Chem 160 Homework 3

Download the github.com repository at the URL:
`https://github.com/mcolvinphd/chem160homework3.git`
After completing the homework, create a GitHub repository called **`chem160homework3`** and upload all your homework files to this repository. Note that this homework will be mostly graded by a computer script, so it is essential that you use precisely the file names indicated for your programs.

For full credit, submit this homework to a repository called `chem160homework3` on your GitHub account by 9/22/19.

1. In addition to the energy and heat capacity, another quantity often calculated in an Ising model simulation is the magnetization of the system, *m*, which is basically the average total spin of the system. For an $n \times n$ Ising model:

$$m = \frac{1}{n^2}\sum_{i}^{n}\sum_{j}^{n}\sigma_{ij}$$

Where σ_{ij} is the spin at site *(i, j)* and $\sigma_{ij} = +1$ for an up spin and $\sigma_{ij} = -1$ for a down spin. Therefore, if all of the spins are pointed up the magnetization is +1.0, if all are pointed down the magnetization is -1.0, and if half are pointed up and half pointed down, the magnetization is zero.

You will modify the Ising model program we wrote in module 4 (and is provided in the GitHub repository as **`ising.py`**) to make a new program called **`isingmag.py`** that will calculate and print out the magnetization of the system. To do this you need to add the calculation of the magnetization just after the calculation of the heat capacity (Cv) near the end of the program. This will require a double loop over i and j to sum up the spin values as in the equation above. Add the magnetization value to the existing print statement so it prints out after the heat capacity.

2. **Extra credit:** In the planetary dynamics program we covered in module 5 the "star" is kept fixed at the origin (0, 0) even though its mass is only 100 times greater than its orbiting planet. (In contrast the sun is more than 300,000 times more massive than the earth). To make this simulation more accurate, we can modify **`orbit.py`** (provided in the GitHub repository along with the module **`drawtraj.py`**) to allow the star to move as well due to the gravitational pull of the planet. Name the new program **`orbit2.py`**. Since this program calls the Python Image Library module, I recommend doing this part of the assignment using the Mu editor which already has that module installed.

Because of Newton's third law you don't have to calculate the force twice, since the force on the star is equal and opposite to the force on the planet, but you do need to calculate the change in position and velocity of the star in response to the force due to the planet. You will need to add new variables for the position and velocities of the star as well as new lists to store the trajectory of the star. Finally, you will need to modify the drawtraj.py file to accept as arguments the trajectory lists for the star and also modify the for loop in the **`drawtraj()`** function to plot the motion of the star. Assuming that the star's trajectory is stored in the variables starx and stary, the following lines in the plotting function need to be modified as follows (deleting what is in red and adding what is in blue):

```
draw.ellipse((-rsun+rmax/2,-rsun+rmax/2,rsun+rmax/2,rsun+rmax/2),(255,0,0))
for i in range(len(trajx)):
    draw.point((scale*trajx[i]+rmax/2,rmax/2-scale*trajy[i]),(0,255,0))
    draw.point((scale*starx[i]+rmax/2,rmax/2-scale*stary[i]),(255,0,0))
```