Practical Work N1

Multimedia and Computer Graphics

Escola Superior de Tecnologia e Gestão
Politécnico de Coimbra

Objectives:

- Creating one application with several functions

Requirements:

- Python 3.9, matplotlib, networkx and pandas

Description

This work consists of three exercises that correspond to a Python application where it is necessary that there is a menu for the user to be able to select the exercise to be executed.

Part I: Modeling the fishing pond population

Suppose we manage a fishing pond that contained a population of 12000 large mount bass on January 1 of this year. With no fishing, the bass population is expected to grow at a rate of 8% per year, which incorporates both the birth rate and death rate of the population. The maximum annual fishing harvest allowed is 1500.

$\boldsymbol{P_{n+1} = r * P_n - h}$ - *where P is the population, r is the annual grow rate and h is the harvest*

a) Write the algorithm to calculate the pond population.

b) Implement a function called "pond" that returns the population after a number of years. The number of years must be an input parameter of the function and requested to the user.

c) Print the information about the population in the next 10 years.

d) Use the pond function to answer the following questions. Is this maximum harvest sustainable? If not, how long the fish population dies out? Should the pond manager reduce the maximum harvest? If so, what should it be reduced to?

e) Generalize the pond function with two additional parameters, the initial population size, and the annual harvest. Using your modified function, compute the number of fish that will be in the pond in the next 15 years if we change the maximum harvest to 800.

f) Generalize the pond function so that it also takes the annual growth rate as parameters and to allow the pond to be annually restocked with an additional quantity of fish.

Part II – 2D data visualization

a) Compute a 2D plot with the evolution of the population for the number of years introduced by the user. The y values correspond to the population size and the x values correspond to the number of years.

b) Compute a 2D plot with the evolution of the population size until it is zero. You should also print the number of years is necessary to this event to occur.

c) The charts title, axis labels and line color of a) and b) exercises should be configurable by the user.

Part III – 3D data visualization

Create a 3D graph using the plot_trisurf method from the Matplotlib library. The data that should be used for the X, Y and Z axis correspond to the existing columns in the fine denominated "data.csv". This file is available in the course support material. The output is represented in Figure 1.

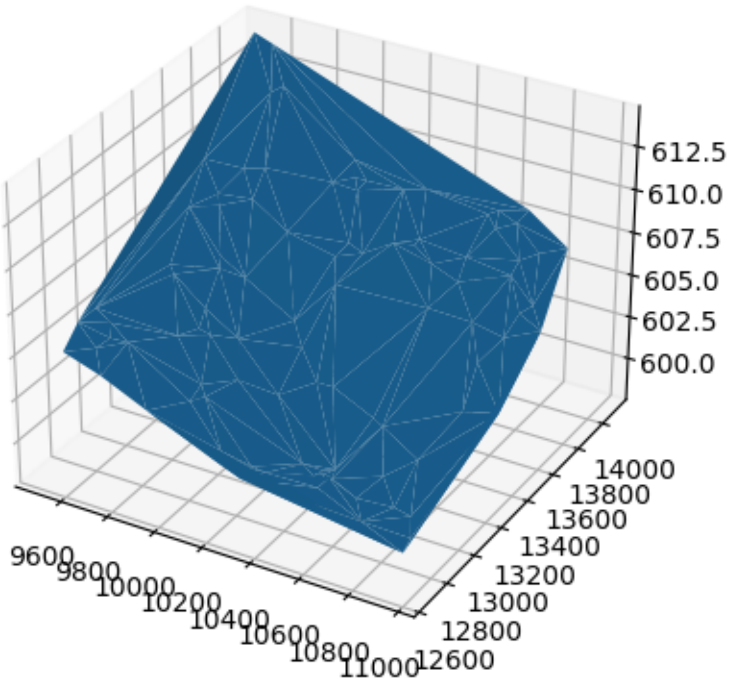
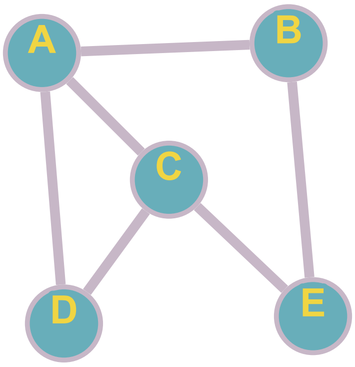


Figure 1 - 3D graph

Part IV – Graph theory:

A graph can be represented in a file by listing once link per line, with each link represented by a pair of nodes. For example, the graph below is represented by the file on the right. Notice that, personalization preferences will be valorized in the score of the exercise.



graph.txt

```
A C
A B
A D
B E
C D
C E
```

Figure 2 – Graph and corresponding txt file.

a) Write a function that reads the file and creates the graph. The user should have the possibility to decide if the graph is directed or undirected. This function should load any file that respect the format of Figure 2.

b) Print the adjacency matrix for the graph.

c) Compute the basic information about the graph: number of nodes and edges.

d) Calculate the average degree of the graph.

e) Calculate the density of the graph.

f) Compute planarity and adjacency matrix.

g) Compute the shorted path for the input nodes entered by the user.

h) Create a method so that the user can update the weight and add other edges to this graph dynamically.

