CTA200H 2017 Problem Set 1

Due: Mon May 15 at the start of class time (1:00).

Please submit your responses electronically by github. Instructions included at the bottom of this assignment. Make sure the instructors will be able to run your code. Also include any figures you make as part of the activity.

For this assignment you may work in pairs and submit your work as a pair. In this case please make sure to discuss as much as possible and to share the coding work equitably.

1. (Dan Green.) Write a script that finds and replaces a given word in all the .txt files in the working directory, with the following properties:

   a The script *name* is called with "*name find replace*" where *find* and *replace* are words the user chooses when running the program.

   b A new directory named *replace* is created within the working directory.

   c A copy of the .txt files with *find* is put in the *replace* directory, but with all occurrences of 'find' replaced by 'replace.' If a file does not contain *find*, it is ignored.

2. (Adapted from Newman.) The binomial coefficient is given by

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!} \tag{1}$$

   for $k \geq 1$ and 1 for $k = 0$.

   a Write your own routine to generate the binomial coefficients for a given n and k. Make sure the result is an integer (not a float) and gives the (correct) value 1 for $k = 0$. (Hint; you can do a cancellation to make it more tractable.)

   b Use your routine to write out the first 20 lines of Pascal's triangle. This gives the coefficients for expanding binomials of the form $(x + y)^n$.

   c Consider a biased (e.g., bent) coin with probability p of coming up heads, and probability $1 - p$ of coming up tails. The probability of obtaining heads k times when flipping the coin n times is given by

$$\binom{n}{k} p^k (1-p)^{n-k}. \tag{2}$$

   Consider an experiment in which you flip the coin n times and hope to obtain at least k values of heads. For values of p, n, and k of your choice, use your routine to calculate the probability of obtaining heads at least k times in n flips. (Example application: a baseball player with batting average $p = 0.250$ has $n = 4$ attempts to obtain a hit, or at-bats, during a given game. What is the chance that s/he will obtain at least one hit during the game, assuming that all at-bats are statistically the same and uncorrelated?)

d Simulate the experiment N times for $N \in \{10, 100, 1000\}$, or a similar range of your choice. For each value of N, which fraction of experiments were successful? Consider showing the results in a plot.

3. (Adapted from Newman.) The Bessel function of the first kind of order m is given by

$$J_m(x) = \frac{1}{\pi} \int_0^\pi d\theta \cos(m\theta - x \sin\theta). \tag{3}$$

a Write a routine to calculate this integral directly, using the method of integration of your choice, using only the library sine and cosine functions (you may use the library Bessel functions to check the results of your integral). Plot the results for a few values of m.

b Suppose that light passing through a circular telescope aperture of radius a is focussed on the telescope's focal plane. If the source is effectively a point on the sky, then the pattern of intensity seen in the focal plane at a distance q from the optical axis is given by

$$I(x) = I_0 \left(\frac{2J_1(x)}{x} \right)^2 \tag{4}$$

where I_0 is the intensity at the center, and $x = \frac{2\pi a q}{\lambda R}$. Here λ is the wavelength of the light and R is the distance from the aperture to the focal plane. This is known as the "point spread function" in optical astronomy, the "beam" in radio astronomy, and the "Airy disk" in general. (The geometrical ratio $R/(2a)$ is known as the "f-number" of the telescope and typically has values of several.) For a given set of parameters, make an image of the two-dimensional point spread function, using a monochrome color scheme. Indicate the x and y axes with real distance units. (It might be hard to see the detail away from the main beam; if so, try increasing the contrast of your image in order to see the features on the edge.)

c Download a high-resolution astronomical (or other) image of your choice. If this image were viewed through a telescope with a finite aperture, each pixel's value would get smeared out to neighboring pixels, degrading the image. This is known as *convolution*. To simulate how this image would look when viewed through a telescope, use a canned convolution routine (e.g., scipy.ndimage.filters.convolve(), scipy.signal.convolve2d()) to convolve the image by your point-spread function. Don't worry about being quantitative in this sub-section; rather play around to see what happens to your image when you convolve it.

4. For each point in the complex plane $c = x + iy$, with $-2 < x < 2$ and $-2 < y < 2$, set $z_0 = 0$ and iterate the equation $z_{i+1} = z_i^2 + c$. Note what happens to the z_i's: some points will remain bounded in absolute value $|z|^2 = \Re(z)^2 + \Im(z)^2$, while others will run off to infinity. Make an

image in which your points c that diverge are given one color and those that stay bounded are given another. (Once you have done this, you can try coloring the points that diverge using a colorscale that indicates the iteration number at which the given point diverged.) Try zooming in on a portion of the image and trying again.

5. (Adapted from Newman.) Analytically compute the derivative of $f(x) = x(x-1)$. Now let's take the derivative numerically (imagine for a moment that this was a function whose expression could not be written in analytical closed form). With the definition of the derivative

$$f'(x) = \lim_{\delta \to 0} \frac{f(x+\delta) - f(x)}{\delta} \tag{5}$$

we cannot actually set $\delta = 0$ but we can approximate it by taking δ to be small. At the point $x = 1$, compute the approximate derivative using $\frac{f(x+\delta)-f(x)}{\delta}$ for a log-spaced range of values of δ between 10^{-4} and 10^{-14}. What do you find? Consider making a plot.

6. Write your own program simulating a space battle. It should have 3 types of spaceships: standard spaceships, warships, and speeders. Each ship has lasers, shields and hull strength (and most importantly, a name!). Warships additionally have high powered missiles which they fire 30% of the time, while speeders have a 50% chance to dodge incoming shots. When a spaceship is shot by another ship, it first depletes its shields equal to the strength of the shot. When it runs out of shields, it takes hull damage at 50% of the shot strength. Once the hull is breached, the ship is destroyed!

   Your program should include a class Ship that defines the standard spaceship, and two classes which inherit from Ship called Warship and Speeder. The classes should store the ship's shield strength, hull strength, laser power and name. There should be, at a minimum, methods to deal with when a ship shoots, is shot at, whether the ship is destroyed or not, and printing a diagnostic summary of the ship's status.

   Your program should instantiate 3 regular ships, 1 warship and 1 speeder. The spaceships should shoot randomly at each other until only one remains (targets cannot be themselves nor ships that are already destroyed). Print a log of the battle as it progresses, and declare a final victor.

7. **Submitting Your Assignment** - Assignments must be submitted by github.

   (a) If you do not already have a github account, go to github.com and create one

   (b) Create your own public git directory named `lastname_firstname_Assignment1`, where you use your firstname and lastname

(c) Make 6 subdirectories, called question1, question2, ..., question6

(d) In each subdirectory include any python script or ipython notebook required. An example of this directory setup can be seen at

`https://github.com/ttricco/cta200h/tree/master/problemset1/Stein_George_Assignment1`

(e) When you have the clean subdirectory system set up and all your questions completed, email cta200@cita.utoronto.ca and include the exact command for me to pull your directory system.

eg. `git clone git@github.com:USERNAME/lastname_firstname_Assignment1`.

(f) **Check to make sure that it works (pull it yourself to a new directory and make sure everything is there) before submitting**