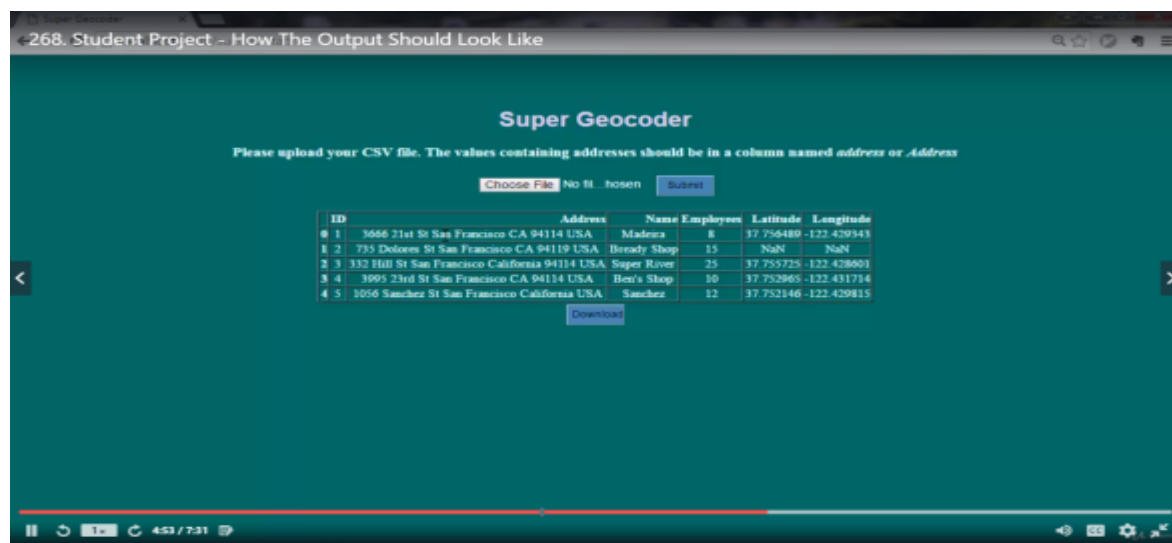


DAILY ASSESSMENT FORMAT

Date:	06-06-2020	Name:	Yamunashree N
Course:	Python	USN:	4AL17EC097
Topic:	geocoding	Semester & Section:	6 th sem 'B' sec
Github Repository:	yamunashree-course		

Image of session



Variables (Practice)

Create a variable named `character` and assign the value `"Jon Snow"` to that variable.

Don't forget that everything in Python is case sensitive, which means `character` is considered different from `Character`.

`Jon` is considered different from `Jon`, and so on.

The solution is in this link.

Important points on how to solve these types of exercises:

1. Write your code solution in the area down below and then click on Check Solution
2. If your solution is wrong try to understand the error message you get, modify your solution, and click on Check Solution again.
3. Repeat the aforementioned process until you get the correct solution or skip if you want to surrender.



Edit with WPS Office

Geocoding request and response (latitude/longitude lookup) The following example requests the latitude and longitude of "1600 Amphitheatre Parkway, Mountain View, CA", and specifies that the output must be in JSON format. you can test this by entering the URL into your web browser (be sure to replace YOUR_API_KEY with your actual API key). The response includes the latitude and longitude of the address.

View the developer's guide for more information about building geocoding request URLs and available parameters and understanding the response.

Below is a sample geocoding response, in JSON:

```
{
  "results": [
    {
      "address_components": [
        {
          "long_name": "1600",
          "short_name": "1600",
          "types": [ "street_number" ]
        },
        {
          "long_name": "Amphitheatre Parkway",
          "short_name": "Amphitheatre Pkwy",
          "types": [ "route" ]
        },
        {
          "long_name": "Mountain View",
          "short_name": "Mountain View",
          "types": [ "locality", "political" ]
        }
      ]
    }
  ]
}
```



```
    },
    {
      "long_name" : "Santa Clara County",
      "short_name" : "Santa Clara County",
      "types" : [ "administrative_area_level_2", "political" ]
    },
    {
      "long_name" : "California",
      "short_name" : "CA",
      "types" : [ "administrative_area_level_1", "political" ]
    },
    {
      "long_name" : "United States",
      "short_name" : "US",
      "types" : [ "country", "political" ]
    },
    {
      "long_name" : "94043",
      "short_name" : "94043",
      "types" : [ "postal_code" ]
    }
  ],
  "formatted_address" : "1600 Amphitheatre Pkwy, Mountain View, CA 94043, USA",
  "geometry" : {
    "location" : {
      "lat" : 37.4267861,
      "lng" : -122.0806032
    },
    "location_type" : "ROOFTOP",
    "viewport" : {
```



```

    "northeast" : {
      "lat" : 37.4281350802915,
      "lng" : -122.0792542197085
    },
    "southwest" : {
      "lat" : 37.4254371197085,
      "lng" : -122.0819521802915
    }
  }
},
"place_id" : "ChIJtYuu0V25j4ARwu5e4wwRYgE",
"plus_code" : {
  "compound_code" : "CWC8+R3 Mountain View, California, United States",
  "global_code" : "849VCWC8+R3"
},
"types" : [ "street_address" ]
}
],
"status" : "OK"
}

