

# Example usage
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
sum_even, sum_odd = sum_odd_even(my_list)
print(f"The list is: {my_list}")
print(f"Sum of even numbers: {sum_even}")
print(f"Sum of odd numbers: {sum_odd}")
```

```
The list is: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Sum of even numbers: 30
Sum of odd numbers: 25
```