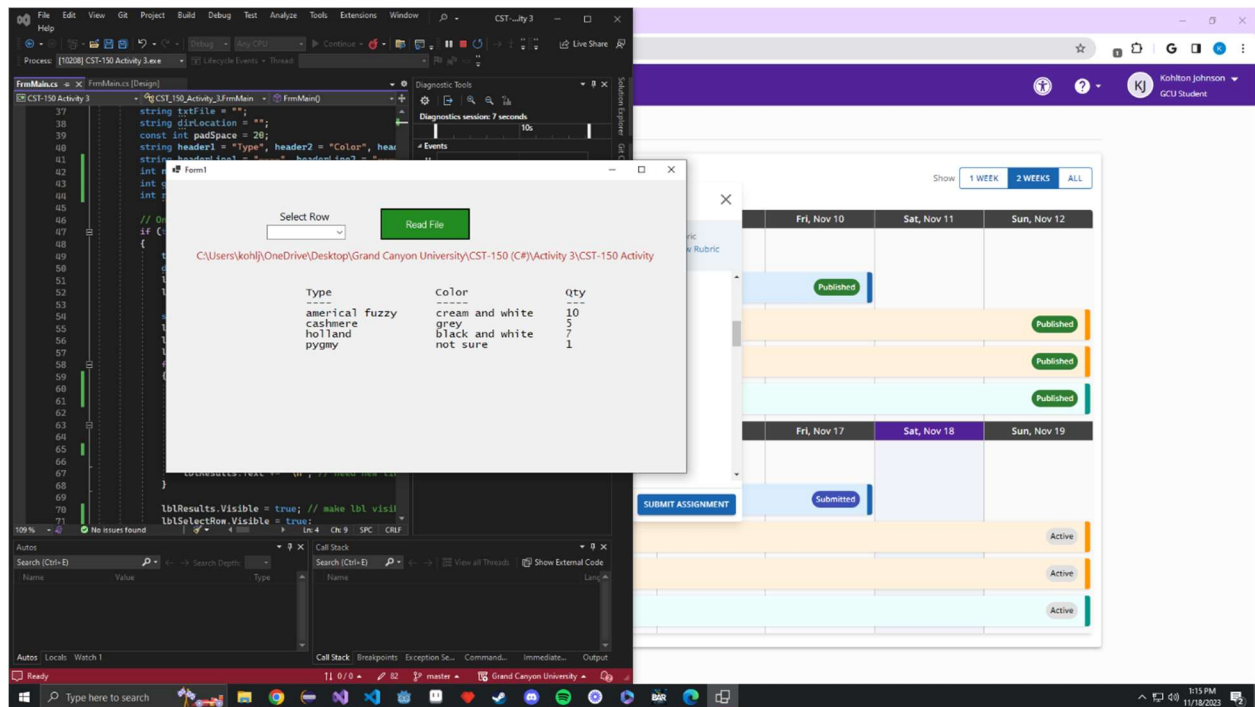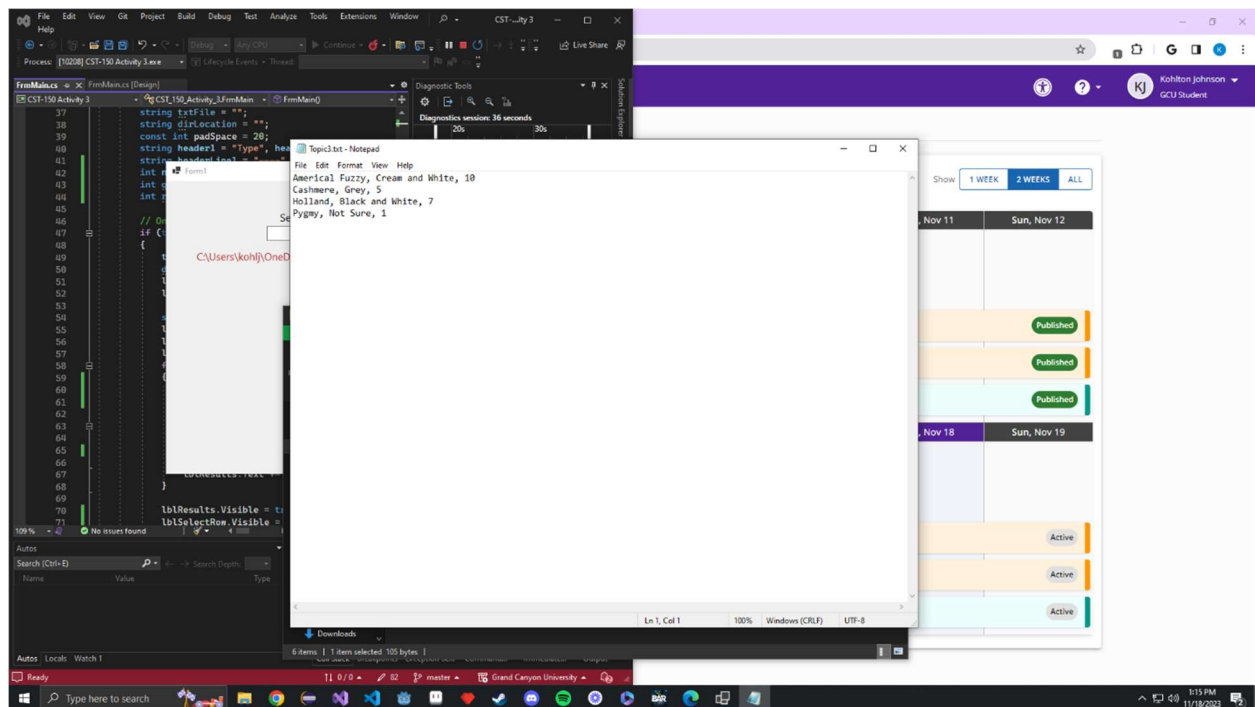Kohl Johnson

