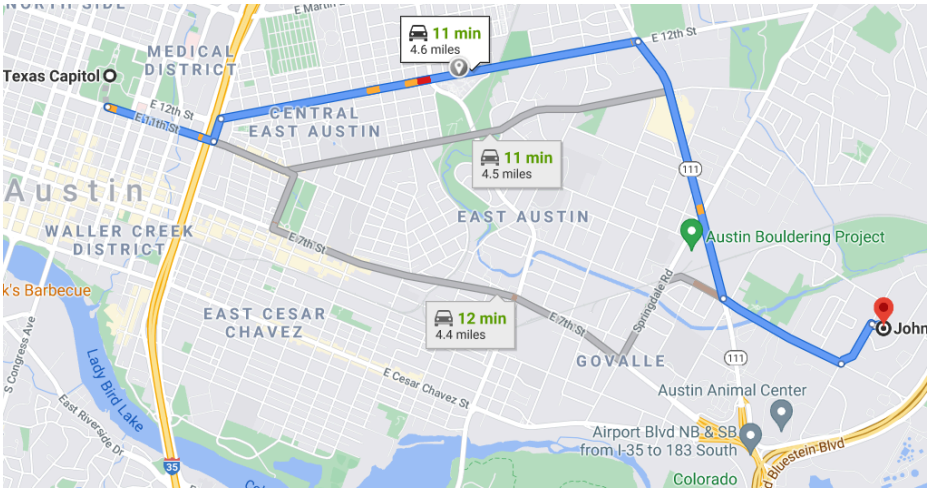
Driving Distances



We can use Python to determine how far and how long it takes to drive between any two locations. One way to do this is to use the OSRM - **O**pen **S**ource **R**outing **M**achine - website's API.

Using OSRM

The OSRM website has an **API** (**A**pplication **P**rogramming **I**nterface) that enables a Python program to request route information from the website. Our program will provide the longitude and latitude of two locations in a request to the API and then will extract the driving distance and driving time from the request's response. For Python to use the OSRM API, you will need to make a proper request to the website and then convert the response into a Python dictionary.

Make sure you close your IDE (Thonny or Idle) and then follow the instructions on how to install the `requests` library on your PC. Open Python Idle and put the following at the top of your program to import the required libraries (the json library does not need to be installed).

```
import requests
import json
```

To make a request to the OSRM API you must use the **`requests.get(`*`string`*`)`** method. The ***`string`*** contains the URL with the latitude & longitude of the two locations. We will use a Python f-string for our ***`string`***. The magic of f-strings is that you can embed Python expressions directly inside them. Any portion of an f-string that's enclosed in curly braces {} is treated as an expression that is evaluated and converted to a string representation. We'll use the following ***`f-string`*** with OSRM where our "expressions" are simply the variables **`lon1`**, **`lat1`**, **`lon2`**, **`lat2`** which simply evaluate to their values.

```
f'http://router.project-osrm.org/route/v1/car/{lon1},{lat1};{lon2},{lat2}?overview=false'
```

Let's test the OSRM API by finding the driving distance and duration between UT Austin and Texas A&M. To make a request from the API all you need is a correctly formed URL. This URL makes a request to the API for the route between the University of Texas and Texas A&M and returns a JSON object to your browser. If you look at the end of the URL you'll see two sets of numbers representing the latitude & longitude of both universities.

The APIs response to a request is a json object (which looks just like a Python dictionary). If you look closely at the response you'll see the "distance" and "duration" key and their values.

To make this request to the OSRM API **from your python program**, first initialize these 4 variables to the coordinates of the two schools.

```
lat1 = 30.282621962032763   # UT Austin
lon1 = -97.73013768586304   # UT Austin
lat2 = 30.609873932278347   # Texas A&M
lon2 = -96.34040026273189   # Texas A&M
```

Now make the request to the OSRM API and store the website's response in the response variable by using the below assignment statement - be sure to put the full f-string shown above in-between the parenthesis of the requests get() method.

```
response = requests.get(f'...')
```

The content of the website's response is a json object, which we can easily convert to a python **dictionary** by using the **json.loads()** method.

```
responseDict = json.loads(response.content)
```

We then access the dictionaries "**routes**" key to retrieve the **list** of routes. Finally, we use **[0]** to get the first route in that **list**.

```
routes = responseDict["routes"]
route = routes[0]
```

Print out the **route** variable. You should see that **route** contains a **dictionary**. Print out the values stored in dictionary's "**distance**" and "**duration**" keys. The distance is in meters and the duration is in seconds. Make sure the distance & duration you get make sense for the trip between UT and A&M (HINT: Google meters to miles conversion).

