CMPE2300 – ICA08 – Collide-o-matic

In this ICA you implement a selection of List<> member and extension methods to. The project shall use 2 instances of CDrawer, sized to 600x300. Use the position property to stack the CDrawers approximately beside the main form. You may note some quirkiness in dealing with form dimension values and title bar size. One acts as our main (input) form, the second is used solely for output. Functionally, utilizing a bouncing ball class, when you left-click Blue balls are added, right-click adds green balls. When blue and green balls collide they are removed and they are uniquely added to a separate collection and displayed in the second CDrawer window.

Create a bouncing ball class similar to previous ICAs, but with this reduced set of fields/methods (note no CDrawer object ! ?) :

- Static Random object, use an initializer
- Field of type PointF holding ball center location
- Field of System.Windows.Vector for velocity (X and Y components) - more appropriate
- Field of int ball radius
- public automatic property of Color for ball color
- A constructor accepting a PointF and color, use NextDouble() to set velocity to a random value between -6 and 6 and radius to a random integer between 20 and 50
- Override Equals(), true if the balls would "touch" *different size here ! (don't forget GetHashCode)
- Move(), accepting a CDrawer (for boundaries), bounce and move inbounds if necessary
- Show(), accepting a CDrawer, and an int. Render the ball, then add the int number into the center of the ball with a text size of 15 in the complement of the ball color.**See Note

Now in your form add members : 3 List<> of your balls, representing Blue ball, Green balls, and Collided balls(Red), add your 2 CDrawer references assigned to null.

In the form constructor, make both CDrawer objects, a Timer with Tick callback and set both Drawer ContinuousUpdates to false.

All functionality is embedded in a timer event. When the timer event fires :

- If a new left click is available in the main drawer (Red/Green), create a new Red Ball, and if the Red List does not IndexOf it, Add() it to the Red List
- If a new right click is available in the main drawer(Red/Green), create a new Green Ball, and if the Green List does not Contains() it, Insert() it to the front of the Green List
- foreach(), all the Lists Move()ing all the Red/Green Balls in the main CDrawer
- use ForEach() and a lambda to Move() the Collided List in the secondary CDrawer.
- Save the Intersect() of the Red and Green balls to a new local temporary collided list via ToList() – this represents the Red vs. Green collisions (Red+Red and Green+Green do not collide)
- Iterate through your temp collision list and Remove() **ALL** those collided balls from your

Red and Green lists as well as set the collided ball to Yellow.

- Now combine the existing Collided balls with the new temporary collided list, ensuring no duplicates – these will be shown in the secondary window – using one or more of Concat(), Union(), Distinct() you must NOT use ToList() you must utilize constructors only.(when dups/collisions occur, only one will remain)
- Clear your CDrawers and add big faded text : Main window format " Red : XX Green : YY", and secondary format "N", where XX is Red ball count, YY is Green ball count and N is Collided ball count.
- Now for() each ball in Red and Green lists, show them in the main CDrawer, using the loop variable as their identifier number. Do the same with the Collide list in the secondary CDrawer, don't forget to Render()

**Note : For drawing the ball text, the AddText() method will center the text in the supplied rectangle (just like AddEllipse will fill the rectangle with the ellipse) So use a point offset by the radius and 2radii big for the text - voila' the text will be centered in the ball. For the complement color you need to invert the RGB values of a Color object, but without changing the Alpha(Opacity) so … `Color comp = Color.FromArgb( origColor.ToArgb() ^ 0x00FFFFFF );`