Hackwagon Academy - DS101

AirBnB Project

Learning Outcomes:

- Learn how to translate business requirements into workable applications
- Declare variables, and manipulate the variables to perform arithmetic operations
- Create a list, append new elements to a list, remove elements from list, and access elements within a list
- Create a dictionary, access data, and update information within the dictionary
- Be able to aptly make use of if and nested if constructs
- Variable conversion
- Produce visualisations
- Able to come up with insights based on the data

In [ ]:

```
#Before you start, please perform the following 2 steps:
#1. Rename the file to <First_Name>_<Last_Name>_DS101_Lab_1 e.g. john_doe_DS101_Lab_1

#2. Fill in your details here:
#Name                    : Dylan Tan

#Start of Course Class(Edit accordingly): 10 Apr 2018 3.45pm____
# FOR TA/INSTRUCTOR
# Total Marks: 100 / 100
# Part 1: 5 / 5
# Part 2: 25 / 25
# Part 3: 10 / 10
# Part 4: 60 / 60
```

This project is split into 4 different parts: 1. Data Cleaning (5 marks) 2. Explorator Data Analysis (25 marks) 3. Interpretation (10 marks) 4. AirBnB Visualisation and Price Recommender App (60 marks) **All questions must follow expected output to be awarded full marks except the following:** 1. Interpretation

References

Important Collections Functions

Creation

Collection Type	Function	Examples
list	None	`new_list = []` `new_list = [1,2,3,4]`
dict	None	`new_dict = {}` `new_dict = {'a': 1, 'b':2}`

Add / Appending Data

Collection Type	Functions	Examples	Resulting Output
list	`.append()`	`new_list = [1,2,3]` `new_list.append(4)`	`[1,2,3,4]`
list	`.extend()`	`new_list = [1,2]` `new_list.extend([3,4])`	`[1,2,3,4]`

dict	None	`new_dict = {}`
		`new_dict['a'] = 1`
		`new_dict['b'] = 2`
			`{'a': 1, 'b':2}`

Updating / Changing Data

Collection Type	Functions	Examples	Resulting Output
list	None	`new_list = [1,2,3]` `new_list[0] = 5`	`[5,2,3]`
dict	None	`new_dict = {'a': 1, 'b':2}` `new_dict['a'] = 10`	`{'a': 10, 'b':2}`

Accessing / Taking Out Data

Collection Type	Functions	x to be	Examples
list	None	3	`new_list = [1,2,3]` `x = new_list[2]`
list of list	None	3	`new_list = [[1,2],[3,4]]` `x = new_list[1][0]`
list of dict	None	2	`new_list = [{'a':1},{'b':2}]` `x = new_list[1]['b']`

Collection Type	Functions	x to be	Examples
dict	None	2	`new_dict = {'a': 1, 'b':2}` `x = new_dict['b']`

CITU Framework & Applied Iterations

1. What variables do you need to answer this question?
2. **Create** the results container
3. **Iterate** the input data/list
4. **Take out the variables you needed in step 1**
5. **Test** conditions of each value
6. **Update** the results container when condition is fulfilled

Sorting Values

```
x = [10,20,50,2,4]
x.sort()
print(x) # [2,4,10,20,50]
x.sort(reverse=True)
print(x) # [50,20,10,4,2]
```

</hr>

Welcome to your final project of Hackwagon Academy DS101! You've come a long way since the start of this course and if you've been on track with our exercises, you should find this doable.

Airbnb is an online marketplace and hospitality service, enabling people to lease or rent short-term lodging including vacation rentals, apartment rentals, homestays, hostel beds, or hotel rooms. The company does not own any lodging; it is merely a broker and receives percentage service fees (commissions) from both guests and hosts in conjunction with every booking. In this project, we aim to use algorithms and libraries to mine the reviews people have submitted on Singapore AirBnB rentals in order to provide descriptive analytics.

Load File

Load the `airbnb_data.csv` as **a list of dictionaries** into a new variable called `airbnb_data` . Once you load the data, you should see something like this:

```
[
        {
         'listing_id': '1133718',
```