```

Reverse geocoding request and response (address lookup) The following example requests the address corresponding to a given latitude/longitude in Brooklyn, NY, USA. It specifies that the output must be in JSON format. You can test this by entering the URL into your web browser (be sure to replace 'YOUR_API_KEY' with your actual API key). The response includes a human-readable address for the latitude and longitude location.

View the developer's guide for more information about building reverse geocoding request URLs and available parameters and understanding the response.



Below is a sample reverse geocoding response, in JSON:

```
{
  "plus_code" : {
    "compound_code" : "P27Q+MC New York, NY, USA",
    "global_code" : "87G8P27Q+MC"
  },
  "results" : [
    {
      "address_components" : [
        {
          "long_name" : "279",
          "short_name" : "279",
          "types" : [ "street_number" ]
        },
        {
          "long_name" : "Bedford Avenue",
          "short_name" : "Bedford Ave",
          "types" : [ "route" ]
        },
        {
          "long_name" : "Williamsburg",
          "short_name" : "Williamsburg",
          "types" : [ "neighborhood", "political" ]
        }
      ]
    }
  ]
}
```



```
    },
    {
      "long_name": "Brooklyn",
      "short_name": "Brooklyn",
      "types": [ "political", "sublocality", "sublocality_level_1" ]
    },
    {
      "long_name": "Kings County",
      "short_name": "Kings County",
      "types": [ "administrative_area_level_2", "political" ]
    },
    {
      "long_name": "New York",
      "short_name": "NY",
      "types": [ "administrative_area_level_1", "political" ]
    },
    {
      "long_name": "United States",
      "short_name": "US",
      "types": [ "country", "political" ]
    },
    {
      "long_name": "11211",
      "short_name": "11211",
      "types": [ "postal_code" ]
    }
  ],
  "formatted_address": "279 Bedford Ave, Brooklyn, NY 11211, USA",
  "geometry": {
```



```
"location" : {  
  "lat" : 40.7142484,  
  "lng" : -73.9614103  
},  
"location_type" : "ROOFTOP",  
"viewport" : {  
  "northeast" : {  
    "lat" : 40.71559738029149,  
    "lng" : -73.9600613197085  
  },  
  "southwest" : {  
    "lat" : 40.71289941970849,  
    "lng" : -73.96275928029151  
  }  
}  
},  
"place_id" : "ChIJT2x8Q2BZwokRpBu2jUzX3dE",  
"plus_code" : {  
  "compound_code" : "P27Q+MC Brooklyn, New York, United States",  
  "global_code" : "87G8P27Q+MC"  
},  
"types" : [  
  "bakery",  
  "cafe",  
  "establishment",  
  "food",  
  "point_of_interest",  
  "store"  
]
```



```
},
```

... Additional results truncated in this example[] ...

```
],
```

```
"status" : "OK"
```

Start coding with our client libraries Client libraries make developing with the Google Maps web service APIs easier by providing simple, native implementations of common tasks, such as authentication, request throttling and automatic retry. The Geocoding API is available in the Java Client, Python Client, Go Client and Node.js Client for Google Maps Services.

Authentication, quotas, pricing, and policies Authentication To use the Geocoding API, you must first enable the API and obtain the proper authentication credentials. For more information, see [Get Started with Google Maps Platform](#).



Certificate of Completion

*This is to certify that **Yamunashree N** successfully
completed 25.5 total hours of **The Python Mega
Course: Build 10 Real World Applications** online
course on June 6, 2020*

Ardit Sulce
Ardit Sulce, Instructor

&



Certificate no: UC-59ad7fea-d8d1-4467-b49e-9b65d10d0aff
Certificate url: ude.my/UC-59ad7fea-d8d1-4467-b49e-9b65d10d0aff

#BeAble





Edit with WPS Office