**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **20/may/2020** | **Name:** | **Prashantha naik** |
| **Course:** | **Tcs ion** | **USN:** | **4al17ec074** |
| **Topic:** | **1.Understand Artificial Intelligence (AI) - Part 1**  **2.Understand Artificial Intelligence (AI) - Part 2**  **3. Assessment** | **Semester & Section:** | **6th b** |
| **Github Repository:** | **prashanth\_course** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report – Report can be typed or hand written for up to two pages.** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:** | **22/may/2020** | **Name:** | **Prashantha naik** | |
| **Course:** | **python** | **USN:** | **4al17ec074** | |
| **Topic:** | 1. **Project Exercise with Python and MySQL: Interactive English** 2. **Dictionary** 3. **Data Analysis with Pandas** 4. **Application 2: Create Web maps with Python and Folium** | **Semester& Section:** | **6th b** | |
| **Git hub repository** | **prashanth\_couse** |  |  | |
| **AFTERNOON SESSION DETAILS** | | | |
| **Image of session** | | | |
| **Report – Report can be typed or hand written for up to two pages.**  **Project Exercise with Python and MySQL: Interactive English**  **Dictionary**  That statement retrieved all the rows of the Dictionary table where the value of the column Expression was rain. The string inside cursor. Execute () is SQL code that Python sends to the database. That kind of language is understood by the database.  Here are some more examples of SQL queries that you can try out from within your Python script just like we did previously:   1. Get all rows where the value of the column Expression starts with r: 2. "SELECT \* FROM Dictionary WHERE Expression  LIKE 'r%'" 3. Get all rows where the value of the column Expression starts with rain: 4. "SELECT \* FROM Dictionary WHERE Expression  LIKE 'rain%'" 5. All rows where the length of the value of the column Expression is less than four characters: 6. "SELECT \* FROM Dictionary WHERE length (Expression) < 4" 7. All rows where the length of the value of the column Expression is four characters: 8. "SELECT \* FROM Dictionary WHERE length (Expression) = 4" 9. All rows where the length of the value of the column Express ion is greater than 1 but less than 4 characters: 10. "SELECT \* FROM Dictionary WHERE length (Expression) > 1 AND length (Expression) < 4" 11. All rows of column Definition where the value of the column Expression starts with r: 12. "SELECT Definition FROM Dictionary WHERE Expression LIKE 'r%' 13. import json 14. from difflib import get\_close\_matches 15. data = json.load(open("data.json")) 16. def translate(w): 17. w = w.lower() 18. if w in data: 19. return data[w] 20. elif w.title() in data: 21. return data[w.title()] 22. elif w.upper() in data: #in case user enters words like USA or NATO 23. return data[w.upper()] 24. elif len(get\_close\_matches(w, data.keys())) > 0: 25. yn = input("Did you mean %s instead? Enter Y if yes, or N if no: " % get\_close\_matches(w, data.keys())[0]) 26. if yn == "Y": 27. return data[get\_close\_matches(w, data.keys())[0]] 28. elif yn == "N": 29. return "The word doesn't exist. Please double check it." 30. else: 31. return "We didn't understand your entry." 32. else: 33. return "The word doesn't exist. Please double check it." 34. word = input("Enter word: ") 35. output = translate(word) 36. if type(output) == list: 37. for item in output: 38. print(item) 39. else: 40. print(output)   **Data Analysis with Pandas**  The code for loading the supermarkets.json file in Python with pandas would be this:   1. import pandas 2. df2 = pandas.read\_json("supermarkets.json")   The df2 dataframe should contain this data:    **Application 2: Create Web maps with Python and Folium**     1. import folium 2. import pandas 4. data = pandas.read\_csv("Volcanoes.txt") 5. lat = list(data["LAT"]) 6. lon = list(data["LON"]) 7. elev = list(data["ELEV"]) 9. html = """<h4>Volcano information:</h4> 10. Height: %s m 11. """ 13. map = folium.Map(location=[38.58, -99.09], zoom\_start=5, tiles="Mapbox Bright") 14. fg = folium.FeatureGroup(name = "My Map") 16. for lt, ln, el in zip(lat, lon, elev): 17. iframe = folium.IFrame(html=html % str(el), width=200, height=100) 18. fg.add\_child(folium.Marker(location=[lt, ln], popup=folium.Popup(iframe), icon = folium.Icon(color = "green")))  21. map.add\_child(fg) 22. map.save("Map\_html\_popup\_simple.html") | | | |