In [2]:

```
# Read file into a list called airbnb_data

import csv
import pprint                    # PPrint represents Pretty Printer to make your output
neater

airbnb_data = []                 # Dictionary - Should keep the headers since headers a
re the "KEY"!!

with open("airbnb_data.csv") as file:
    output = csv.DictReader(file)

    for row in output:
         airbnb_data.append(dict(row))

pprint.pprint(airbnb_data[0:2])
```


```
[{'accommodates': '12',
  'bathrooms': '',
  'bedrooms': '1.0',
  'borough': '',
  'city': 'Singapore',
  'country': '',
  'host_id': '6219420',
  'last_modified': '2017-05-17 09:10:25.431659',
  'latitude': '1.293354',
  'listing_id': '1133718',
  'location': '0101000020E6100000E84EB0FF3AF159409C69C2F693B1F43F',
  'longitude': '103.769226',
  'minstay': '',
  'neighborhood': 'MK03',
  'overall_satisfaction': '4.5',
  'price': '74.0',
  'reviews': '9',
  'room_type': 'Shared room',
  'survey_id': '1280'},
 {'accommodates': '12',
  'bathrooms': '',
  'bedrooms': '1.0',
  'borough': '',
  'city': 'Singapore',
  'country': '',
  'host_id': '15295886',
  'last_modified': '2017-05-17 09:10:24.216548',
  'latitude': '1.310862',
  'listing_id': '3179080',
  'location': '0101000020E6100000E738B709F7F659403F1BB96E4AF9F43F',
  'longitude': '103.858828',
  'minstay': '',
  'neighborhood': 'TS17',
  'overall_satisfaction': '5.0',
  'price': '77.0',
  'reviews': '15',
  'room_type': 'Shared room',
  'survey_id': '1280'}]
```

Data Cleaning (5 marks)

Once this is done correctly, you do not need to change the type for the remaining parts of your project.

Very Big Hint:

```
for row in data:
    row['score'] = int(row['score'])
```

Preview your data and clean them to appropriate type. Namely these columns:

1. overall_satisfaction
2. price
3. longitude
4. latitude
5. reviews

Expected Output:


```
{
        'listing_id': '1133718',
        'survey_id': '1280',
        'host_id': '6219420',
        'room_type': 'Shared room',
        'country': '',
        'city': 'Singapore',
        'borough': '',
        'neighborhood': 'MK03',
        'reviews': 9.0,
        'overall_satisfaction': 4.5,
        'accommodates': '12',
        'bedrooms': '1.0',
        'bathrooms': '',
        'price': 74.0,
        'minstay': '',
        'last_modified': '2017-05-17 09:10:25.431659',
        'latitude': 1.293354,
        'longitude': 103.769226,
        'location': '0101000020E6100000E84EB0FF3AF159409C69C2F693B1F43F'
    }
```

In [9]:

```
#Write code below

for x in airbnb_data:
    x['overall_satisfaction'] = float(x['overall_satisfaction'])
    x['price'] = float(x['price'])
    x['longitude'] = float(x['longitude'])
    x['latitude'] = float(x['latitude'])
    x['reviews'] = float(x['reviews'])

pprint.pprint(airbnb_data[1])

# Correct
```

```
{'accommodates': '12',
 'bathrooms': '',
 'bedrooms': '1.0',
 'borough': '',
 'city': 'Singapore',
 'country': '',
 'host_id': '15295886',
 'last_modified': '2017-05-17 09:10:24.216548',
 'latitude': 1.310862,
 'listing_id': '3179080',
 'location': '0101000020E6100000E738B709F7F659403F1BB96E4AF9F43F',
 'longitude': 103.858828,
 'minstay': '',
 'neighborhood': 'TS17',
 'overall_satisfaction': 5.0,
 'price': 77.0,
 'reviews': 15.0,
 'room_type': 'Shared room',
 'survey_id': '1280'}
```

Exploratory Data Analysis (35 marks)

The data team at AirBnB wishes to find out the answers to a few simple questions on the existing listings in Singapore. Your goal is to manipulate the data you have stored in the list of dictionaries and **understand some of the basic statistics of your dataset.** The following are some of the common *first* questions asked.

Q1. List out each neighborhoods and their number of listings (5 marks)

Hint

1. *Counting with dictionaries, where key is neigbhorhood id, value is counts. </i>*

Expected Output:

```
print(results['TS17']) # 342 counts.
```

In [8]:

```
#Write code below

results = {}

for i in airbnb_data:
    if i['neighborhood'] not in results:
        results[i['neighborhood']] = 1
    else:
        results[i['neighborhood']] += 1

print(results['TS17'])

# Correct
```

342

Q2. List out each neighborhood and their average overall_satisfaction (5 marks)

Note: You should filter out listings whose reviews are 0.

