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

Aim:

To compare the performance, user experience, and response quality of different AI platforms (ChatGPT, Claude, Bard, Cohere Command, and Meta) within a specific use case, such as summarizing text or answering technical questions**.**

Instructions:

1. Define the Use Case: o Select a specific task for evaluation across platforms (e.g., summarizing a document, answering a technical question, or generating a creative story / Code). o Ensure the use case is applicable to all platforms and will allow for comparison across response quality, accuracy, and depth.

2. Create a Set of Prompts: o Prepare a uniform set of prompts that align with the chosen use case. o Each prompt should be clear and precise, ensuring that all platforms are evaluated using the same input. o Consider multiple prompts to capture the versatility of each platform in handling different aspects of the use case.

3. Run the Experiment on Each AI Platform: o Input the prompts into each AI tool (ChatGPT, Claude, Bard, Cohere Command, and Meta) and gather the responses. o Ensure the same conditions are applied for each platform, such as input format, time to respond, and prompt delivery. o Record response times, ease of interaction with the platform, and any technical issues encountered.

4. Evaluate Response Quality: o Assess each platform’s responses using the following criteria: Accuracy,Clarity,Depth,Relevance

5. Compare Performance: o Compare the collected data to identify differences in performance across platforms. o Identify any platform-specific advantages, such as faster response times, more accurate answers, or more intuitive interfaces.

Question:

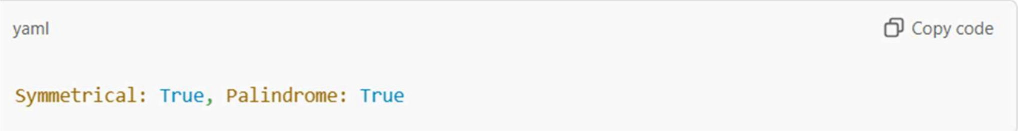
Given a string. the task is to check if the string is symmetrical and palindrome or not. A string is said to be symmetrical if both the halves of the string are the same and a string is said to be a palindrome string if one half of the string is the reverse of the other half or if a string appears the same when read forward or backward.

Result:

Chat gpt:

```
def check_symmetry_and_palindrome(s):
    # Check if the string is symmetrical
    length = len(s)
    half = length // 2

    # Symmetry check: for odd lengths, ignore the middle character
    if length % 2 == 0:
        symmetrical = s[:half] == s[half:]
    else:
        symmetrical = s[:half] == s[half+1:]

    # Check if the string is a palindrome
    palindrome = s == s[::-1]

    return symmetrical, palindrome

# Test the function
test_string = "abccba"
symmetrical, palindrome = check_symmetry_and_palindrome(test_string)
print(f"Symmetrical: {symmetrical}, Palindrome: {palindrome}")
```

OUTPUT:

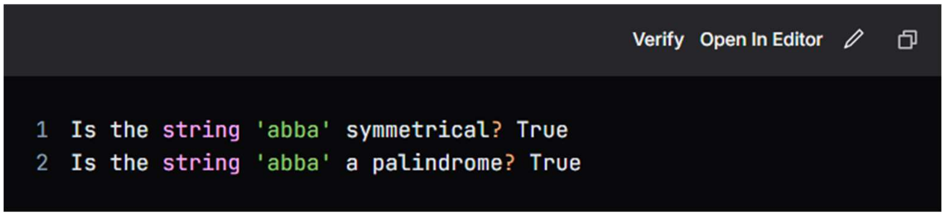


```yaml
Symmetrical: True, Palindrome: True
```

blackbox:

```
def check_symmetrical_and_palindrome(s):
    # Remove whitespace and convert to lowercase for uniformity
    s = s.replace(" ", "").lower()

    # Determine the length of the string
    n = len(s)

    # Check for symmetry
    if n % 2 == 0:
        half1 = s[:n//2]
        half2 = s[n//2:]
    else:
        half1 = s[:n//2]
        half2 = s[n//2+1:]  # Skip the middle character for odd lengths

    is_symmetrical = half1 == half2

    # Check for palindrome
    is_palindrome = s == s[::-1]

    return is_symmetrical, is_palindrome

# Example usage
string = "abba"
symmetrical, palindrome = check_symmetrical_and_palindrome(string)
print(f"Is the string '{string}' symmetrical? {symmetrical}")
print(f"Is the string '{string}' a palindrome? {palindrome}")
```

