Up to four points reduced from your driving record

A discount of up to 10% on collision and liability insurance rates

The confidence of being a better, safer driver

Safe driving parameters

Speed, Cornering, Time, Acceleration & Braking

**gas mileage**

Get the miles traveled from the trip odometer, or subtract the original odometer reading from the new one.

Divide the miles traveled by the amount of gallons it took to refill the tank. The result will be your car's average Miles Per Gallon yield for that driving period.

e.g. 312miles ÷ 16gallons = 19.5mpg

import prathapani\_proj1 as pp

pp.func3('alicedata.json')

a=pp.func7('alicedata.json')

from matplotlib import

pylab as pl

import numpy as np

import json

data = []

def func1():

with open('alicedata.json') as json\_file:

for line in json\_file:

global data

data.append(json.loads(line))

#print(line)

return data

e1=func1()

def func2(e1):

from pprint import pprint

for line2 in e1[:10]:

pprint(line2)

func2(e1)

def func3(e1):

lst=[]

lst1=[]

lst\_1=[]

count = {}

for i in e1:

if not i['name'] in lst:

lst.append(i['name'])

print(lst)

for i in e1:

lst1.append(i['name'])

for word in lst1:

if word in count:

count[word]+= 1

else:

count[word] = 1

print(count)

'''

from collections import Counter

c=Counter(lst1)

print(c)

'''

inp=input("Enter a SIGNAL NAME from the following: --> 'vehicle\_speed','accelerator\_pedal\_position', 'engine\_speed', 'torque\_at\_transmission', 'latitude', 'longitude', 'steering\_wheel\_angle', 'fuel\_consumed\_since\_restart', 'odometer','fuel\_level', 'brake\_pedal\_status', 'transmission\_gear\_position'\n")

for i in e1:

lst1.append(i['name'])

if(i['name']== inp):

lst\_1.append(i['value'])

print("The minimum value of",inp, "is" ,min(lst\_1),"\n","The maximum value of",inp,"is",max(lst\_1))

print("number of occurrences and value range of ",inp," is:",count[inp])

func3(e1)

def func4(e1):

lst1=[]

lst2=[]

lst\_1=[]

lst\_2=[]

inp='latitude'

inp1='longitude'

for i in e1:

if(i['name']== 'latitude'):

lst\_1.append(i['value'])

print("1st and last latitude of recorded data:",lst\_1[-1],lst\_1[0])

lat1=lst\_1[0]

lat2=lst\_1[-1]

for i in e1:

if(i['name']== 'longitude'):

lst\_2.append(i['value'])

print("1st and last longitude of recorded data:",lst\_2[-1],lst\_2[0])

lon1=lst\_2[0]

lon2=lst\_2[-1]

from math import radians, cos, sin, asin, sqrt

#def distancelatlon(lon1, lat1, lon2, lat2):

"""

Calculate the great circle distance between two points

on the earth (specified in decimal degrees)

"""

# convert decimal degrees to radians

lon1, lat1, lon2, lat2 = map(radians, [lon1, lat1, lon2, lat2])

#latitude-longitude formula

dlon = lon2 - lon1

dlat = lat2 - lat1

a = sin(dlat/2)\*\*2 + cos(lat1) \* cos(lat2) \* sin(dlon/2)\*\*2

c = 2 \* asin(sqrt(a))

miles = 3956.2691\* c

for i in e1:

if(i['name']== 'vehicle\_speed'):

lst2.append(i['timestamp'])

print("total trip time period:",lst2[-1]-lst2[0],"seconds")

print("The vehicle trip distance over recorded data:", miles," miles")

func4(e1)

def func5(e1):

lst1=[]

lst2=[]

lst3=[]

lst4=[]

lst5=[]

lst6=[]

lst7=[]

lst8=[]

lst9=[]

lst10=[]

lst11=[]

lst12=[]

lst\_1=[]

from matplotlib import pylab as pl

from matplotlib import pyplot as plt

fig = plt.figure()

import numpy as np

for i in e1:

lst\_1.append(i['timestamp'])

for i in e1:

if(i['name']== 'vehicle\_speed'):

lst1.append(i['value'])

pl.plot(lst\_1[0:len(lst1)],lst1)

pl.show()