Hint

1. *Create 2 dictionaries*
   - *Dictionary 1: Key is neighborhood, value is an accumulation of scores*
   - *Dictionary 2: Key is neighborhood, value is an accumulation of counts*
2. *Create 3rd dictionary*
3. *Loop through 1 dictionary (using 1 for loop only because both share same key!), calculate average and store in 3rd dictionary </i>*

Expected Output:

```
print(results['TS17']) # it should give you an average score of 2.85944700460829
5.
```

In [10]:

```
#Write code below

scores = {}
counts = {}
results = {}

for i in airbnb_data:
    if i['reviews'] > 0:
        if i['neighborhood'] not in counts:
            counts[i['neighborhood']] = 1
            scores[i['neighborhood']] = i['overall_satisfaction']
        else:
            counts[i['neighborhood']] += 1
            scores[i['neighborhood']] += i['overall_satisfaction']

for x,y in scores.items():
    if y == 0:
        avg = 0
    else:
        avg = y / counts[x]
        results[x] = avg

print(results['TS17'])

# Correct
```

```
2.859447004608295
```

In [ ]:

```

```

Q3. List out each neighborhood and their average price (5 marks)

Hint

1. *Similar to previous question </i>*

Expected Output:

```
print(results['TS17']) # it should give you an average price of 95.567251461988
3.
```

In [11]:

```
#Write code below

price = {}
counts = {}
results = {}

for i in airbnb_data:
    if i['neighborhood'] not in counts:
        counts[i['neighborhood']] = 1
        price[i['neighborhood']] = i['price']
    else:
        counts[i['neighborhood']] += 1
        price[i['neighborhood']] += i['price']

for x,y in price.items():
    if y == 0:
        avg = 0
    else:
        avg = y / counts[x]
        results[x] = avg

print(results['TS17'])

# Correct
```

```
95.5672514619883
```

In [ ]:

```

```

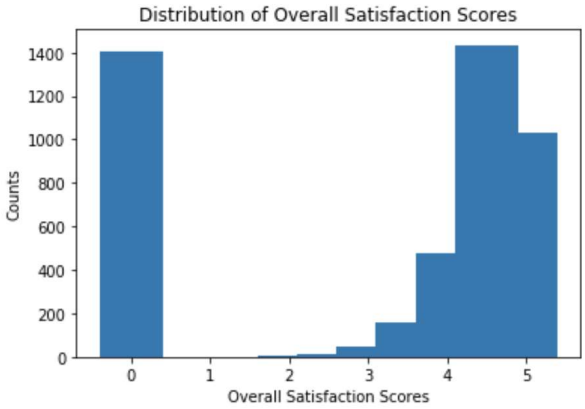
Q4. Plot a distribution of counts of the overall_satisfaction (5 marks)

Note: You should filter out listings whose reviews are 0.

Hint

1. *Counting with dictionaries*
2. *Get a list of tuples with* `.items()`
3. *Create 2 lists:*
   - *1 for all the scores labels*
   - *1 for all the counts*
4. *Plot with the 2 lists </i>*

Expected Output:

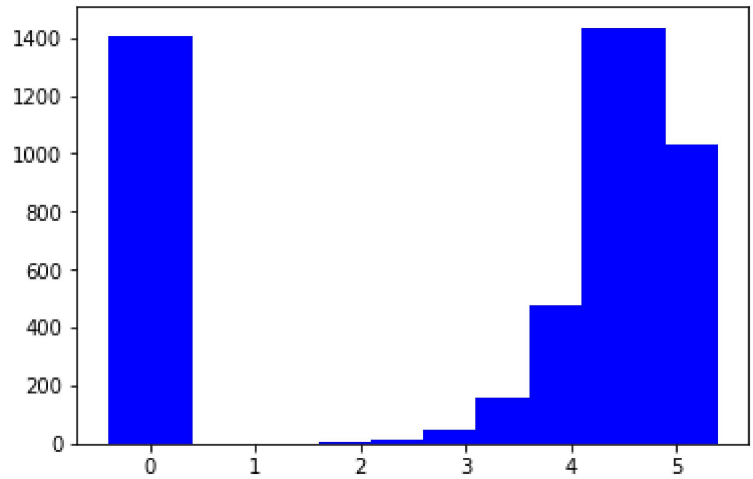


In [15]:

```
# Remember to import the relevant library/libraries!
# Write code below

import matplotlib.pyplot as plt        #bar graph automatically arranges all the keys a
nd corresponding values in the dictionary
import operator                        #accordingly without having the user sorting out
the data physically
neighbourhood_scores1={}
for i in airbnb_data:
    if i["reviews"]>0:
        if i['overall_satisfaction'] not in neighbourhood_scores1:
            neighbourhood_scores1[i["overall_satisfaction"]]= 1
        else:
            neighbourhood_scores1[i["overall_satisfaction"]]+= 1
sorted_x = sorted(neighbourhood_scores1.items(), key=operator.itemgetter(0))
neighbourhood_listcount=[]
neighbourhood_listscores=[]

for i in sorted_x:
    neighbourhood_listcount.append(i[1])
    neighbourhood_listscores.append(i[0])
plt.bar(neighbourhood_listscores, neighbourhood_listcount, color="b")



# Correct
```

