Challenge Task 4

Reza Shisheie

2708062

The goal of this challenge task are:

1. Keep the hit count,
2. count should reset if ball falls to the bottom,
3. displaying the hit count at the ball,
4. consider the angle of the hit to the ball and adjust the x velocity accordingly,
5. allowing other joints to hit the ball too.

The achieve the first task, I set a flag which avoids faulty counting. Once the joint enters the certain region in vicinity of the ball, this flag is set to 1 and no counting is done until joint exists this region. This is like debouncing in mechanical switches.

This is the time is gets released:

Point joint1 = this.ScalePosition(skeleton.Joints[JointType.WristRight].Position);

if (thing.Center.Y - joint1.Y< -50)

{

flag\_release = 0;

}

And once it is outside and joint hits the ball, only the first count is counted:

if (thing.Hit(handPtLeft))

{

this.thing.YVelocity = (-2.0) \* this.thing.YVelocity \*

Math.Abs(thing.Center.Y - handPtLeft.Y) / (Math.Abs(thing.Center.Y - handPtLeft.Y) + Math.Abs(thing.Center.X - handPtLeft.X));

this.thing.XVelocity = (-2.0) \* this.thing.YVelocity \*

Math.Abs(thing.Center.X - handPtLeft.X) / (Math.Abs(thing.Center.Y - handPtLeft.Y) + Math.Abs(thing.Center.X - handPtLeft.X));

if (flag\_release == 0)

{

count = count + 1;

flag\_release = 1;

}

this.textBlock\_count.Text = count.ToString();

}

Count gets reset once hit the ground:

// if goes out of bound, reset position, as well as velocity

if (thing.Center.Y >= canvas1.Height)

{

thing.Center.Y = 0;

thing.XVelocity = 0;

thing.YVelocity = 0;

count = 0;

this.textBlock\_count.Text = count.ToString();

}

It display the hit count:

this.textBlock\_count.Text = count.ToString();

The angle of hitting is calculated, and velocities are adjusted by trigonometry:

this.thing.YVelocity = (-2.0) \* this.thing.YVelocity \*

Math.Abs(thing.Center.Y - handPt.Y) / (Math.Abs(thing.Center.Y - handPt.Y) + Math.Abs(thing.Center.X - handPt.X));

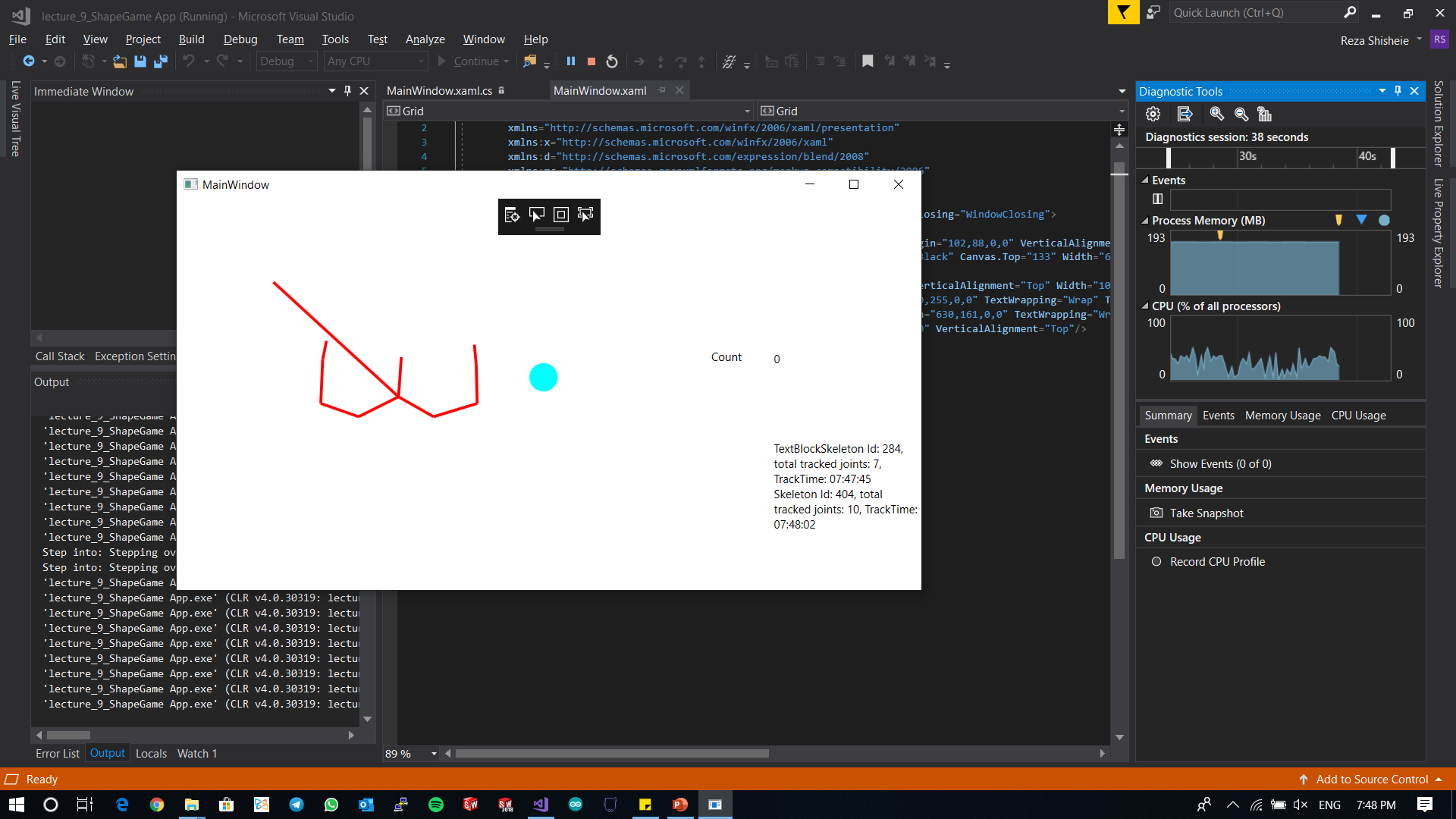
this.thing.XVelocity = (-2.0) \* this.thing.YVelocity \*

