Aim: Implement python program using geopy and folium libraries.

Theory:

geopy is a Python client for several popular geocoding web services.

geopy makes

it easy for Python developers to locate the coordinates of addresses, cities, countries, and landmarks across the globe using third-party geocoders and other data sources.

## Installation

Install using pip with:

```
pip install geopy
```

## Measuring Distance

Geopy can calculate geodesic distance between two points using the geodesic distance or the great-circle distance, with a default of the geodesic distance available as the function *geopy.distance.distance*.

Here's an example usage of the geodesic distance, taking pair of `(lat, lon)` tuples:

```
>>> from geopy.distance import geodesic

>>> newport_ri = (41.49008, -71.312796)

>>> cleveland_oh = (41.499498, -81.695391)

>>> print(geodesic(newport_ri, cleveland_oh).miles)
```

```
538.390445368
```

Using great-circle distance, also taking pair of `(lat, lon)` tuples:

```
>>> from geopy.distance import great_circle

>>> newport_ri = (41.49008, -71.312796)

>>> cleveland_oh = (41.499498, -81.695391)

>>> print(great_circle(newport_ri, cleveland_oh).miles)

536.997990696
```

## Getting Address Information

Now that we have established the connection, let's try to retrieve information using the coordinates.

```
location = geocode.reverse((lat, long))
# returns geopy Location object
```

Location objects have instances of **address, altitude, latitude, longitude, point**. `address` returns a concatenated string of all information that belongs to the address. This may or may not include information like an **amenity, road, neighborhood, city block, suburb, county, city, state, postcode, country, country code**. We can look more directly into the structure of information by using the `raw` instance.

```
location.raw
# returns a dictionary
```

Again, each API will have a different architecture, so it’s a good idea to always check the raw version. From this raw dictionary we can pull the information we would like. In this case, let’s extract postal-code, which is nested under `address`.

```
zipcode = location.raw['address']['postcode']
```

## Getting Distance

Another important variable we can extract from the coordinates is the distance between different locations. This is a simple task using Geopy. Geopy offers both the great-circle distance and the geodesic distance.

```
from geopy.distance import geodesicdistance_in_miles = geodesic(coordinate1, coordinate2).miles
distannce_in_km = geodesic(coordinate1, coordinate2).km
# note: coordinates must be in a tuple of (lat, long)
```

# Folium:

One of the most important tasks for someone working on datasets with countries, cities, etc. is to understand the relationships between their data’s physical location and their geographical context. And one such way to visualize the data is using .

Folium is a powerful data visualization library in Python that was built primarily to help people visualize geospatial data. With Folium, one can create a map of any location in the world. Folium is actually a python wrapper for leaflet.js which is a javascript library for plotting interactive maps.

Installation:

```
pip install folium
```

```
# initialize the map and store it in a m object
```

```
m = folium.Map(location = [40, -95],
            zoom_start = 4)

# show the map
m.save('my_map.html')
```

output:

Golfe du Saint-Laurent / Gulf of Saint Lawrence
Ottawa
New York
Washington
United States
Phoenix
Los Angeles
México
Ciudad de México
La Habana
The Bahamas
Cuba
República
Leaflet | Data by © OpenStreetMap, unde

**Example:**

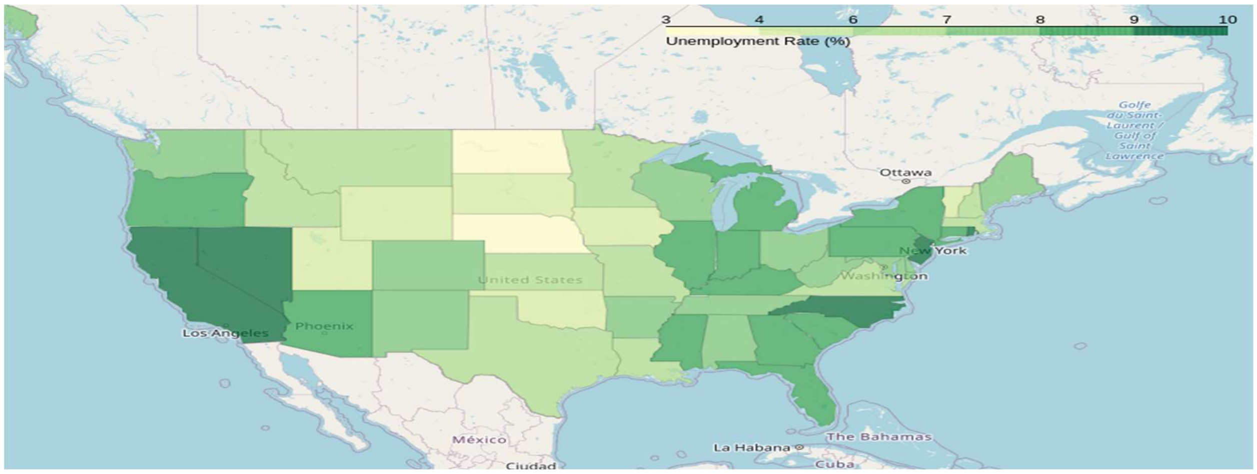
```
folium.Choropleth(


        # geographical locations
        geo_data = state_geo,
        name = "choropleth",
        # the data set we are using
        data = state_data,
        columns = ["State", "Unemployment"],

        # YlGn refers to yellow and green
        fill_color = "YlGn",
        fill_opacity = 0.7,
        line_opacity = .1,
        key_on = "feature.id",
        legend_name = "Unemployment Rate (%)",
).add_to(m)

m.save('final_map.html')
```

Output:



## Conclusion:

Learned about geopy and folium ,implemented various functions ,finding locations using latitude and longitude etc.