Cannon Fodder

March 18, 2004

In this problem you will create a class Cannon that fires cannon balls. You can set the angle it is fired with respect to horizontal and the initial velocity of the cannon ball. The Cannon class will simulate the trajectory of the cannonball, trace its path graphically, and compute the distance to its landing point on level ground. Then you will create an instance of the Cannon class and fire it for a number of different angles. This problem should remind you of a previous lab exercise....

Email your completed program to `tolnasb@evergreen.edu` by Monday March 29^{th}. It should meet as many of the following specifications as you can get to work. Anything not specified in the question you are free do design as you please.

1. You program should define a class called "Cannon". Its initialization method should take one parameter, `dt`, the length of time steps used to simulate the motion of the cannon ball using Euler's method. If this argument is not specified when a Cannon object is created, it should default to a value of .01.

2. The"Cannon" class should have (at least) 3 methods called `setAngle()`, `setInitialSpeed`, and `FIRE`.

3. The `setAngle()` method should control the angle of the cannon measured in radians up from horizontal.

4. The `setInitialSpeed()` method controls how fast the cannon ball is moving in meters per second at time $t = 0$ when fired from the cannon. It should take a single argument specifying the initial speed.

5. The `FIRE()` method will fire the cannon ball from the origin computing its trajectory at successive time intervals until it hits the ground at some distance away. Assume uniform acceleration of 9.8 meters per second downwards and no air resistance.

6. If an initial speed was not specified by calling the `setInitialSpeed()` method the cannon ball should be fired with an initial speed of 100 m/s.

7. If an angle was not specified by calling `setAngle()` an angle of $\pi/2$ radians should be used.

8. The `FIRE()` method should return a tuple whose first element is the distance to the point where the cannon ball lands. The second element should be the time in seconds when the ball lands.

9. The `FIRE()` method should use a Visual Python `curve` (not `gcurve`) to trace the curve the cannonball traces in flight. Each successive call to `FIRE()` should create a new curve.

10. Create an instance of a `Cannon` and set its initial speed to 300 m/s.

11. Use a loop to fire it at 19 different angles spaced evenly between but not including 0 and π radians.

12. Make a graph plotting the final cannon ball distance as a function of *angle* (not time) for each of the angles you fired the cannon.

13. print out the final times and distances in two columns for each cannon ball.