Math.Abs(thing.Center.X - handPt.X) / (Math.Abs(thing.Center.Y - handPt.Y) + Math.Abs(thing.Center.X - handPt.X));

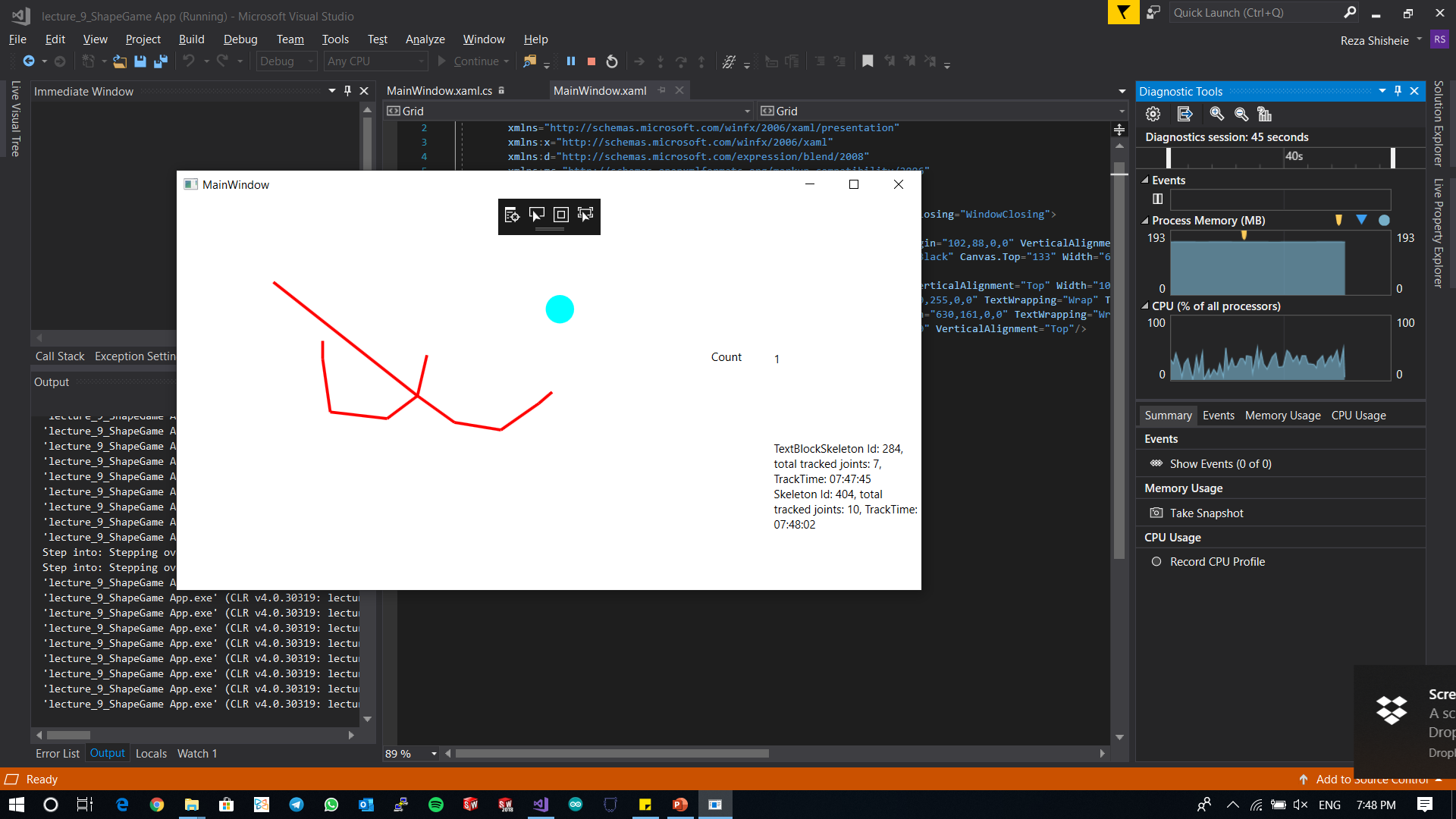
And left and right joint are included to be able to hit the ball:

Attached are some screenshots of different moments:

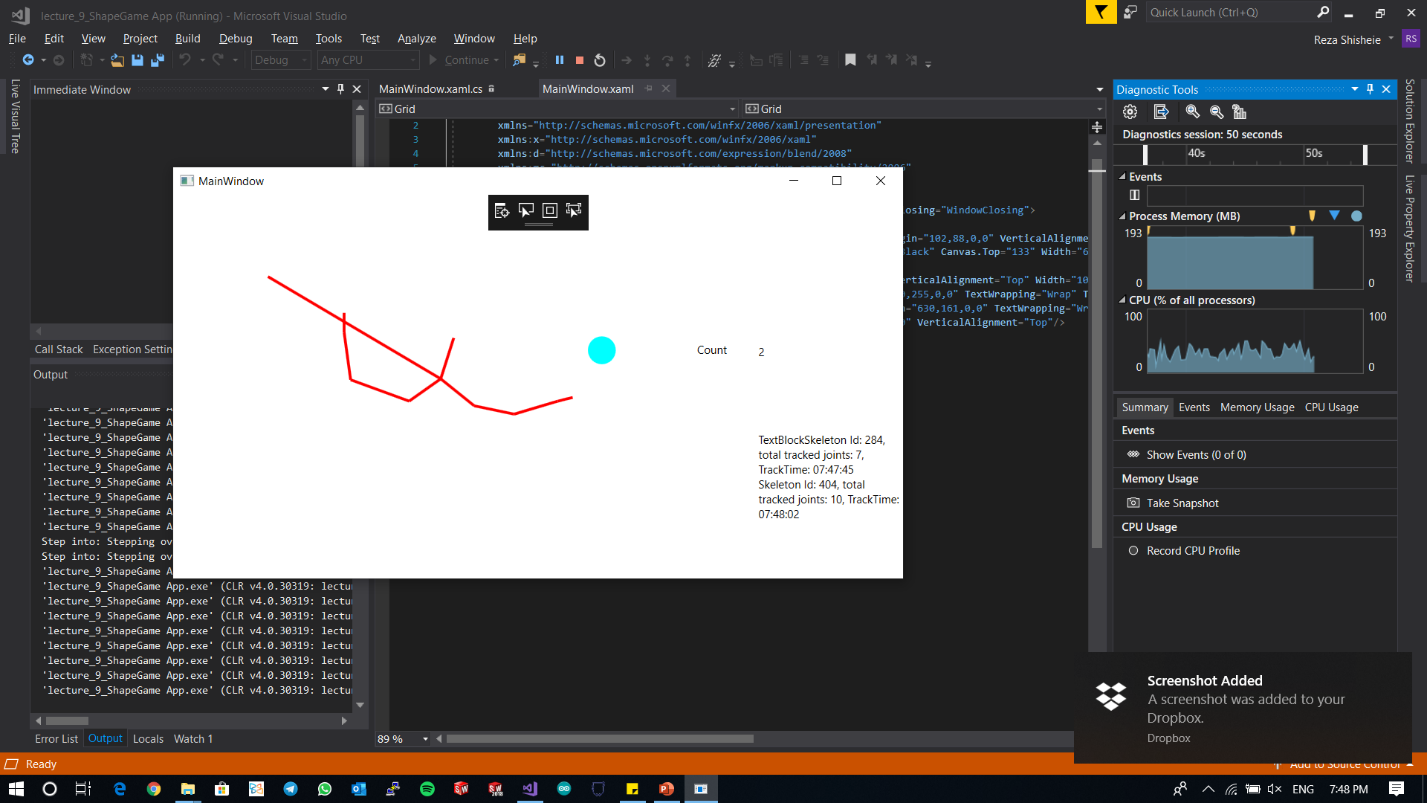
Initial point, count =0;