Part V: Data and Statistics with Python

USGS earthquake data is available in many formats, the simplest of which is a tabular file called CSV, short for "comma-separated values. CSV files contain one row of text per line, with columns separated by commas. A CSV file containing data about all the earthquakes in the past 30 days is available at:

http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_month.csv

a) Count the number of missing values in this dataset and remove the lines with missing data.

b) Compute the descriptive statistics of the data.

c) Using matplotlib create a 2D graph with the longitude data as the x values and the latitude data as the y values. Hint: use a scatter plot to represent the data. The output is represented in Figure 3.

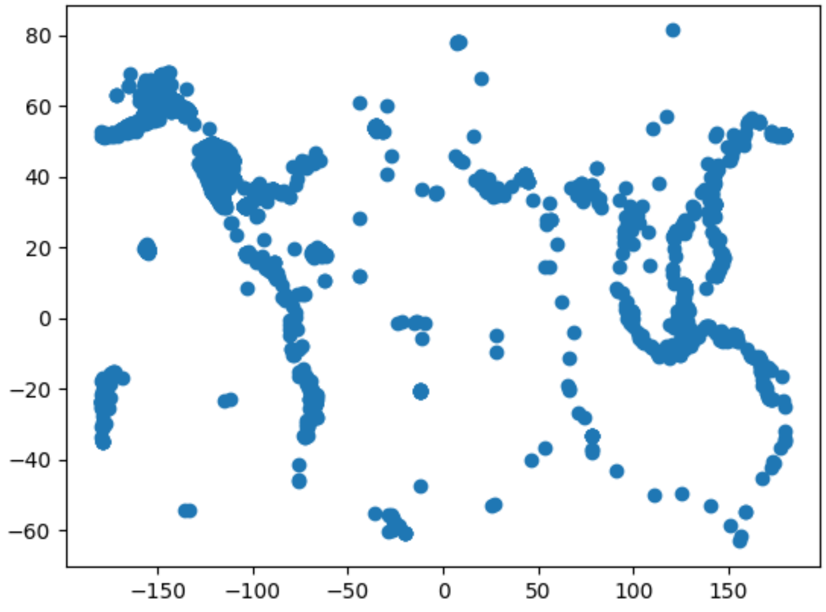


Figure 3 - Output of exercise c)

d) Modify the exercise c) to use a different color according to the depth of the earthquake. Shallow (< 10 km deep) earthquakes will be yellow, medium depth (>10 and <50 km deep) earthquakes will be red and deep earthquakes (>50 km) will be blue. The output is represented in Figure 4.

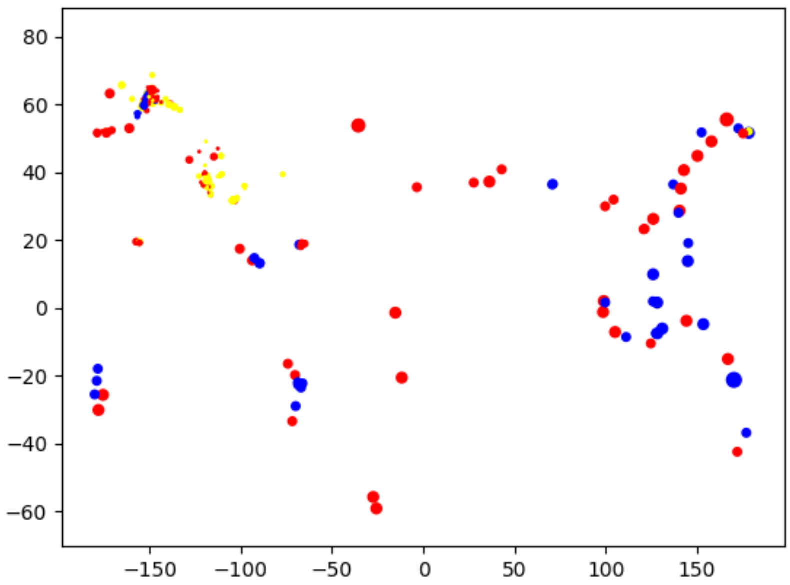


Figure 4 - Output of exercise d)

e) Modify the exercise d) to read the magnitude and draws larger circles for higher magnitude earthquakes. The size of each point should be the square of the magnitude of the corresponding earthquake. The output is represented in Figure 5.

