Due Wednesday of Week 3 at the start of lecture.

Name (print) ____ Section:__________

Instructor Signature ____

Once you have your program running, ask one of the instructors to check it and then sign above. Your assignment will not be accepted without the signature of an instructor. Your score will be based equally on the signature and your answers to questions in the boxes.

Problems 2.P58 and 2.P59: Planetary Orbits

In this problem you will study the motion of a planet around a star. To start with a somewhat familiar situation, you will begin by modeling the motion of our Earth around the Sun. Write your answers to the questions on this sheet.

Planning

(a) What initial momentum should you use for the Earth? Note that the Earth orbits the Sun once per year and that there are approximately $\pi \times 10^7$ seconds in a year.

(b) What is a reasonable value for the time step to use? If it is too large, you will have accuracy problems, while if it is too small, the program may not run fast enough.

Circular Orbits

(c) Run your program and check to see that the Earth returns to its starting point, and closes the orbit. Do this by displaying the trail of the Earth as it moves.

Computational Accuracy:

(d) As a check on your computation, is the orbit a circle as expected? Run the calculation until the accumulated time, t, is the length of a year: does this take the Earth around the orbit once, as expected?

What is the largest step size, dt, you can use that still gives a circular orbit and the correct length of a year? What happens if you use a much larger step size?

Non-circular Trajectories:

(e) Set the initial speed to 1.2 times the Earth's actual speed. Make sure the step size is small enough that the orbit is nearly unaffected if you make the step size smaller.

What kind of orbit do you get? What step size do you need to use to get results that you can trust?

What happens if you use a much larger step size?

Produce at least one other qualitatively different non-circular trajectory for the Earth. What difficulties would humans have surviving on the Earth if it had such a non-circular orbit?

Force and Momentum:

(f) Choose initial conditions that give a non-circular orbit. Continuously display a vector showing the momentum of the Earth (with its tail at the Earth's position), and a different colored vector showing the force on the Earth by the Sun, (with its tail also at the Earth's position). You must scale these vectors appropriately to fit into the scene. A method to figure out what scale factor to use is to print out numerical values of the momentum and the force and compare the magnitudes with the scale (distance) of the plots. For example, if the width of the scene is W meters, and the force vector has a magnitude of F (in Newton's), you might scale the force by a factor of (0.1xW/F), which would make the length of the vector about 10% of the size of the picture. Is the force in the same direction as the momentum? How does the momentum depend on the distance from the Sun? (**SHOW THIS PROGRAM TO AN INSTRUCTOR FOR THE SIGNATURE.**)

Additional Problems:

Change the form of the gravitational force from Gm_1m_2/r^2 to $Gm_1m_2/r^{1.95}$. Start with the same initial conditions that you had for the earth in a circular orbit around the Sun.

What sort of orbit do you observe?

Change the form of the gravitational force from Gm_1m_2/r^2 to Gm_1m_2/r^3. You will also need to increase G by multiplying by the radius of the orbit. Start with the same initial conditions that you had for the earth in a circular orbit around the Sun.

What sort of orbit do you observe?
Slow the earth down by 15% and describe the orbit you observe. Do the same with it

speeded up by 15%.

Useful VPython Constructs::

The following three VPython constructs will be quite useful in this exercise.

```
#
earth.pos = vector(A,B,0)
#
# rhat is a unit vector in the direction of the vector.
#
rhat = norm(earth.pos)
#
# rsqrd is the square of the magntude of the vector.
# rleng is the magnitude of the vector.
#
rsqrd = mag2(earth.pos)
rleng = mag(earth.pos)
#
```

Program Shell:

The following is a shell to help those with limited programing experience get started with the code. You are not required to use this shell, it is only provided as a template to help people get going on the code. Much of the preparation work above will allow one to initialize things to the correct values (the zeroes in the code). You will also need to code up the physics inside the main loop to cause the earth to move.

```
from math import *
from vpython import *
#
# Needed Constants
#
sun_earth_distance = 0
speed_of_earth = 0
#
# Set up the displays
#
scene2 = canvas(title='Earth orbiting the Sun',caption='Animated Display',
     center=vector(0,0,0), background=color.black)
#
# Make the radius of each object large enough to see them
#
earth = sphere(pos=vector(sun_earth_distance,0,0),radius=1e9,color=color.green)
sun   = sphere(pos=vector(0,0,0),radius=5e9,color=color.yellow)
#
G = 0
#
earth.mass = 0
sun.mass   = 0
#
#
earth.momentum = earth.mass*vector(0,speed_of_earth,0)
sun.momentum = sun.mass*vector(0,0,0)
#
# Create a trail for the earth, and vectors for the force on the earth.
# scale should be a number that lets is see the force arrow on the plot.
#
earth.trail = curve(color=color.magenta)
earth.trail.append(pos=earth.pos)
#
earth.point = arrow(pos=earth.pos,color=earth.color,axis=-norm(earth.pos))
scale = 3e-12
#
# Initial time is 0, and the time step is twelve hours
#
time = 0
dt = 12*3600
#
# We will initially run for one year
```

```
#
while time < 3.15e8:
    rate(100)
#
    time += dt
#
#  Compute the force on the earth using our force function.
#
    earth.force = 0
#
#  Update the momentum and position of the Earth
#
    earth.momentum = 0
    earth.pos = 0
#
#   Update the Earth’s Momentum arrow
#
    earth.trail.append(pos=earth.pos)
    earth.point.pos = earth.pos
    earth.point.axis = earth.force*scale
#
days = time/(3600*24)
print(’days’,days)
```