Python Project Code Submission

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt


df_cities = pd.read_csv('cities.csv')
df_cities.head()

def total_distance(dfcity,path):
    prev_city = path[0]
    total_distance = 0
    step_num = 1
    for city_num in path[1:]:
        next_city = city_num
        total_distance = total_distance + np.sqrt(pow((dfcity.X[city_num] -
dfcity.X[prev_city]),2) + pow((dfcity.Y[city_num] - dfcity.Y[prev_city]),2))
        prev_city = next_city
        step_num = step_num + 1
    return total_distance



dumbest_path = list(df_cities.CityId[:].append(pd.Series([0])))
print('Total distance with the dumbest path is '+
"{:,}".format(total_distance(df_cities,dumbest_path)))



#Using the sorted order of X and Y coordinates
sorted_cities = list(df_cities.iloc[1:,].sort_values(['X','Y'])['CityId'])
sorted_cities = [0] + sorted_cities + [0]
print('Total distance with the sorted city path is '+
"{:,}".format(total_distance(df_cities,sorted_cities)))




#Using Sorted Cities within a grid of squares
df_cities['Ycuts'] = pd.cut(df_cities.Y,500)
df_cities['Xcuts'] = pd.cut(df_cities.X,500)
grid_sorted_cities =
list(df_cities.iloc[1:].sort_values(['Xcuts','Ycuts','X','Y'])['CityId'])
grid_sorted_cities =  [0] + grid_sorted_cities + [0]
print('Total distance with the sorted cities with a grid size of 500*500 is '+
"{:,}".format(total_distance(df_cities,grid_sorted_cities)))



#Nearest Neighbour/greedy Algorithm
def nearest_neighbour():
    cities = pd.read_csv("../input/cities.csv")
    ids = cities.CityId.values[1:]
    xy = np.array([cities.X.values, cities.Y.values]).T[1:]
    path = [0,]
    while len(ids) > 0:
        last_x, last_y = cities.X[path[-1]], cities.Y[path[-1]]
        dist = ((xy - np.array([last_x, last_y]))**2).sum(-1)
        nearest_index = dist.argmin()
        path.append(ids[nearest_index])
```

```
        ids = np.delete(ids, nearest_index, axis=0)
        xy = np.delete(xy, nearest_index, axis=0)
    path.append(0)
    return path



nnpath = nearest_neighbour()
print('Total distance with the Nearest Neighbor path '+  "is
{:,}".format(total_distance(df_cities,nnpath)))
```

Python Turtle Graphics

```
import random

import colours as colours
import pandas as pd
import turtle
import math


colours = ["yellow", "gold", "orange", "red", "maroon", "violet", "magenta", "purple",
"navy", "blue", "skyblue", "cyan", "turquoise", "lightgreen", "green", "darkgreen",
"chocolate", "brown", "black", "gray"]
dataset_cities = pd.read_csv('cities.csv')
# print(dataset_cities['X'][1])
# print(dataset_cities['Y'][1])


wn = turtle.Screen()  # setup screen and its attributes
tess = turtle.Turtle()  # sets tess
wn.title("Path Optimiser")
tess.speed(1000)
tess.setpos(0, 0)



#going through all the coordinates
for ite in range(100000):
  print("Latitude: %f"  %dataset_cities['X'][ite])
  print("Longitude: %f"  %dataset_cities['Y'][ite])
  tess.setpos(dataset_cities['X'][ite]/10 - 200,dataset_cities['Y'][ite]/10 - 200)
  tess.color(random.choice(colours))
  tess.dot(10)

wn.exitonclick()
```