#fig.suptitle('vehicle speed vs trip time', fontsize=20)

#plt.xlabel('trip time', fontsize=18)

#plt.ylabel('vehicle speed', fontsize=18)

for i in e1:

if(i['name']== 'accelerator\_pedal\_position'):

lst2.append(i['value'])

pl.plot(lst\_1[0:len(lst2)],lst2)

pl.show()

for i in e1:

if(i['name']== 'engine\_speed'):

lst3.append(i['value'])

pl.plot(lst\_1[0:len(lst3)],lst3)

pl.show()

for i in e1:

if(i['name']== 'torque\_at\_transmission'):

lst4.append(i['value'])

pl.plot(lst\_1[0:len(lst4)],lst4)

pl.show()

for i in e1:

if(i['name']== 'latitude'):

lst5.append(i['value'])

pl.plot(lst\_1[0:len(lst5)],lst5)

pl.show()

for i in e1:

if(i['name']== 'longitude'):

lst6.append(i['value'])

pl.plot(lst\_1[0:len(lst6)],lst6)

pl.show()

for i in e1:

if(i['name']== 'steering\_wheel\_angle'):

lst7.append(i['value'])

pl.plot(lst\_1[0:len(lst7)],lst7)

pl.show()

for i in e1:

if(i['name']== 'fuel\_consumed\_since\_restart'):

lst8.append(i['value'])

pl.plot(lst\_1[0:len(lst8)],lst8)

pl.show()

for i in e1:

if(i['name']== 'odometer'):

lst9.append(i['value'])

pl.plot(lst\_1[0:len(lst9)],lst9)

pl.show()

for i in e1:

if(i['name']== 'fuel\_level'):

lst10.append(i['value'])

pl.plot(lst\_1[0:len(lst10)],lst10)

pl.show()

for i in e1:

if(i['name']== 'brake\_pedal\_status'):

lst11.append(i['value'])

pl.plot(lst\_1[0:len(lst11)],lst11)

pl.show()

'''

#transmission gear position is a string and hence cannot be plotted

for i in e1:

if(i['name']== 'transmission\_gear\_position'):

lst12.append(i['value'])

pl.plot(lst\_1[0:len(lst12)],lst12)

pl.show()

'''

func5(e1)

def func6(e1):

inp='vehicle\_speed'

lst=[]

lst1=[]

lst\_1=[]

count = {}

for i in e1:

if(i['name']== 'vehicle\_speed'):

lst1.append(i['value'])

for i in e1:

lst.append(i['name'])

for word in lst:

if word in count:

count[word]+= 1

else:

count[word] = 1

print("The minimum value of vehicle\_speed is" ,min(lst1),"\n","The maximum value of vehicle\_speed is",max(lst1))

print("number of occurrences of vehicle\_speed is:",count[inp])

print("Average speed of Alice's vehicle:",sum(lst1)/count[inp],"miles/hour")

print("Maximum speed of Alice's vehicle:",max(lst1),"miles/hour")

func6(e1)

def func7(e1):

lst\_1=[]

lst1=[]

lst2=[]

import pygmaps

import webbrowser

for i in e1:

lst\_1.append(i['timestamp'])

for i in e1:

if(i['name']== 'latitude'):

lst1.append(i['value'])

for i in e1:

if(i['name']== 'longitude'):

lst2.append(i['value'])

mymap = pygmaps.maps(lst1[int((len(lst1))/2)], lst2[int((len(lst2))/2)], 16)

path = list(zip(lst1,lst2))

mymap.addpath(path,"#00FF00")

mymap.draw('./mymap.html')

func7(e1)