## repeat the assignment you are implementing;

Finish the ResolveCollisionWith method to handle collisions with balls of a different mass in a generic manner.

## explain your approach;

Implement a \_Mass property and a \_InverseMass property along with their respective class scope variables. The \_InverseMass property is a read only property as the inverse mass is set while the mass gets set. This is to ensure that the mass and inverse mass are always respective too one another.

Then detect if there is a collision, then simply executed the vector math required for this. But now also involving the inverseMass value.

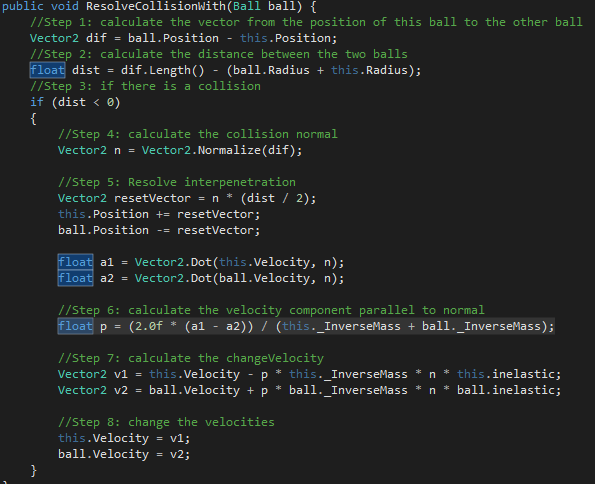
## describe your code;

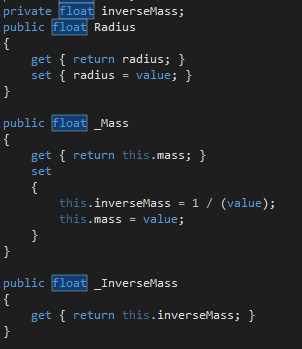
If the difference in length is smaller then the two radii added tighter there is an overlap, then get the collision normal. Fix interpenetration. Dot both ball’s velocity’s and place the value’s in their own respective floats. Then calculate

float p = (2.0f \* (a1 - a2)) / (this.\_InverseMass + ball.\_InverseMass);

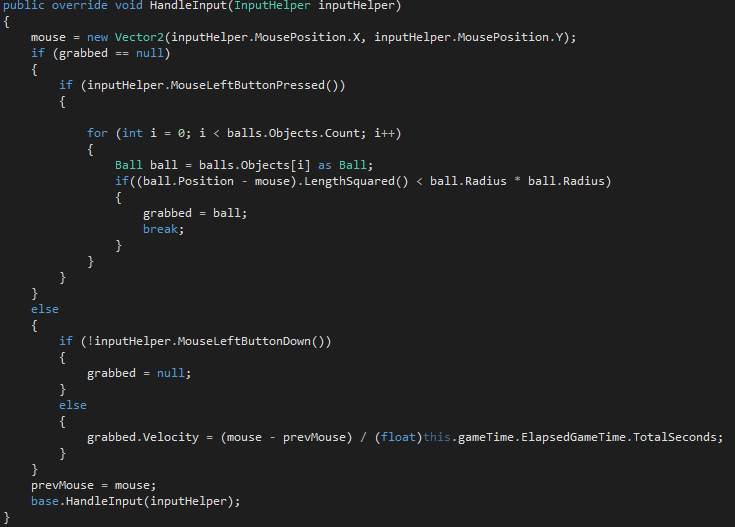
From here add/minus the balls respective InverseMass and the collision Normal. Added in the inelastic value for good measure.  
For fun I also added in that when you left click on a ball you grabbed said ball and can throw it around.

## show (relevant) code snippets;

  
The entire method.



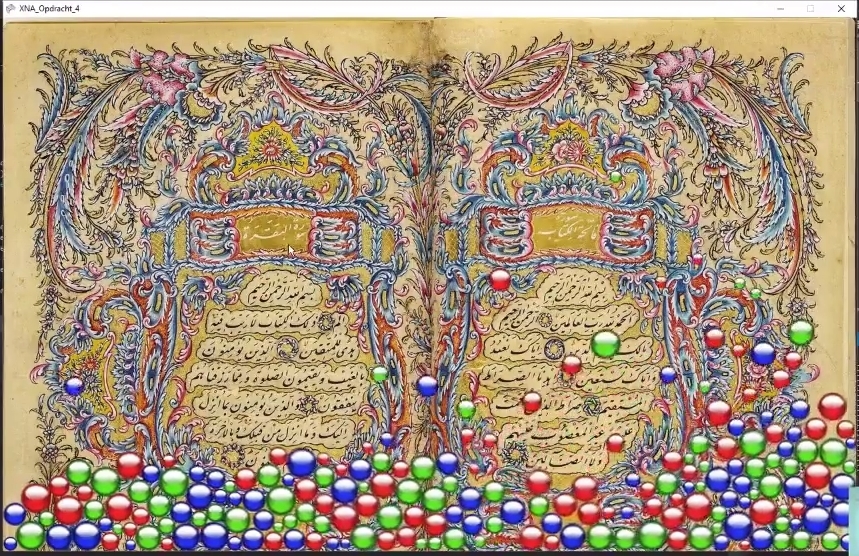
The properties I setup to support this



Grabbing code

## include a screenshot of your program



Just before throwing the ball into the pile.

Directly after the ball crashed into the pile.

Again I have made video’s for the class as well These are found here: <https://www.youtube.com/watch?v=oYf3RrOWsyI&list=PLARkMALdMekM6EMkY0gcQSKvADVAx9zK5&index=7>