# Cricket Matches Assignment Solution

Cricket Matches Assignment Solution

Problem Statement

You will be given an API which contains a list of recent cricket matches with their data.

The task is to print a few key results from the given set of matches.

Solution Approach (Java):

1. Trigger the API to get the JSON response using HttpURLConnection.

2. Parse the JSON response to extract relevant fields.

3. Calculate the highest score in one innings with the team name.

4. Calculate the number of matches with a total score of 300+.

5. Print and return the computed results.

Sample Java Code:

```
import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import org.json.JSONArray;

import org.json.JSONObject;

public class CricketMatchesAssignment {

    private static final String API_URL = "https://api.cuvora.com/car/partner/cricket-data";
```

```
private static final String API_KEY = "test-creds@2320";


public static void main(String[] args) {
    try {
        // Trigger API call
        String response = sendGetRequest(API_URL, API_KEY);
        // Parse JSON response
        JSONArray matches = new JSONArray(response);
        int highestScore = 0;
        String teamWithHighestScore = "";
        int matchesWith300Plus = 0;

        for (int i = 0; i < matches.length(); i++) {
            JSONObject match = matches.getJSONObject(i);
            String t1 = match.getString("t1");
            String t2 = match.getString("t2");
            String t1s = match.optString("t1s", "0");
            String t2s = match.optString("t2s", "0");

            int team1Score = parseScore(t1s);
            int team2Score = parseScore(t2s);

            // Check for highest score
            if (team1Score > highestScore) {
                highestScore = team1Score;
                teamWithHighestScore = t1;
```

```
            }
            if (team2Score > highestScore) {
                highestScore = team2Score;
                teamWithHighestScore = t2;
            }

            // Check for 300+ total match score
            if (team1Score + team2Score > 300) {
                matchesWith300Plus++;
            }
        }

        // Print the results
                System.out.println("Highest Score: " + highestScore + " and Team Name is: " +
teamWithHighestScore);
        System.out.println("Number Of Matches with total 300 Plus Score: " + matchesWith300Plus);

    } catch (Exception e) {
        e.printStackTrace();
    }
}

// Method to send GET request and get API response
private static String sendGetRequest(String apiUrl, String apiKey) throws Exception {
    URL url = new URL(apiUrl);
    HttpURLConnection connection = (HttpURLConnection) url.openConnection();
```

```
        connection.setRequestMethod("GET");
        connection.setRequestProperty("apiKey", apiKey);

                                        BufferedReader in = new BufferedReader(new
InputStreamReader(connection.getInputStream()));
        String inputLine;
        StringBuffer content = new StringBuffer();
        while ((inputLine = in.readLine()) != null) {
            content.append(inputLine);
        }
        in.close();
        return content.toString();
    }

    // Method to parse score from string to integer
    private static int parseScore(String score) {
        if (score == null || score.isEmpty()) {
            return 0;
        }
        String[] parts = score.split("/");
        try {
            return Integer.parseInt(parts[0]);
        } catch (NumberFormatException e) {
            return 0;
        }
    }
```

}

Key Results:

- Highest Score: Computed by comparing team scores.

- Number of Matches with 300+ total score: Computed by summing team scores for each match.