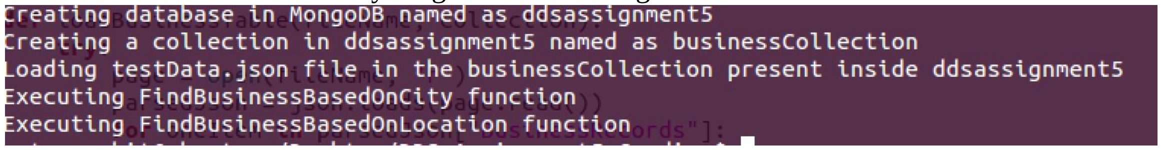
The required task is two write two functions, which will perform some textual and spatial searching on MongoDB. Details are explained below.

Steps for Assignments: - 1.
Install *MongoDB 2.6.11*.
2. Install *pymongo* to act as helper interface with MongoDB.
3. Run the tester file to check everything runs and nothing fails.



```
Creating database in MongoDB named as ddsassignment5
Creating a collection in ddsassignment5 named as businessCollection
Loading testData.json file in the businessCollection present inside ddsassignment5
Executing FindBusinessBasedOnCity function
Executing FindBusinessBasedOnLocation function
```

4. Now, Implement the function provided in Assignment5_Interface.py to perform the operations as listed below:
   a. *FindBusinessBasedOnCity(cityToSearch, saveLocation1, collection)*
      This function searches the 'collection' given to find all the business present in the city provided in 'cityToSearch' and save it to 'saveLocation1'. For each business you found, you should store name Full address, city, state of business in the following format.
      Each line of the saved file will contain,
      Name$FullAddress$City$State. ($ is the separator and must be present)
   b. *FindBusinessBasedOnLocation(categoriesToSearch, myLocation, maxDistance, saveLocation2, collection)*
      This function searches the 'collection' given to find name of all the business present in the 'maxDistance' from the given 'myLocation' (please use the distance algorithm given below) and save them to 'saveLocation2'. Each line of the output file will contain the name of the business only.

NOTE: - Please Make sure all fields are in Uppercase while writing in file, makes it easy to check.
Files given to you: -

- **Assignment5_Interface.py -** You should complete this file. You would need to implement the function present in this file.

- **testData.json –** This is the json file which is used as a document to put inside MongoDb. The structure of one record of this file is < Key value pair>: -

```
{
'type': 'business',
'business_id': (encrypted business id),
'name': (business name),
'neighborhoods': [(hood names)],
'full_address': (localized address),
'city': (city),
'state': (state),
'latitude': latitude,
'longitude': longitude,
'stars': (star rating, rounded to half-stars),
'review_count': review count,
'categories': [(localized category names)]
'open': True / False (corresponds to permanently closed, not business hours), }
```

Example: -

```
{"city": "Ahwatukee",
"review_count": 3,
"name": "McDonald's",
"neighborhoods": [],
"type": "business",
"business_id": "LNdwp-9Isnd6xmBKUz4K_A",
"full_address": "10823 S 51st St\nAhwatukee, AZ 85044",
"state": "AZ",
"longitude": -111.975004,
"stars": 2.0,
"latitude": 33.348560900000003,
"open": true,
 "categories": ["Burgers", "Fast Food", "Restaurants"]}
```

Note: - The order of key value pair does not matter.

- **tester.py –** DO NOT change this file. This is just to help you run the code for your implementation.

Distance Algorithm needs to be used:

Given two pair of latitude and longitude as [lat2, lon2] and [lat1, lon1], you can calculate the distance between them using the formula given below:

```
DistanceFunction(lat2, lon2, lat1, lon1):
    dlon = lon2 - lon1  dlat
    = lat2 - lat1
    a = (sin(dlat/2))^2 + cos(lat1) * cos(lat2) * (sin(dlon/2))^2  c
    = 2 * atan2( sqrt(a), sqrt(1-a) )
    dist = R * c (where R is the radius of the Earth) return
    dist
```

dist is the distance between the given pair of latitude and longitude.