Introduction to programming, Lesson 7

1. Pie chart of weather forecasts

import matplotlib.pyplot as plt
import numpy as np

plt.figure(1, figsize=(6,6))
labels = 'Rain', 'Sunny', 'Cloudy', 'Partly cloudy'
fracs = np.array([15, 30, 45, 10])
plt.pie(fracs, labels=labels, autopct='%1.1f%%', shadow=True)
plt.title('Weather forecasts')
plt.savefig('forecasts.png')
plt.show()

2. (*) Pyramid of ages: Pie chart of Alsace population

Draw two pie charts, one for female age groups and one for male age groups in Alsace.

3. Bar plot for programming language usage

import matplotlib.pyplot as plt
import numpy as np

languages = np.array(['Python', 'C++', 'Java', 'Perl', 'Scala', 'Lisp'])
number_of_users = np.array([10, 8, 6, 4, 2, 1])

plt.title('Programming language usage')
x_pos = np.arange(len(languages))
bc = plt.bar(x_pos-0.4, number_of_users, color='b')
plt.xticks(x_pos, languages)
plt.ylabel('Usage')
plt.savefig('programming_languages.png')
plt.show()

4. (*) Pyramid of ages: Bar plot of male and female populations

Draw a bar chart of male and female populations by ages. To add a legend to your graph, use plt.legend((bc_men[0], bc_women[0]), ('Men', 'Women')).

5. Plots of “scientific” functions

import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(-5, 5, 101) # 101 equally spaces ticks from -5 to 5
y = np.sin(x)
plt.plot(x, y)
plt.show()

6. (*) Plots of “scientific” functions

Draw a function which is equal to $-x-5$ on $[-10, -5]$, to $x^2-25$ on $[-5, 4]$, and to $\sin(x-4)-9$ on $[4, 10]$.

7. HyperText Markup Language

Go to a webpage http://dataquestio.github.io/web-scraping-pages/simple.html. Use Developer Tools (View → Developer → View source code) in your browser to see the webpage's code. HTML consists of elements called tags (<html>, <head <title>, <body>, etc).

8. Download an HTML document
import requests
page = requests.get("http://dataquestio.github.io/web-scraping-pages/simple.html")
page.content

9. **Finding all instances of a tag at once**

```python
import requests
from bs4 import BeautifulSoup
page = requests.get("http://dataquestio.github.io/web-scraping-pages/simple.html")
soup = BeautifulSoup(page.content, 'html.parser')
for tag in soup.find_all('p'):
    print tag.get_text()
```

10. **Classes and ids**

The class and id properties give HTML elements names, and make them easier to interact with when we're scraping. One element can have multiple classes, and a class can be shared between elements. Each element can only have one id, and an id can only be used once on a page. Classes and ids are optional, and not all elements will have them.

```html
<html>
<head>
</head>
<body>
<p class="bold-paragraph">
    Here's a paragraph of text!
    <a href="https://www.dataquest.io" id="learn-link">Learn Data Science Online</a>
</p>
<p class="bold-paragraph extra-large">
    Here's a second paragraph of text!
    <a href="https://www.python.org" class="extra-large">Python</a>
</p>
</body>
</html>
```

11. **Searching for tags by class and id**

```python
import requests
from bs4 import BeautifulSoup
page = requests.get("http://dataquestio.github.io/web-scraping-pages/ids_and_classes.html")
soup = BeautifulSoup(page.content, 'html.parser')

# We can find all p tags that have class 'outer-text'
print "Contents of all tags p with class 'outer-text'"
for tag in soup.find_all('p', class_='outer-text'):
    print tag.get_text()

# We can find all tags that have class 'outer-text'
print "Contents of all tags with class 'outer-text'"
for tag in soup.find_all(class_='outer-text'):
    print tag.get_text()

# We can find all tags that have id 'first'
print "Contents of all tags with id 'first'"
for tag in soup.find_all(id='first'):
    print tag.get_text()
```

12. **Using CSS selectors**

```python
import requests
from bs4 import BeautifulSoup
```
page = requests.get("http://dataquestio.github.io/web-scraping-pages/ids_and_classes.html")
soup = BeautifulSoup(page.content, 'html.parser')

# We can find all p tags that have class 'outer-text'
print "Contents of all tags p that are inside of a div tag"
for tag in soup.select('div p'):
    print tag.get_text()

You can also use p a — finds all a tags inside of a p tag; body p a — finds all a tags inside of a p tag inside of a body tag; html body — finds all body tags inside of an html tag; p.outer-text — finds all p tags with a class of outer-text; p#first — finds all p tags with an id of first; body p.outer-text — finds any p tags with a class of outer-text inside of a body tag, etc.

13. (*) Scraping the names of ENS courses that use Moodle

Go to moodle.di.ens.fr. Scrape the names of all ENS courses that use Moodle.