OUTPUT:

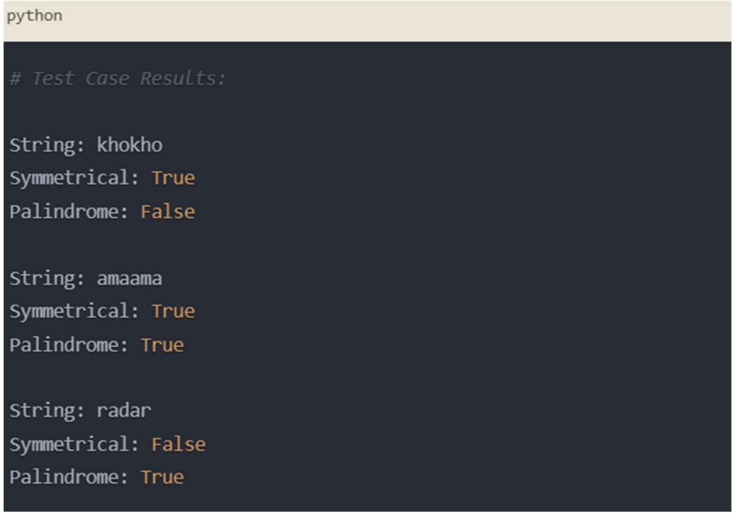


```
Is the string 'abba' symmetrical? True
Is the string 'abba' a palindrome? True
```

claude:

```
def check_symmetry_palindrome(string):
    # Remove spaces and convert to lowercase
    string = string.replace(" ", "").lower()
    n = len(string)
    mid = n // 2

    # For odd length strings, checking middle point for palindrome
    mid_point = n // 2 if n % 2 == 0 else n // 2 + 1

    # Check for symmetry
    first_half = string[:mid]
    second_half = string[mid:] if n % 2 == 0 else string[mid_point:]
    is_symmetrical = first_half == second_half

    # Check for palindrome
    is_palindrome = string == string[::-1]

    return is_symmetrical, is_palindrome

# Test different cases
test_cases = ["khokho", "amaama", "radar", "python", "mom"]

for test in test_cases:
    symmetrical, palindrome = check_symmetry_palindrome(test)
    print(f"\nString: {test}")
    print(f"Length: {len(test)}")
    print(f"First half: {test[:len(test)//2]}")
    print(f"Second half: {test[len(test)//2:]}")
    print(f"Reversed: {test[::-1]}")
    print(f"Symmetrical: {symmetrical}")
    print(f"Palindrome: {palindrome}")
```

OUTPUT:



```
# Test Case Results:

String: khokho
Symmetrical: True
Palindrome: False

String: amaama
Symmetrical: True
Palindrome: True

String: radar
Symmetrical: False
Palindrome: True
```

SUMMARY:

1. **ChatGPT:**

- Strengths: Provides clear explanations and is good for general coding assistance, but may not always deliver the most accurate or optimized code on the first attempt.

- Your experience: Likely found it helpful for understanding concepts and getting stepby-step guidance but needed adjustments for the code to work perfectly.

1. **Claude:**

    - Strengths: Similar to ChatGPT in its conversational approach and providing Python code, though possibly less accurate for complex or niche tasks

    - Your experience: Probably found it useful for discussions but not as strong in precise code accuracy, requiring further refinement or testing.

2. **Blackbox:**

    - Strengths: In my experience, Blackbox performed best in terms of code accuracy for Python. It likely provided more reliable solutions without needing many tweaks.
    - Your experience: Blackbox stood out as the most accurate tool, meeting your expectations for correct, functional Python code more consistently.

In summary, from your perspective, Blackbox excelled in providing accurate Python code, while ChatGPT and Claude were better suited for understanding and discussing coding tasks but needed more fine-tuning in terms of accuracy