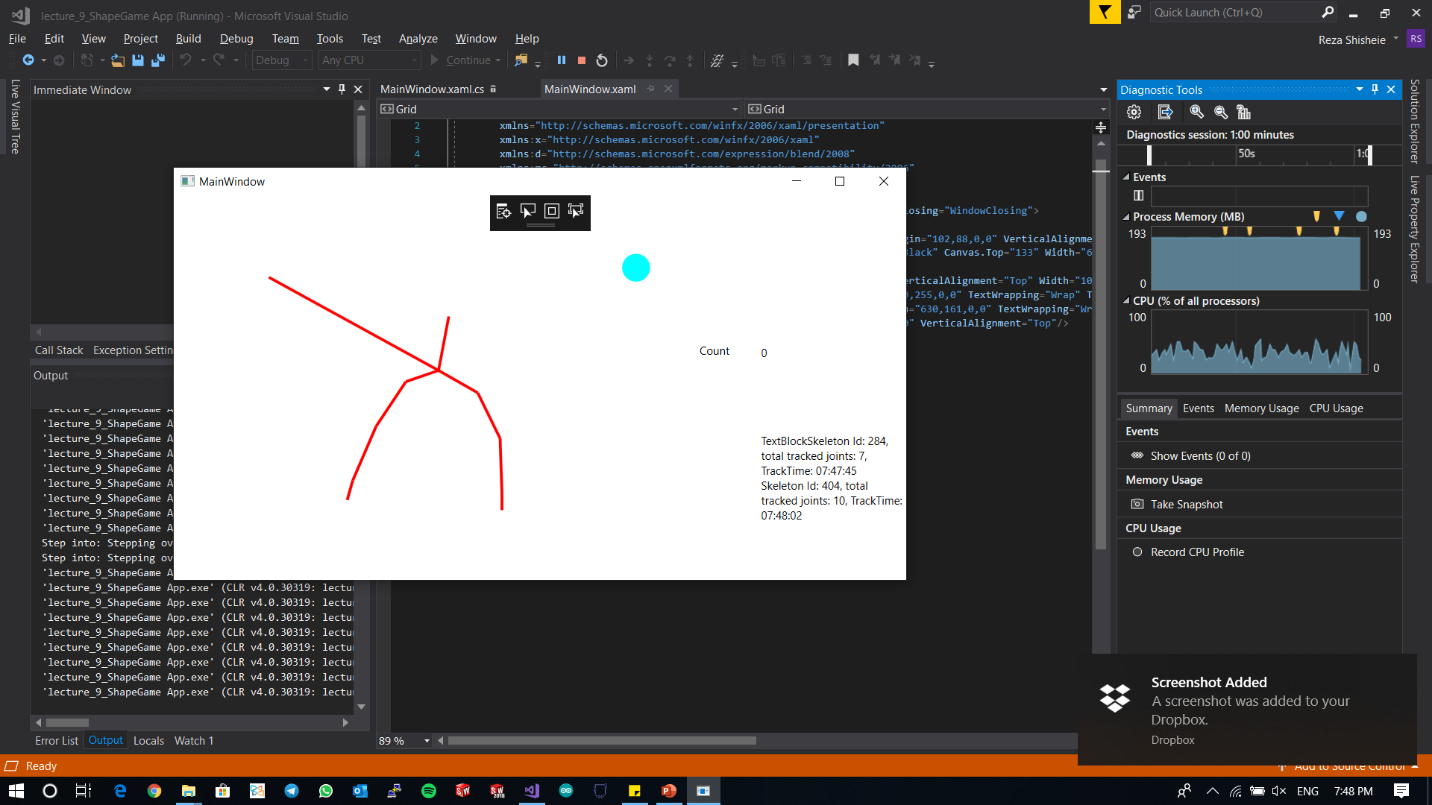
After one time hitting just vertically from bottom:

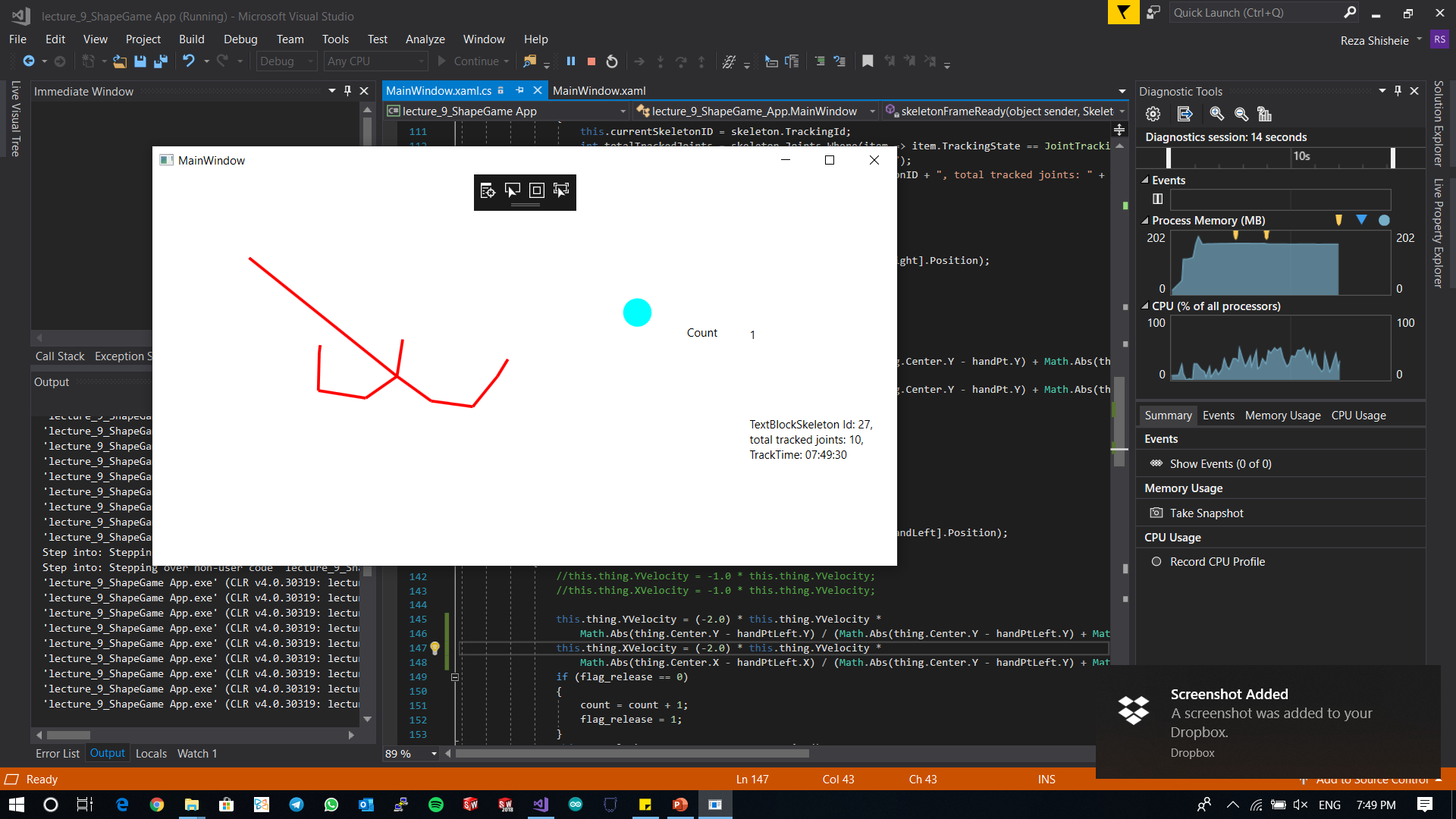
