PHY4410 HW 6 Fall 2019
Due ~~Wednesday~~ Friday (extension) Nov 13 by 4:30pm -- start **early!**

1. Write and upload a python program, "gas_1d.py", to simulate N *collisionless* "balls" moving in a 1-dimensional "box" with boundaries at x=0 at x=1. Give the balls these attributes:
   - Position: randomly uniformly distributed (via numpy.random.uniform())
   - Velocity: random with a normal (Gaussian) distribution, centered about 0 with a standard deviation of 0.1, e.g.,
     ```
     vx = np.random.normal(size=N,loc=0.0,scale=0.1)
     ```
   - Use whatever time step size you like (start small). You may use basic kinematics or a Hamiltonian approach, whichever you like.
   - No gravity.
   - ***Use a time step of 0.1 and run for 1000 total iterations.***

   When each ball hits (or is about to hit) the wall, it needs to "bounce" elastically, i.e. reversing its velocity.

   a) Produce a measure of the *average force F* on the walls of the box, by calculating the time-average of the change in momentum ("impulse") of balls hitting the walls. (Just use a unit mass m=1, so momentum "=" velocity.) What is the value of *F* for *N*=100, 1000, and 10000 (as the time-of-averaging becomes sufficiently large)? Use an output syntax of
   ```
   N = ___, Right boundary at ___: Force = ___
   ```

   b) For 1000 balls, demonstrate how the force changes when you *double* the size of the box, i.e. move the right-hand boundary in x from 1 to 2: Produce an combined output log (e.g. a text file) showing the run of a) and then b). Simply run the code twice with two different values for the right boundary, namely 1 and 2, and then copy and paste the output into a file, called `hw6_p1b.txt.` Use a syntax of
   ```
   N = ___, Right boundary at ___: Force = ___
   ```

2. Extend this work to 2-dimensions ("gas_2d.py") for a box with sides at x,y = 0 & 1, where now…
   - The direction of the velocity can now be any angle in the plane.
   - When balls bounce off the walls, it is the *perpendicular component* of their momentum that changes
   - Calculate the "pressure" by dividing the average force by the *perimeter* of the box

   a) Demonstrate how the pressure changes when you double the size of the box, i.e. move the boundary in x from 1 to 2 (leaving the boundaries in y the same as before) by a series of ***steps of size 0.1***.

   b) For 1000 balls, Show that the product of pressure and the *area* of the box stays (roughly) constant: Upload an output log called `hw6_p2b.txt.` Use a syntax
   ```
   print(f"N = {N}, RB = {bounds[1]}: P = {pressure}, P*A = {pa}")
   ```

3. Remove the "collisionless" part! Make the balls bounce off each other. Try a ball radius of 0.02. Use N=200, boundaries of [0,1] in both directions, a timestep of 0.1, and 1000 iterations.
   a) Produce a measure of the *average* kinetic energy ("temperature") at beginning and the end:
      `Initial: N = ______: <KE> = _____`
      `Final:   N = ______: <KE> = _____`
   b) ~~Double volume but place two different populations of balls in the two "sides" of the box: One as before, but another with *half* the (average) speed – i.e. change "loc" to 0.25. (An easy way to do combine two arrays is to use np.concatenate()). Allow the two populations to intermingle (for 1000 iterations) and observe the change in average kinetic energy of the populations, call them 1 and 2. Print out the initial and final average kinetic energies:~~
      ~~`Initial: N = ______: <KE_1> = _____, <KE_2> = _____`~~
      ~~`Final:   N = ______: <KE_1> = _____, <KE_2> = _____`~~
      Too hard to keep track of 1 and 2 and do collisions *between* them.
   c) Change the original velocity distributions from `normal` to `uniform`. Plot histograms of ball speeds initially and finally, e.g.

```
plt.hist( initial_speeds, bins='auto', density=True)
plt.show()
plt.hist( final_speeds, bins='auto', density=True)
plt.show()
```