Plotting with Python

One reason Python has become so popular of a language recently is due to the extensive libraries that have been written for it. In particular *SciPy* offers powerful scientific operations with a relatively simple interface. Within the SciPy project are other libraries such as *NumPy*, and *Matplotlib*, which we will be playing with today.

A Simple Bar Graph

Download the file titled `bar_chart.py` from the assignment page by left-clicking on it (make sure you are using `firefox` or the file may not download properly). The source code for this file is shown below.

```
#!/usr/bin/env python3.4

# 'import' statements allow us to use functions that are written in
# another file, such as the libraries matplotlib and numpy.
import matplotlib.pyplot as plt
plt.rcdefaults()

# 'import ___ as ___' allows us to give a name to the group of functions
# we are importing.
import numpy as np
import matplotlib.pyplot as plt


# Data taken from Distrowatch at
# http://distrowatch.com/

# Here we define our data, making sure that each bar (Linux distribution)
# is in the same position as it's data.
distros = ('Mint', 'Debian', 'Ubuntu', 'openSUSE')
hits_per_day = (3005, 1851, 1553, 1272)

y_pos = np.arange(len(distros))

# Creation of the bar graph
plt.barh(y_pos, hits_per_day, align='center', alpha=0.4)
plt.yticks(y_pos, distros)

# Label and title
plt.xlabel('Hits Per Day')
plt.title('Popularity of Linux Distributions Based on Hits Per Day on Distrowatch')

plt.show()
```

Although SciPy and Matplotlib were created for ease of use, this program still looks relatively complicated at first glance. Usually when trying to create graphs, or take advantage of some other functionality, one starts with a template. Example programs / graphs can be found at the link titled `Plotting Templates` on the assignment page. If you open the bar graph example on that page you will find many similarities to this program.

After downloading the program execute the following commands in a terminal to run the program.

```
denver:~$ cd ~/python_fun
denver:~/python_fun$ mv ~/Downloads/bar_chart.py ./
denver:~/python_fun$ gedit bar_chart.py &
denver:~/python_fun$ chmod +x bar_chart.py
denver:~/python_fun$ ./bar_chart.py
```

Note that you will need to close the window which was created to use your terminal again.

After closing the window visit the DistroWatch link provided in the source code (and on the assignment page), find two more distributions and their Hits Per Day (right hand column) and add them to the graph by editing the file `bar_chart.py`. Save your `bar_chart.py` file and run the same way as above. Try changing the title of the graph, or the x axis label. Once done you can close the `gedit` window.

Run your program one more time and when the graph appears click the button in the lower right hand corner, the button that looks like a floppy. Name the image `bar_chart.png` and make sure it saves to your `python_fun` directory.

Pie Charts

Download the file titled `pie_chart.py` from the assignment page by left-clicking on it (make sure you are using `firefox` or the file will be named differently). Now move this file to you `python_fun` directory as you did above for the other program, open it in `gedit`, add execution for it, and run it from your terminal (exactly the same as above).

Complete the following for this program.

- Follow the link in the source code (or the assignment page) to `TIOBE` site
- Find the first 6 most popular languages and their percentages and replace the data in the source file with this
- Make sure that the remaining percentage is covered by a seventh "other" category so the percentages add up to 100
- Make sure that the variables `languages, lang_pop_percentage, pie_colors, and pie_explode` all have the same number of items in their list (and looking at the variables already there, keep consistent with the syntax)
- Use the `pie_explode` to "explode" you favorite language in the list
- Read through the comments and complete anything else listed to do (comments starting with `TODO:`

Run your program one more time and when the graph appears click the button in the lower right hand corner, the button that looks like a floppy. Name the image `pie_chart.png` and make sure it saves to your `python_fun` directory.

Function Graphs

Download the file titled `function_graph.py` from the assignment page by left-clicking on it (make sure you are using `firefox` or the file will be named differently). Now move this file to you `python_fun` directory as you did above for the other program, open it in `gedit`, add execution for it, and run it from your terminal (exactly the same as above).

Complete the following for this program.

- Read through the comments to complete what is asked in the comments starting with "TODO:" (Ones marked optional are optional, but recommended. You are encouraged to ask for help if you do not understand, some of this material may be new for you!)
- Complete the items in the order they appear
- For the first "TODO:" item use the following equation for the function:
  $(x - 3.5) * (x - 5) * (x - 7) + 85$
- After each item you complete save the file in `gedit`, run it on the terminal, then close the graph before try to run it again

Run your program one more time and when the graph appears click the button in the lower right hand corner, the button that looks like a floppy. Name the image `function_graph.png` and make sure it saves to your `python_fun` directory.

Creating Your Own Graph

Now you get to create your own graph! Find the link on the assignment page titled `Plotting Templates` and search through the page for an example you would like to implement. It may take you a few tries to find an example that you can use. You are also welcome to extend on one of the example presented here, just make sure to leave the other copy (i.e. make a copy of the one you would like to extend upon) as you will need to turn in all four parts.

Find some set of data online that can be represented with the type of graph you chose. Complete the following with you graph.

- Make sure your python program is named `my_graph.py`
- Correctly title your graph
- Correctly label your axis (if appropriate)
- Use the data you found in the graph
- Make sure that labels, titles, and the like do not overlap
- Add one more feature to your graph

Run your program one more time and when the graph appears click the button in the lower right hand corner, the button that looks like a floppy. Name the image `my_graph.png` and make sure it saves to your `python_fun` directory.

When you are finished refer to the assignment page for instructions on submission.