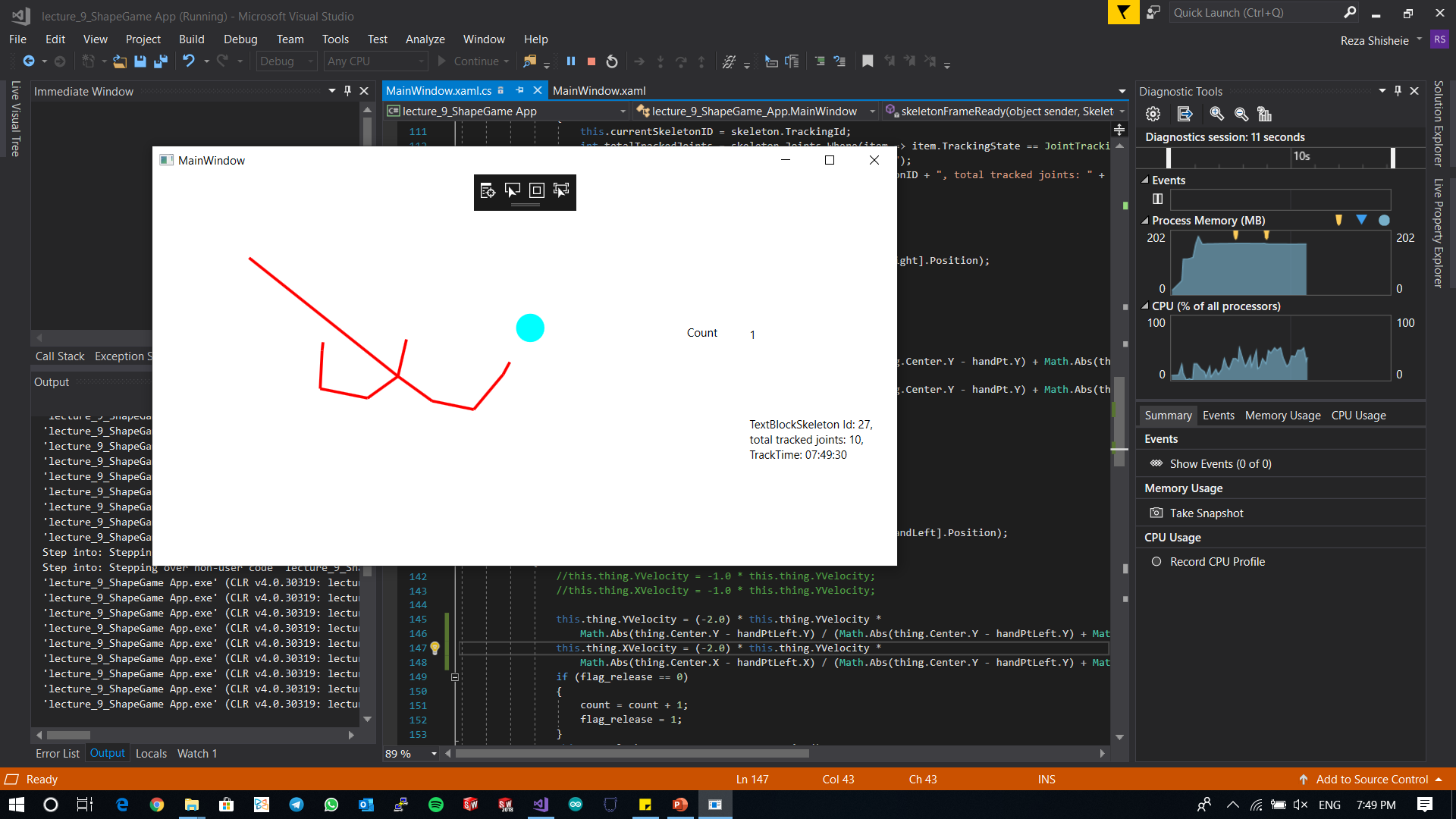


