EXP 3: Evaluation of Prompting Tools Across Diverse AI Platforms: ChatGPT, Claude, Bard, Cohere Command, and Meta

Aim:

The goal of this evaluation is to compare how prompting tools perform on five AI platforms: ChatGPT, Claude, Bard, Cohere Command, and Meta. This report analyzes each tool's response quality, accuracy, creativity, and practical application across various types of prompts. By examining how each platform responds to different types of prompts, we aim to identify strengths and limitations, ultimately guiding users in choosing the best tool for their specific needs.

Procedure:

1. Specified task:
   - ➔ Generation of Python code for a given question.

2. Set of Prompts:

   Given an array of coin amounts from students, find the sum of all distinct coin values, ignoring duplicates. The input includes an array of coin values and the number of students, and the output is the sum of unique coin values.

Generate a Python code for the above question with better time and space complexity.

ChatGPT	**Gemini**	**BlackBox.AI**

def
  sum_unique_coins(coin_values,
  num_students):
   # Use a set to store unique coin
  values
   unique_coins =
  set(coin_values)
 	
   # Return the sum of unique
  coin values
   return sum(unique_coins)
 	
# Example Usage: 	
coin_values = [1, 2, 2, 3, 4, 5, 3]
  # List of coin values
num_students = len(coin_values)
# Number of students 	
	def
	
	sum_distinct_coins(coin_values):
	
	   """
	
	   Calculates the sum of distinct
	
	coin values in an array.
	
	   Args: coin_values: An
	array of coin values.
	
	Returns:
	
	     The sum of distinct coin
	
	values.
		def
		sum_of_unique_coins(coin_values):
		
		   # Convert the list of coin values
		to a set to remove duplicates
		
		   unique_coins = set(coin_values)
		
		   # Calculate the sum of the unique
		coin values
		
		   total_sum = sum(unique_coins)
		
		   return total_sum

result = sum_unique_coins(coin_values, num_students) print(f"The sum of distinct coin values is: {result}") Output: Output is not satisfied and the execution time is more.	""" # Use a set to store unique coin values unique_coins = set(coin_values) # Return the sum of unique coin values return sum(unique_coins) # Example usage: coin_values = [1, 2, 3, 2, 1, 5] result = sum_distinct_coins(coin_values) print(result) Output: Generates error.	# Example usage: if __name__ == "__main__": # Sample input coin_values = [1, 2, 2, 3, 4, 4, 5] result = sum_of_unique_coins(coin_values) print("Sum of unique coin values:", result) Output: Output satisfied and the execution time is efficient.

Conclusion:

By analyzing the results from ChatGPT, Gemini, and BlackBox.ai for the given question, it can be concluded that the code generated by BlackBox.ai demonstrates greater clarity, accuracy, and execution efficiency.