# CMPE2300 – StackyListyDraw – 2018 Version

In this ICA, you will operate a `Stack` of `LinkedList` to create a crummy drawing application.


**LineSeg**

Begin by creating a class called `LineSeg`. `LineSeg` will contain a `Point` for Start, `Point` for End, `byte` for Thickness, `float` for Alpha, and `Color` for Colour. All fields will be public automatic properties with private setters.

A line segment is a straight line that line collections are made of. Note: Alpha will be visually represented as a value of 0-255, but will be stored as a float (0-1).

Create a constructor to initialize/check the members (accepts all five arguments).

Add a render method that will accept a `CDrawer` and will render the line segment to that drawer.

**Main Form**

In the main form, create and initialize a `Stack` of `LinkedList` of `LineSeg`. Each linked list will contain the line segments that make up a line, and the stack will contain the collection of drawn lines.

Add a `CDrawer` of size 1024 x 768. Continuous update on.

Add a `bool` that will indicate if a line is currently being drawn – this will be known as the 'drawing flag'.

Add a `Point` that will save the last known mouse position.

Add a timer with a 10ms interval to the form. In the event hander for this timer you will manage the starting, continuation, and stopping of lines. Lines will be started when the left mouse button is clicked. A line will be stopped when the right mouse button is clicked. A line will be continued if the drawing flag is true and the mouse has moved. Do this by:

- Determine if the last mouse left-click is a new coordinate and the drawing flag is `false`. Set the drawing flag to `true`. Save the coordinates of the left click as the last known mouse position. `Push` a new linked list onto the stack. Return.
- Determine if the last mouse position is a new coordinate and the drawing flag is `true`. Add a new line segment from the last known mouse position to the current position. This line segment is added to the end of linked list on the top of the stack. Update the last known mouse position to be the current position. Render this line segment in place (just this line segment).
- Determine if the last mouse right-click is a new coordinate and the drawing flag is `true`. Set the drawing flag to `false`.

You may implement the above mechanism with drawer events if you wish, or with a different technique. You are responsible for whatever ripple changes this causes to your design.

In the UI, create a label that will show the line and segment stats as shown. The stats must always be accurate for the image shown. You should write a function to do this, and call it whenever the data in the application changes. Other UI elements must be represented/managed as shown/discussed.

Add a render method that will clear the drawer and render all the lines. This method will be required for the functions that follow.

In the 'Undo Line…' button event handler, pop the last line (if possible) and update the UI and drawing. Kill the drawing flag.

In the 'Undo Segment…' button event handler, remove the last line segment from the current line (if present). If you have removed all the segments from the line, pop the line. Update the UI and drawing. Kill the drawing flag.

In the 'Reduce Complexity…' button event handler, you will reduce the complexity of the current line. Reducing complexity requires that you consolidate two adjacent line segments into one. This requires that there be at least two line segments. Start at the front of the list and reduce by segment pairs until you run out of pairs. You are required to traverse the linked list manually. Note: walking through the linked list node by node will reveal `LinkedListNode` objects (not line segments).