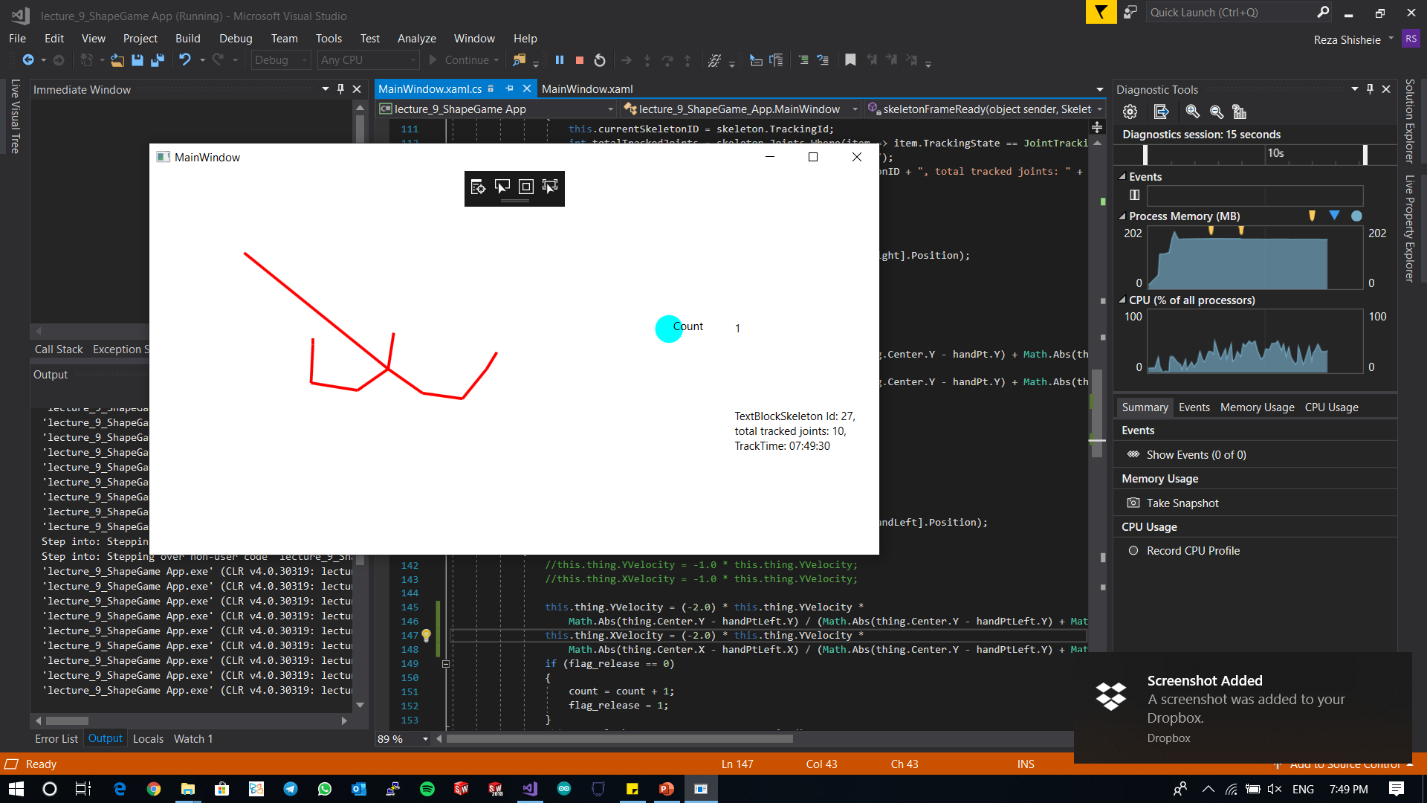
Hitting for the second time with angle:



Letting the ball drop and set count to 0 again:



Rerunning the app t take screen shot of motion of ball if hit with angle:



Showing robustness of counting: Joint is inside ball but no extra counting:

