Lab 6

Learning Objectives:

I. **What characters do I type next?** *How do I interact with data in Python and then display this data?* This lab will focus on using the powerful NumPy, SciPy and matplotlib libraries to be able to mathematically manipulate data and display it using plots.

II. **How do I plan a project and execute my plan?** *How do I integrate Git into my workflow for version control?* This lab will provide the opportunity to practice using Git within a Unix environment to track changes and communicate about them with an external server. How do I get all the way from raw data to plots? This lab will provide the opportunity to practice mapping out a data flow and writing code from scratch that fulfills it.

III. **Let me Google that for you.** *How do I use function 'X'?* This lab will require use of many different functions from libraries. One of the most useful skills a programmer can have is the ability to read documentation and understand how to use certain codes. Thus for this lab we want you to look up documentation on functions and figure out how to use them to help you get the results you want.

While You Work is an opportunity to get in the habit of Googling functionality efficiently.
Part 1 gives you practice with Python's and NumPy's handy indexing, this time with 2D arrays.
Part 2 helps you practice creating and implementing a data flow for an applied problem.
Part 3 gives you practice with logical slicing in an applied context.

While You Work: Google Documentation

It's hard to remember every little detail of a function, so it's useful to be able to look them up. When you need to learn or relearn how to use it up, try Google first. If you figure out that certain search terms work better than others, take note of that for future reference.

Part 0: Pull the Lab from GitHub

This lab has starter code. In case you need a reminder of how to find it, it's here:

```
git clone https://github.com/physics91si/username-lab6.git
```

Part 1: Indices and 2D matrices

This part is a chance to practice indexing into 2D arrays using a few simple tasks.

1. Start a file `matrix.py`.
2. Import `numpy` as `np` and `matplotlib.pyplot` as `plt`.
3. Make a 9×9 array of zeros called `arr`. (Hint: use `np.zeros`.)
4. Assign all the entries in the left 3 columns and the last row to the value 1.
5. In one line of code, assign three individual entries the value 1: (4,5), (7,7), and (1,8).

6. Plot your data using `plt.spy(arr)` and save it inside the `lab6` repository.

Part 2: Theme Park Data Analysis

For this part you'll play theme park engineer. You're assigned to analyze the performance of a drop tower, which is like a roller coaster that only moves vertically. Riders are strapped into seats in the cab, and then motors within the obelisk-shaped structure accelerate the cab up and down. **Your task is to plot the position, velocity, and acceleration of the drop tower over time.** Write code in a file `tower.py`, and save the plots inside your `lab6` repository.

We won't provide you any directions for this part. Use what you've learned about Googling and data flow to solve the task!

Here's additional information that you may find helpful at some point.

- The data you'll need is stored in `droptower_vdata.txt`.
- The function `numpy.loadtxt` and its optional argument unpack might be helpful.
- The function `scipy.integrate.cumtrapz` might be helpful. For the purposes of this lab, you should call it with the optional `parameter initial=0`.
- The function `numpy.diff` might be helpful. The array it returns is one element shorter than the array it accepts; make sure you understand why.
- You can save a matplotlib figure by clicking the floppy disk icon.

Part 3: Drop Tower Continued, If You Have Time

Your supervisor wants to know how powerful the drop tower's motors are. They're exerting the most force when they're causing the cab to accelerate upwards. **Your task is to calculate the average acceleration of the cab when it's accelerating upwards.**

Use the array of accelerations you calculated in Part 2. We don't care about the negative values of acceleration (because that's when gravity is doing the work), so create a new array containing only the values of acceleration that are positive. *(Hint: use logical indexing.)* Then calculate the average of the positive accelerations and print it.