**Problem Statement**

The goal of this assignment is to create an OpenGL program that implements the Cohen-Sutherland clipping algorithm. The program must use the algorithm to clip an assortment of randomly chosen lines that fall outside of the target window.

**Proposed Solution**

In the design phase of the assignment we discussed three main points: the requirements that the program needed to meet, how we were going to implement the algorithm, and how would we know our algorithm was successful. These three ideas helped our team focus on what needed to be discussed so each of us could contribute to the assignment. In the sections below we will outline the steps we took to complete our program.

**Strategies and Algorithms**

The requirements of the assignment helped structure the strategies that we used for our solution. We started by setting up the initial framework for the program. We used the Canvas class for this purpose. Canvas gave us quite a bit of functionality through the tools that it provides. All we needed to do was instantiate a Canvas object to utilize them.

After the framework was setup, we focused on the points that we were going to use to create the lines. One of the most important pieces of information that we needed for the algorithm was to identify where a point was in relation to the target window. To do this we used a series of bits to identify the location of that point. This information is stored in a struct that holds the Point2 data type from the canvas class and the screen location bits discussed above.

Now we needed to pair the points together to create our lines. We decided to create two arrays to hold the struct objects. The first array was named startPoint, and the second was named endPoint. To create our lines we aligned the indexes of the two arrays (startPoint[0] connects to endpoint[0]).

Now that each point has a bit sequence and our points are paired, we moved to the trivial accept and trivial reject algorithm. These algorithms check if the two points of a line can be trivially accepted or rejected. If the two points of that line are trivially accepted the line is drawn and no clipping takes place. If the algorithm is not trivially accepted then both points are either outside of the window or one point is inside and the other point is outside. In this scenario, the two points are passed to the clipping algorithm.

The clipping algorithm is responsible for clipping the lines and drawing them in the target window. The clipping is done by calculating a new point that is inside the target window. Lines that are clipped are drawn a different color than the lines that lie inside the window. This algorithm is discussed in more detail below.

**Mathematical Techniques**

The clipping algorithm uses the principle of similar triangles to calculate a new point that lies inside the target window. The new point is calculated by setting the x or y value to the edge of target window. After this is done, the other value is calculated using this example equation: (p1.y)new = (p1.y)old – e(dely/delx). This equation is for the calculating a new y point when the point is outside and to the right side of the screen. Other scenarios use a similar equation to calculate the new points for the line.

**Program Output/Results**

Our program uses the algorithms we just described and displays a window with several randomly placed lines. The main window has a simulated target window. The randomly placed lines that extend beyond the boundary of the target window are clipped using our algorithm. Each of the lines are color coded to indicate if they have been clipped and where the clipping took place. This lets us know that our algorithm worked to the assignments specifications.

**Summary**

This assignment asked us to implement a program that clips randomly placed lines based on the Cohen-Sutherland clipping algorithm. As a team, we discussed the requirements for implementing the algorithm. Once the requirements were outlined, we began developing the code required for the program. The output of our program is displayed to a graphics window so the accuracy of the results can be verified. The visual representations of the clipped lines are color coded to show exactly where they were clipped. Finally, our program is an efficient solution that fully meets the requirements specified by the assignment.