Driving Distances #1

Write a program that determines the routes between ***LASA*** and all of the other locations shown below.

LASA HS	30.258334759602516	-97.68177086019294
LBJ HS	30.31367095875231	-97.65665627368612
Bowie HS	30.187838958159965	-97.85851876019456
Anderson HS	30.37510126060306	-97.75188828902523

Austin HS	30.273705933308037	-97.7665524123247
Ann Richards HS	30.238301839328155	-97.78845740252305
TX Capitol	30.274841228454136	-97.74032904485188

Copy the ***list of tuples*** below which contains all of the above data. Each tuple contains the name of the location, as well as the location's latitude and longitude. You should be able to add more locations to the list without changing any other part of your code.

```
locations = [("LASA HS",30.258334759602516, -97.68177086019294),
             ("LBJ HS",30.31367095875231, -97.65665627368612),
             ("Bowie HS",30.187838958159965, -97.85851876019456),
             ("Anderson HS",30.37510126060306, -97.75188828902523),
             ("Austin HS",30.273705933308037, -97.7665524123247),
             ("Ann Richards HS",30.238301839328155, -97.78845740252305),
             ("TX Capitol",30.274841228454136, -97.74032904485188)]
```

Use this formatted string when you print the **Sorted by distance** output

f' {distance:5.2f} miles {duration:5.2f} minutes from {loc1} to {loc2}'

Use this formatted string when you print the **Sorted by duration** output

f' {mins:2d} min {secs:2d} sec {distance:5.2f} miles from {loc1} to {loc2}'

The information after the **:** indicate both the data type expected (d=decimal aka integer, f=float) and the total width (i.e. number of digits) as well as in the case of a float how many digits after the decimal point the numbers should have (specified by the digit after the decimal point).

Your program output should look as follows (values may differ slightly)

```
Sorted by distance (time in decimal minutes)
  4.76 miles 10.60 minutes from LASA HS to TX Capitol
  6.09 miles 13.86 minutes from LASA HS to LBJ HS
  8.48 miles 18.12 minutes from LASA HS to Austin HS
 10.75 miles 16.52 minutes from LASA HS to Ann Richards HS
 11.42 miles 22.00 minutes from LASA HS to Anderson HS
 17.17 miles 25.17 minutes from LASA HS to Bowie HS
Sorted by duration (time in min & sec)
 10 min 36 sec  4.76 miles from LASA HS to TX Capitol
 13 min 51 sec  7.06 miles from LASA HS to LBJ HS
 16 min 31 sec 10.75 miles from LASA HS to Ann Richards HS
 18 min  7 sec  8.48 miles from LASA HS to Austin HS
 22 min  0 sec 11.42 miles from LASA HS to Anderson HS
 25 min 10 sec 17.17 miles from LASA HS to Bowie HS
```

Submit your program using the assignment name **distances1**

Driving Distances #2

Now have your program find the distance between **any two** of the locations. Note that the distance and duration usually differs depending whether you go from location A to location B or from location B to location A.

Your program output should look as follows (values may differ slightly)

```
Sorted by distance (time in decimal minutes)
  2.47 miles  5.85 minutes from Austin HS to TX Capitol
  4.40 miles 10.67 minutes from Ann Richards HS to TX Capitol
  4.40 miles 10.98 minutes from TX Capitol to Ann Richards HS
     ⋮      ⋮      ⋮
 22.03 miles 30.39 minutes from Bowie HS to LBJ HS
Sorted by duration (time in min & sec)
  5 min 51 sec  2.47 miles from Austin HS to TX Capitol
  7 min 20 sec  4.70 miles from Ann Richards HS to Austin HS
  8 min 42 sec  4.82 miles from TX Capitol to Austin HS

     ⋮      ⋮      ⋮
 30 min 59 sec 18.62 miles from LBJ HS to Bowie HS
```

Submit your program using the assignment name **distances2**

Find the location within the AISD boundary which looks to be the furthest and/or longest drive away from LASA and add it to your list of locations. Let's see which location is the furthest away from LASA. I found one that is **23.97 miles from LASA**

If you want, you can see how your home address compares as well.

NOTE: You can obtain the latitude and longitude of a location by right-clicking on the location in Google Maps.