Out[15]:

```
<BarContainer object of 10 artists>
```



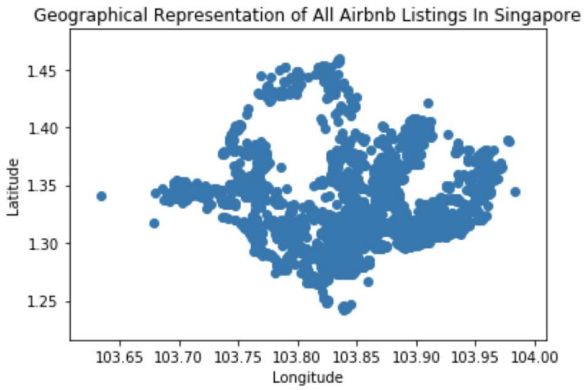
In [ ]:

Q5. Plot a geographical representation of all of the listings in Singapore (5 marks)

Hint

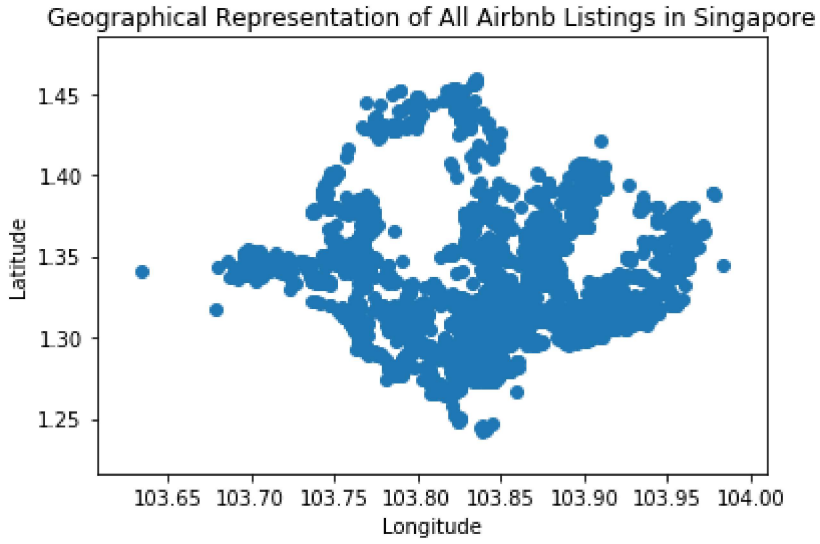
1. *Create a list for latitude*
2. *Create a list for longitude*
3. *Append each listing's latitude and logitude to the lists*
4. *Plot a scatter plot using both lists </i>*

Expected Output:



In [26]:

```
#Write code below

latitude = []
longitude = []

for i in airbnb_data:
    x = i['latitude']
    y = i['longitude']

    latitude.append(x)
    longitude.append(y)

plt.scatter(longitude,latitude)
plt.title('Geographical Representation of All Airbnb Listings in Singapore')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()

# Correct
```



Interpretation

Based on the earlier questions, answer the following questions to understand the Airbnb dataset better.

You're free to make some assumptions

Q1. Why `overall_satisfaction` is in intervals of 0.5? (5 marks)

In [ ]:

```
# Answer: This is to ensure the data collected are easy to manipulate in order to perfo
rm data analytics to plot charts and graphs
        # in forecasting trends and patterns in the long run.

# Correct
```

Q2. Why was there a need to filter reviews greater than 0 in question 2 and 4? (5 marks)

In [ ]:

```
# Answer: The reason is because reviews that are equivalent to 0 will correspond to 0 o
verall satisfaction being given.
        # This will indicate that the specific data has poor satisfaction. Moreover, it
is considered as a count and
        # this will affect the average score for each neighbourhood. This is an inaccur
ate misrepresentation of score
        # when presented to Airbnb and its consumers.

# Correct
```

AirBnB Visualisation and Price Recommender App (60 marks)

Attempts to create the functions are awarded 2 marks each Scenario: *Based on the earlier EDA, we have seen that it is not modular and does not allow the AirBnB team to look into each neighborhood. Nevertheless, the AirBnB data team have tasked you to build a simple application to improve the earlier EDA while serving its 2 users: Guests and Hosts.* ***Your objective:*** *Develop an* ***app*** *which will serve the 2 main users: 1. Guests - Visualisation tool to recommend them the best listings based on price and overall satisfaction score in a neighborhood 2. Hosts - Recommend a price to set for their listing in a given neighborhood based on better performing listings*

THIS NEXT CELL IS IMPORTANT FOR YOUR APPLICATION. *Run it to install this package called* ***mplleaflet****. How do you know if you installed the library correctly? Try running the cell after this one (not the line that says "!pip install mplleaflet", its the other one), if you don't get an error, you are good to go! If you face any issues, please contact any of your TAs or Instructors.*

In [8]:

```
!pip install mplleaflet
```

```
Requirement already satisfied: mplleaflet in c:\users\gelat\anaconda3\lib
\site-packages (0.0.5)
Requirement already satisfied: jinja2 in c:\users\gelat\anaconda3\lib\site
-packages (from mplleaflet) (2.10)
Requirement already satisfied: six in c:\users\gelat\anaconda3\lib\site-pa
ckages (from mplleaflet) (1.12.0)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\gelat\anaconda
3\lib\site-packages (from jinja2->mplleaflet) (1.1.0)
```

How do you know if you installed the library correctly? Try running the next cell, if you don't get an error, you are good to go!

In [10]:

```
import mplleaflet
```

Building the App

To begin building the App, there are 2 things to do:

1. *Building the functions*
2. *Testing the functions*

After we are done building the functions in part 1, we will test them in part 2

> *Every single function you create must have the **airbnb_data** variable as the **first** parameter so that you can use it inside the function.*

```
def example_function_1(data, x, y, ..): # << do not name as airbnb_data again!
    for i in data:
        print(i)

# when using it.. notice that airbnb_data is placed first, followed by the other
parameters
example_function_1(airbnb_data, some_x, some_y, ...)
```

There are a total of 5 functions:

1. *get_all_latitudes*
2. *get_all_longitudes*
3. *listings_recommender*
4. *price_recommender*
5. *visualise_listings*

In [11]:

```
# Running this cell shows no output. This is normal because the function has been creat
ed but not called/used.
def example_function(interesting_data):
    for i in interesting_data:
        print(i)
```

In [21]:

```
# Running this cell shows an output because the function is called/used.
example_function([1,2,3,4,5])
```

```
1
2
3
4
5
```

`get_all_latitudes()` - Functions to get all latitudes given a list of listing_ids (5 marks)

Hint

1. *Create a results list*
2. *Extract the latitude and listing id of each row*
3. *Check listing id exists within the all listings*
4. *If true, append the latitude into the results list*
5. *Return results list </i>*

***Input**: `airbnb_data as data` , a `list` of listing_ids*

***Return**: A `list` of latitudes*

In [12]:

```
#Write code below

def get_all_latitude(data,lists):
    results = []
    for i in data:
        for j in lists:
            if j == i['listing_id']:
                results.append(i['latitude'])
    return results

# Correct
```

***Tester Cell** - To test the above function to see if it's working.*

Expected Output:

```
[1.305702, 1.296138, 1.304393]
```

In [13]:

```
get_all_latitude(airbnb_data, ['10350448','13507262','13642646'])
```

Out[13]:

```
[1.305702, 1.296138, 1.304393]
```

`get_all_longitudes()` - Functions to get all longitudes given a list of listing_ids (5 marks)

Hint

1. *Same as previous question, just that it's about longitudes now </i>*

***Input**: `airbnb_data as data` , a `list` of listing_ids*

***Return**: A `list` of longitudes*

In [33]:

```
#Write code below

def get_all_longitude(data,lists):
    results = []
    for i in data:
        for j in lists:
            if j == i['listing_id']:
                results.append(i['longitude'])
    return results


# Correct
```

***Tester Cell** - To test the above function to see if it's working.*

Expected Output:

```
[103.79878, 103.767841, 103.784174]
```

In [34]:

```
get_all_longitude(airbnb_data, ['10350448','13507262','13642646'])
```

Out[34]:

```
[103.79878, 103.767841, 103.784174]
```

`listings_recommender()` - Function to recommend all listings based on a given price, satisfaction score and neighborhood (15 marks)

Note:

1. *Lesser than or equal to that price*
2. *Equal or more than that overall satisfaction score*
3. *In that neighborhood*

Hint

1. *Create a results list*
2. *Extract the relevant of each row*
3. *Check it satisfies all conditions passed into the function*
4. *If true, append the listing id into the results list*
5. *Return results list </i>*

***Input**: `airbnb_data as data` , price, overall_satisfaction, neighborhood_id*

***Return**: A `list` of listing_ids*

In [31]:

```
#Write code below
def listings_recommender(data,a,b,c):
    results = []
    for i in data:
        if i['price'] <= a and i['overall_satisfaction'] >= b and i['neighborhood'] ==
c: #ALL conditions must be satisfied
            results.append(i['listing_id'])
    return results


# Correct
```

***Tester Cell** - To test the above function to see if it's working.*

Expected Output:

```
['10350448',
 '13507262',
 '13642646',
 '15099645',
 '6451493',
 '4696031',
 '2898794',
 '13181050',
 '9022211',
 '5200263',
 '6529707',
 '14433262']
```

In [32]:

```
listings_recommender(airbnb_data, 60, 5, 'MK03')
```

Out[32]:

```
['10350448',
 '13507262',
 '13642646',
 '15099645',
 '6451493',
 '4696031',
 '2898794',
 '13181050',
 '9022211',
 '5200263',
 '6529707',
 '14433262']
```

`price_recommender()` - Function to recommend a price in a neighborhood based on average price and overall satisfaction (15 marks)

*For this function, we want to build a **simple** price recommendation function that will give a potential host a suggested price.*

To build this, these are the requirements:

1. *Take all listings in that neighborhood and check for listings with a least 1 review and an overall satisfaction score of 4 or more.*
2. *From that filtered listings, calculate the average price and return that as the suggested price rounded to 2 decimal places.*

***Input**: `airbnb_data as data` , a neighborhood_id*

***Return**: A `float` of recommended price*

In [29]:

```
#Write code below

def price_recommender(data,x):
    price = 0
    count = 0
    for i in data:
        if i['neighborhood'] == x and i['reviews'] >= 1 and i['overall_satisfaction'] >
= 4:
            count += 1
            price += i['price']
    average_price = price / count
    return round(average_price,2)


# Correct
```

Tester Cell *- To test the above function to see if it's working.*

Expected Output:

```
66.28
```

In [16]:

```
price_recommender(airbnb_data, 'TS17')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-16-9303dd41681d> in <module>
----> 1 price_recommender(airbnb_data, 'TS17')

<ipython-input-15-586d43739740> in price_recommender(data, x)
      5     count = 0
      6     for i in data:
----> 7         if i['neighborhood'] == x and i['reviews'] >= 1 and i['overall_satisfaction'] >= 4:
      8             count += 1
      9             price += i['price']

TypeError: '>=' not supported between instances of 'str' and 'int'
```

`visualise_listings()` - Function to geographically visualise a given list of listings (15 marks)

NOTE

```
# To do any visualtion, the last part of the code you would normally do this..
    # some code to prepare data
    # some code to design the visualisation
    plt.show() #<< last line in code to show visualisation
```

With the new mplleaflet package, you can do the same thing, but just change from `plt.show()` to `mplleaflet.show()` .

DO NOT DO THIS

```
for i in data:
        # some codes here
        mplleaflet.show() # << Do not put the .show() function inside the for lo
op

    mplleaflet.show()# .show() should exist outside your for loop
```

Hint

1. Use the 2 functions you've created earlier and make 2 lists, latitude and longitude
2. Use .scatter() to plot the scatter plot
3. .show() the scatter plot </i>

Input: `airbnb_data as data` , a list of listing_ids

Output: Visualisation of locations the listings (nothing to return)

In [12]:

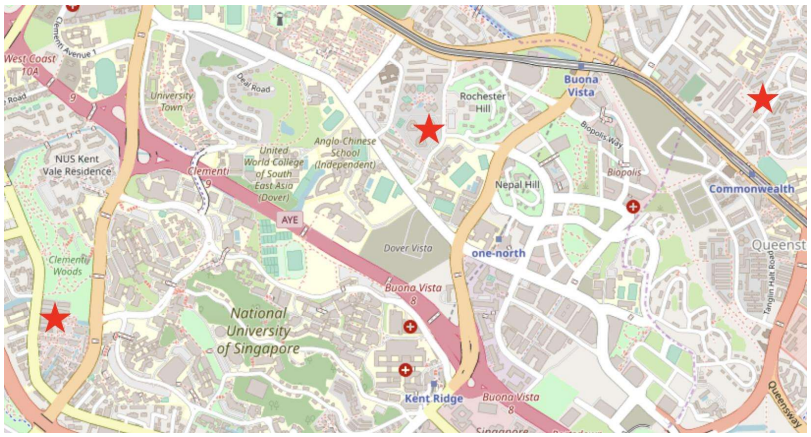
```
import mplleaflet
import matplotlib.pyplot as plt
#Write code below

def visualise_listings(data,y):
    latitude = []
    longitude = []
    for i in data:
        for j in y:
            if j == i['listing_id']:
                latitude.append(i['latitude'])
                longitude.append(i['longitude'])

    plt.scatter(longitude, latitude, s = 1000, color = 'r', marker = '*')
    mplleaflet.show()

# Correct
```

Tester Cell *- To test the above function to see if it's working.*

Expected Output:

Do not have to look exactly like this, as long as the locations are the same, it is fine!

If it's working, a new tab will pop out. This is normal.



In [13]:

```
visualise_listings(airbnb_data, ['10350448','13507262','13642646'])
```

Testing

Here, we will test if your functions are working as they are supposed to.

> ***Your task****: Use the functions created above, if necessary interchangeably, to answer the questions below.*

User - An Airbnb Host

Imagine now you're an Airbnb host and you are going to use the app you've developed to ask for a recommended price to list your place.

Based on your assigned neighborhood, what is the recommended price for your neighborhood (2.5 marks)

Expected output: *98.52*

In [18]:

```
neighborhood_to_test = 'TS23'

#Write code below

price_recommender(airbnb_data, neighborhood_to_test)

# Correct
```

Out[18]:

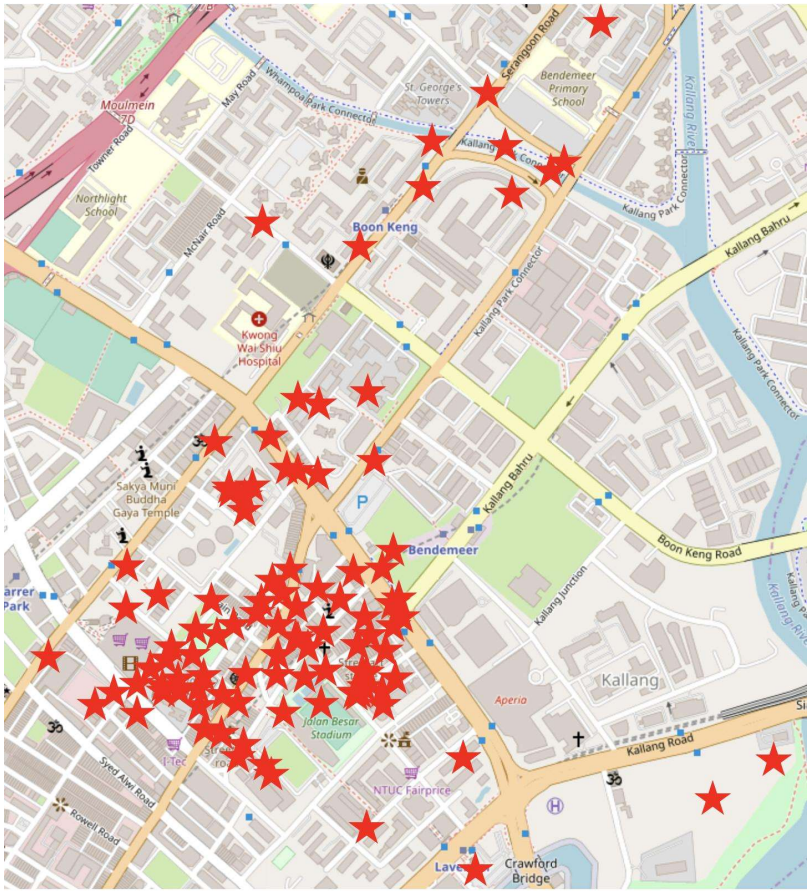
98.52

User - An Airbnb Guest

Imagine now you're an Airbnb guest and you are going to use the app to find a list of listings you want based on your search filter/restrictions.

Based on your assigned price, overall_satisfaction and neighborhood, using the functions created above and plot them out on a map (2.5 marks)

Expected output:



If it's working, a new tab will pop out. This is normal.

In [35]:

```
neighborhood_to_test = 'TS17'
price_to_test = 100
overall_satisfaction_to_test = 4

#Write code below

visualise_listings(airbnb_data, listings_recommender(airbnb_data, price_to_test, overal
l_satisfaction_to_test , neighborhood_to_test))

# Correct
```

In [14]:

```
import pandas as pd

df = pd.read_csv('airbnb_data.csv')
df[ df['overall_satisfaction'] >= 3.5]
```

Out[14]:

	listing_id	survey_id	host_id	room_type	country	city	borough	neighborhoo
0	1133718	1280	6219420	Shared room	NaN	Singapore	NaN	MK
1	3179080	1280	15295886	Shared room	NaN	Singapore	NaN	TS
7	10001615	1280	50960671	Shared room	NaN	Singapore	NaN	TS
8	12525278	1280	17526618	Shared room	NaN	Singapore	NaN	TS
9	13274598	1280	17526618	Shared room	NaN	Singapore	NaN	TS
10	12525276	1280	17526618	Shared room	NaN	Singapore	NaN	TS
11	3479493	1280	17526618	Shared room	NaN	Singapore	NaN	TS
12	3861984	1280	15745786	Shared room	NaN	Singapore	NaN	TS
17	8626676	1280	45368183	Shared room	NaN	Singapore	NaN	TS
18	3479446	1280	17526618	Shared room	NaN	Singapore	NaN	TS
20	9142805	1280	47611818	Shared room	NaN	Singapore	NaN	TS
24	10020525	1280	46545593	Shared room	NaN	Singapore	NaN	TS
25	8620667	1280	44803514	Shared room	NaN	Singapore	NaN	TS
31	10106290	1280	47611818	Shared room	NaN	Singapore	NaN	TS
32	16522880	1280	100353323	Shared room	NaN	Singapore	NaN	TS
33	9967697	1280	45368183	Shared room	NaN	Singapore	NaN	TS
35	1277919	1280	6954638	Shared room	NaN	Singapore	NaN	MK
39	8654132	1280	45368183	Shared room	NaN	Singapore	NaN	TS
40	15089063	1280	11441748	Shared room	NaN	Singapore	NaN	MK
41	14489603	1280	87731750	Shared room	NaN	Singapore	NaN	TS
42	12499007	1280	66697452	Shared room	NaN	Singapore	NaN	TS
63	10443717	1280	48110833	Shared room	NaN	Singapore	NaN	MK
75	8212422	1280	43228625	Shared room	NaN	Singapore	NaN	TS
83	9928402	1280	50960671	Shared room	NaN	Singapore	NaN	TS

	listing_id	survey_id	host_id	room_type	country	city	borough	neighborhoo
89	10716737	1280	43043281	Shared room	NaN	Singapore	NaN	MK
96	12367610	1280	65679130	Shared room	NaN	Singapore	NaN	TS
106	9738917	1280	50158481	Shared room	NaN	Singapore	NaN	TS
107	12211385	1280	65679130	Shared room	NaN	Singapore	NaN	TS
108	8245241	1280	43043281	Shared room	NaN	Singapore	NaN	MK
110	9055103	1280	46052159	Shared room	NaN	Singapore	NaN	TS
...	...	...	...	...	...	...	...	
7175	15209068	1280	96613568	Private room	NaN	Singapore	NaN	TS
7176	15208476	1280	96607622	Private room	NaN	Singapore	NaN	TS
7177	14532126	1280	81107861	Private room	NaN	Singapore	NaN	MK
7178	5993384	1280	21065429	Private room	NaN	Singapore	NaN	MK
7179	13984969	1280	39890192	Private room	NaN	Singapore	NaN	TS
7180	13998008	1280	51781892	Private room	NaN	Singapore	NaN	MK
7181	16715451	1280	110445631	Private room	NaN	Singapore	NaN	TS
7182	18274827	1280	101367456	Private room	NaN	Singapore	NaN	MK
7183	7843168	1280	40912923	Private room	NaN	Singapore	NaN	MK
7184	5609608	1280	28954164	Private room	NaN	Singapore	NaN	MK
7185	17928326	1280	18893050	Private room	NaN	Singapore	NaN	MK
7187	9635192	1280	49380207	Private room	NaN	Singapore	NaN	TS
7188	16504122	1280	94765000	Private room	NaN	Singapore	NaN	TS
7189	15120568	1280	41270193	Private room	NaN	Singapore	NaN	MK
7191	17595630	1280	30031571	Private room	NaN	Singapore	NaN	MK
7195	15826584	1280	96249364	Private room	NaN	Singapore	NaN	MK
7196	12138455	1280	59279765	Private room	NaN	Singapore	NaN	TS
7197	12658514	1280	49222064	Private room	NaN	Singapore	NaN	MK

	listing_id	survey_id	host_id	room_type	country	city	borough	neighborhoo
7199	9260208	1280	23374252	Private room	NaN	Singapore	NaN	MK
7200	17709076	1280	3507099	Private room	NaN	Singapore	NaN	TS
7201	18011660	1280	78238544	Private room	NaN	Singapore	NaN	MK
7202	17142437	1280	33096269	Private room	NaN	Singapore	NaN	MK
7203	15314063	1280	96607622	Private room	NaN	Singapore	NaN	TS
7204	16538491	1280	108819422	Private room	NaN	Singapore	NaN	MK
7205	18219103	1280	125804276	Private room	NaN	Singapore	NaN	MK
7207	16621513	1280	107624021	Private room	NaN	Singapore	NaN	TS
7208	3775738	1280	2150632	Private room	NaN	Singapore	NaN	TS
7209	17616042	1280	118900770	Private room	NaN	Singapore	NaN	MK
7210	13039292	1280	72105933	Private room	NaN	Singapore	NaN	TS
7211	17741336	1280	51275099	Private room	NaN	Singapore	NaN	MK

3098 rows × 19 columns

In [ ]: