Draw Stars

psp-08-02

1 Overview

Astronomers collect lots of data about stars and there are many catalogs that identify the locations of stars. In this lab, you will use data from a star catalog to create a picture that plots the locations of stars. This lab will give you practice creating, modifying, and extracting data from dictionaries, and give you practice working with image and text files.

2 Learning Outcomes

By the end of this project students should be able to:

- read and write programs that create, modify, and extract data from dictionaries;
- read and write programs that work with both image and text files;
- work effectively with a partner using pair-programming;
- write an effective report that describes the students' problem solving process.

3 Pre-Lab Instructions

Do this part before coming to lab:
In this lab you are going to be implementing a function that reads data from text files and stores that data in dictionaries.

- Read Problem Space Chapter 8: Input and Output;
- Read through the entire lab;
- Look through Python's turtle graphics api.
  https://docs.python.org/3.4/library/turtle.html
- Practice drawing with turtle graphics by making some basic shapes;
  Here is an example:

```
import turtle

# A simple right-sided triangle
turtle.forward(90)
turtle.left(135)
turtle.forward(90)
turtle.left(90)
turtle.forward(90)
turtle.left(135)
turtle.forward(90)
turtle.hideturtle()
turtle.exitonclick()
```

- Create pseudo-code for each of the functions described in the tables below;
- Download and save the following text files into a folder that will be accessible in lab. Alternatively, these files are available in BBLearn in your lab section under lab files.

  Star Catalog

  stars.txt

  Constellation files (need for extra credit only)

  - Cassiopeia
  - Cygnet
  - Big Dipper
  - Bootes
  - Gemini
  - Hydra
  - Ursa Major
  - Ursa Minor

- Be prepared to show your pseudo-code to the lab aide at the beginning of lab.

4 Lab Instructions

Do this part in lab:
The file stars.txt you downloaded in the pre-lab contains one line for each star that is represented in the catalog. The meaning of each field (column) is described below.

- The first three fields are the x, y and z coordinates for the star. We will ignore the z coordinate, and use only the x and y coordinates. Each axis in the coordinate system goes from -1 to +1, and the centre point is 0,0. (See the figure below.)
- The fourth field is the Henry Draper number, which is simply a unique identifier for the star.
- The fifth field is the magnitude (or brightness) of the star.
- The sixth field is Harvard Revised number, another identifier.
- The seventh field exists only for a small number of stars and is a semicolon-separated list of names for a star. A star may have several names.

The fields that you will need for this assignment include the x and y coordinates, the magnitude, the Henry Draper number, and the name (or names) of each star.

Part 1: Plotting star charts

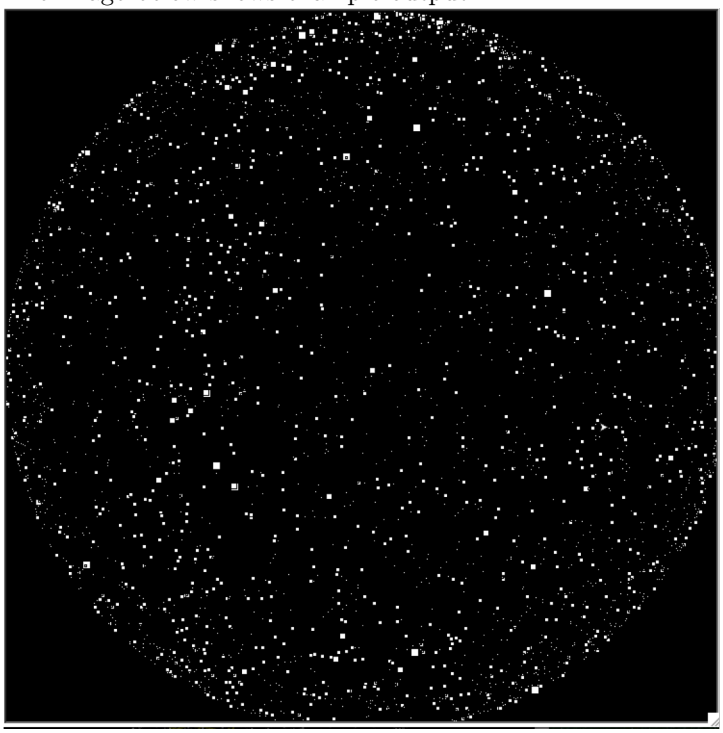
This project will use Python's turtle graphics package to draw your star chart and constellations. You will need to include the statement 'import turtle' at the top of your project.
Turtle graphics can be a little slow, you can turn off the tracing to make your drawing appear much faster by using the command turtle.tracer(0).
If you still want to see the progress of the drawing, you can also just speed up the turtle by using the command turtle.speed(10).

Implement the following functions in plot_stars.py:

Function Name	Description
read_coords(file)	Given an open text file that contains a star catalog as specified above, return three dictionaries using a tuple. The first is keyed on the Henry Draper number and the values are tuples containing the x and y coordinates of each star. The second dictionary is also keyed on the Henry Draper numbers and contains the magnitudes (float) of the stars. The third dictionary is keyed on the names of the stars and the values are the Henry Draper numbers. If a star has more than one name, each name will appear as a key in the third dictionary. If a star does not have a name it will not be represented in the third dictionary. HINTS: (1) You can return multiple dictionaries by using a tuple. (2) Star names may contain spaces, thus they are separated using semicolons.

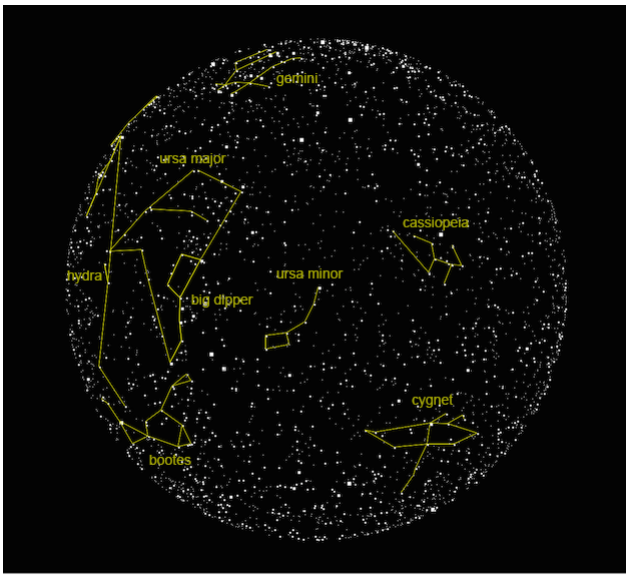
plot_plain_stars(picture_size, coordinates_dict)	Given the size of the the picture in pixels, and a dictionary containing the x,y coordinates of stars as values in the dictionary, plot the stars and have turtle draw the picture. Be sure to map the coordinate values (i.e., -1 to 1) approproiately based on the picture size you plan to use. The middle of the picture is considered (0,0). If you plot x,y values without an appropriate value all starts will appear in the very middle of the screeen. Set the picture background to black. The stars are plotted as 2x2 pixel white-filled rectangles. (The fillcolor() function in the turtle module will help you out.) This funcion is to help you get started and it will be replaced with the plot_by_magnitude function below.
plot_by_magnitude(picture_size, coordinates_dict, magnitudes_dict)	Given a the size of the the picture in pixels, a dictionary containing the x,y coordinates, and a dictionary containing the magnitudes, plot the stars and have turtle draw the picture. Set the picture background to black. In this case the stars are plotted as white-filled rectangles, but the size of the rectangle is a function of the magnitude of the stars. Since brighter stars have smaller magnitude values, you will calculate the size of the rectangle as follows: star_size = round$\left(\frac{10.0}{(magnitude+2)}\right)$. Limit the size of the star to 8 pixels or the scaled value whichever is smaller.

