PHYSICS 113 – Recitation Assignment 5

Name

Recitation Assignment # 5

Oct. 25, 2006

You may complete this in class. However, if you are unable to do so, it is expected that you complete this for recitation next time.

Where a box appears, call over Travis to check over your progress. When the sheet is complete, you will hand it in. Also, you are expected to email your final programs to Travis.

Today: Ideal Gas

On the course website, we've posted a program called `prog5_starter.py`. This program contains a fair amount of physics. It creates a number of "atoms", and subjects them to two forces: 1) Ordinary gravity near the surface of the earth, and 2) collisional forces when the atoms hit each other, or hit the walls.

Do not worry, for the moment, about collisional forces. We'll discuss them later in the term.

Always run your code between steps!

1. Download the program, and save it as prog5.py in your "Contemporary" directory.

2. Run it. Does it do anything? Why not?

   Well, for one thing it's set up so that all of the atoms are simply sitting at the bottom of the well. Instead, take atom 1 and give it a position of (0.5,0.5,0.5). What happens now?

□ 3. In many simulations, it is far better to start the particles with "random" conditions. Thus, your simulation will change each time you run it.

   In order to use random commands, your code must have the statement "from random import *" at the top.

   To pick a random number between zero and one, simply type (for example) x=random(), and x will be a random number. Using this, when you initialize the positions of your atoms, set the x,y,and z components to be random numbers.

4. At this point, you should have your "gas" bouncing around. For each atom, calculate the potential energy:

$$U = mgy$$

   and the kinetic energy:

$$K = 1/2mv^2$$

   at every timestep.

5. At every timestep, calculate the *total* kinetic and potential energy (for all particles).

6. We'd like to display this. In order to make displays, you will need yet another package. At the top of your code, you will need to write,"from visual.graph import *".

   Also, in the code right after you've set up your atoms, type:

   ```
   g2=gdisplay(xmin=0,xmax=1000.,ymin=0,ymax=10*natom*1.)
   ```

   This will create a new window to be used for plotting date. The x-range is 1000 (the length of the simulation), and the y-range is set by the number of atoms.

7. In order to plot to your plotting window, you need to create a "gcurve". For example, to create a curve called, say, ekin (for kinetic energy, for example), type:

   ```
   ekin=gcurve(display=g2,color=color.red)
   ```

   This format should look very similar to the "curve" objects we've used before.

   Now, to plot a new point on the curve (which you'll want to do once per timestep), simply type:

   ```
   ekin.plot(pos=(t,K))
   ```

   Assuming, of course, that you've called your total kinetic energy K.

□ 8. Do the same thing for potential energy, (color it blue), and total (K+U – color it white). All three gcurves should be plotted to the g2 display.

9. Increase the number of particles to 60. Run the code a number of times. What happens after a while in each randomization of the code?

10. Finally, we want to create a "histogram" of the heights of the gas atoms. A histogram is a display which shows (in this case) how many atoms are at each y-position.

    I'll walk you through this one.

    Are there more near the bottom than near the top? There seem to be. Create a new gdisplay called "g1" and give it an xmin=0, xmax=1.5, ymin=0,and ymax=natom. This should be created right around the same time you make your "g2" window.

    The command:

    ```
    f1=ghistogram(bins=arange(0,2,0.2),gdisplay=g1)
    ```

    should be written right after you make g1. This creates an object called a ghistogram which can be used to plot histograms.

    Additionally, add the line

    ```
    ypos[i]=atom[i].pos.y
    ```

    immediately after you compute the y-position of each atom in each timestep.

    Finally, add the line:

    ```
    f1.plot(data=ypos,accumulate=0)
    ```

once per timestep (after you've calculated the new positions and velocities).

What do you see? Well, you should see that there is a relatively constant distribution of atoms at various heights. This is known as the "exponential atomsphere" and is one of the first things you'll learn in thermodynamics.