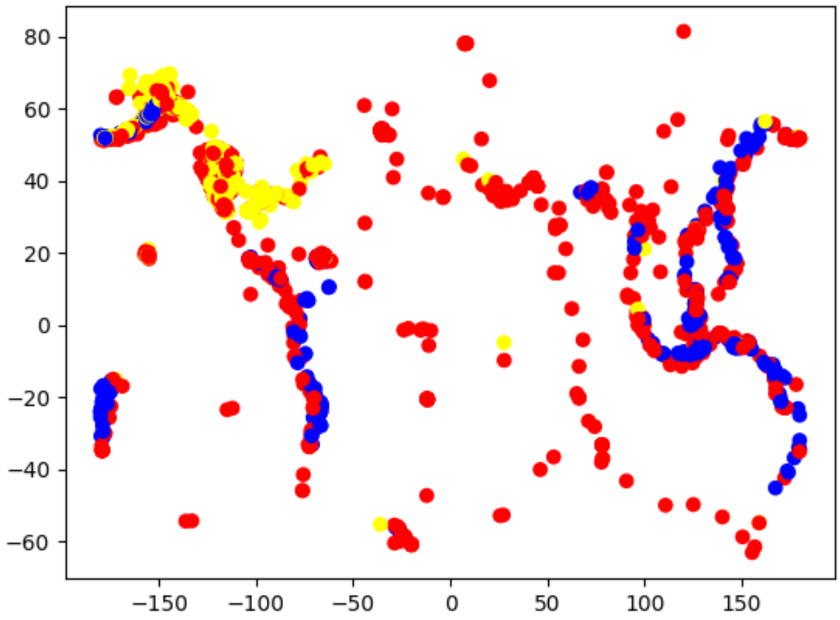


Figure 5 - Output of exercise e)

Notice that, exercises c), d) and e) will not show a map background. However, if you would like to add it, look into the Basemap class from the mpl_toolkits.basemap module. Kindly check Figure 6.

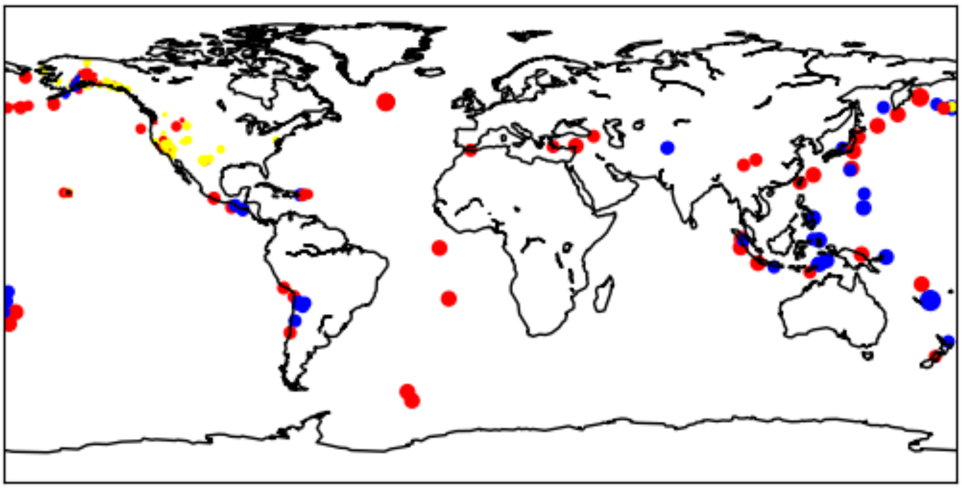


Figure 6 - Output with map background

<u>Delivery date</u> - 23:59, 30 October 2022

In this work, the authors must deliver one Python application with a menu interface to choose the exercise to be executed and a report. The application should have exception treatment and validate every input of the user.

This documentation should be technical and justify the selected options.

The code should be commented.

The report should include:

- Introduction
- Methods
- Results with applications tests
- Discussion with a proper and critical justification of the choices taken in carrying out the work
- The time spent with the subject *per week* since the beginning of the semester: attendance in classes and extra-class.
- Conclusion
- References

A zip file with the report in PDF format and the Python scripts should be submitted.

Important note: The work after submission will be defended by the author.

The submission of this work without oral defense will not be evaluated.

The score will only be provided after oral defense.