The image below shows example output.



Part 2: EXTRA CREDIT : Drawing Constellations

The image below is an example of the output of your work in this part of the assignment. The constellations in the example have been labeled, but your output is not required to have labels. **Reminder: turtle graphics can be a little slow, you can turn off the tracing to make your drawing appear much faster by using the command turtle.tracer(0)**



We are providing several text files that describe constellations. Each file contains two star names per line of the file. Each pair of star names represents one line to be drawn for the constellation. You will write a function to read this data into a dictionary, and add the lines between the stars on to a picture containing a star map.

Constellation files

Using the constellation data from the files you downloaded in the pre-lab, implement the following functions in plot_stars.py:

Function Name	Description
read_constellation_lines(file)	Given an open text file that contains constellation data in the form described above, return a dictionary with star names of type str as keys and lists of star names as values. For each key, the associated list contains all stars connected by a line to the key star.
plot_constellations(pic_size, star_names, star_coords, constellations)	Given the following arguments: a dictionary of coordinates where the keys are Henry Draper numbers, values are a tuple of the x, y coordinates. A dictionary containing the lines between stars. A dictionary containing the mapping between star name and the Henry Draper numbers The constellation should be drawn in a non-white color.

When you have completed the lab run pep8 against your code until all formatting errors have been corrected and your code is PEP 8 compliant. See the Getting Started lab if you need instructions on running the program, or the pep8 documentation found here.

5 Lab Report

Each pair of students will write a single lab report together and each student will turn in that same lab report on BBLearn. Submissions from each student on a pair should be identical.

Your lab report should begin with a preamble that contains:

- The lab assignment number and name
- Your name(s)
- The date
- The lab section

It should then be followed by four numbered sections:

1. Problem Statement

In this section you should describe the problem in **your** own words. The problem statement should answer questions like:

- What are the important features of the problem?
- What are the problem requirements?

This section should also include a reasonably complete list of requirements in the assignment. Following your description of the problem, include a bulleted list of specific features to implement. If there are any specific funtions, classes or numeric requirements given to you, they should be represented in this bulleted list.

2. Planning

In the second section you should describe what planning you did in order to solve the problem. You should include planning artifacts like sketches, diagrams, or pseudocode you may have used. You should also describe your planning process. List the specific data structures or techniques you plan on using, and why.

3. Implementation and Testing

In the third section you should describe how you implemented your plan. As directed by the lab instructor you should (as appropriate) include:

- a copy of your source code (Submitted in BBLearn as a .py file)
- a screen shot of your running application / solution
- results from testing

4. Reflection

In the last section you should reflect on the project. Consider different things you could have done to make your solution better. This might include code organization improvements, design improvements, etc.

You should also ask yourself what were the key insights or features of your solution? Were there alternative approaches or techniques you could have employed? How would these alternatives have impacted a different solution?

5. Partner Rating

Every assignment you are required to rate your partner with a score -1, 0 or 1. This should be submitted in the comment section of the BBLearn submission, and not in the report document. You do not have to tell your partner the rating you assign them. A rating of 1 indicates that your partner was particularly helpful or contributed exceptional effort. A rating of 0 indcates that your partner

met the class expectations of them. Rating your partner at -1 means that they refused contribute to the project, failed to put in a resonable effort or actively blocked you from participating. If a student recieves three ratings of -1 they must attend a mandatory meeting with the instructor to dicuss the situation, and recieving additional -1 ratings beyond that, the student risks losing a letter grade, or even failing the course.

Colophon

Credit for this project belongs to Karen Reid - University of Toronto . The original assignment can be found at http://nifty.stanford.edu/2009/reid-starmap/starmap.html.

This project was developed by Dr. James Dean Palmer of Northern Arizona University.
More projects in this series can be found at `http://problemspace.org`.