11-18-2023

Activity 4 Part 1

Pre-Form Population Screenshot

Shows the prepopulated form (before you selected a row).
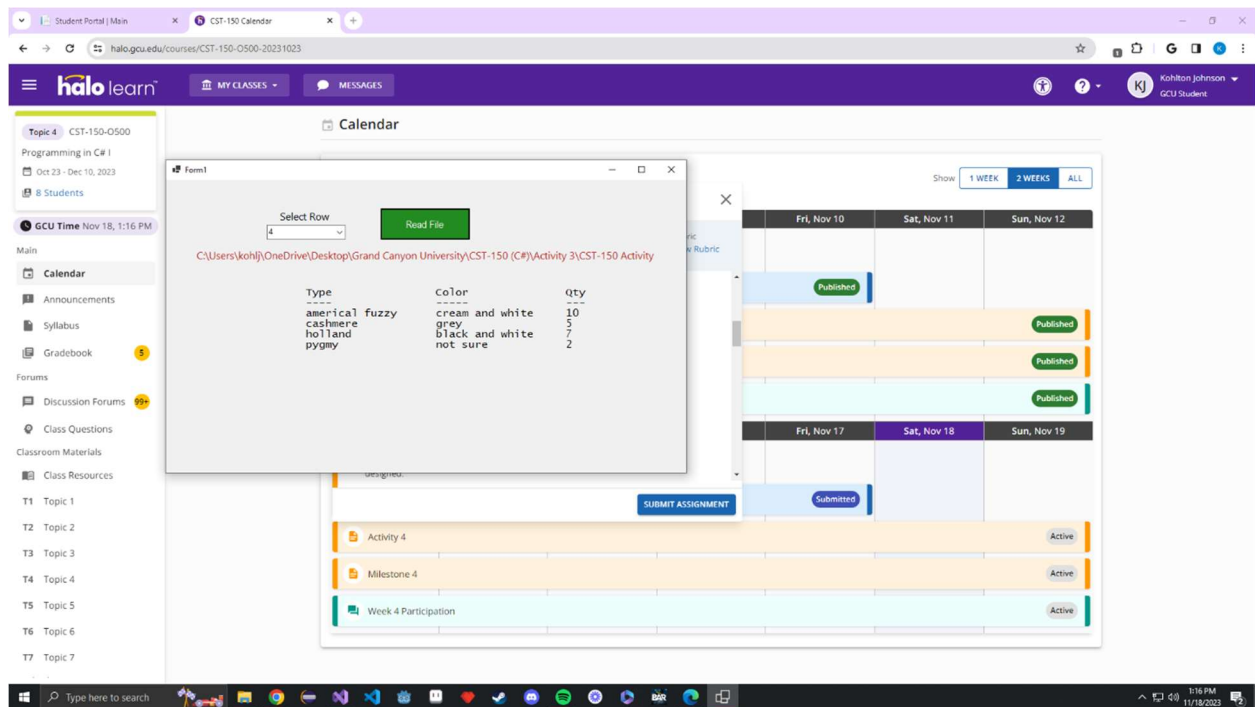
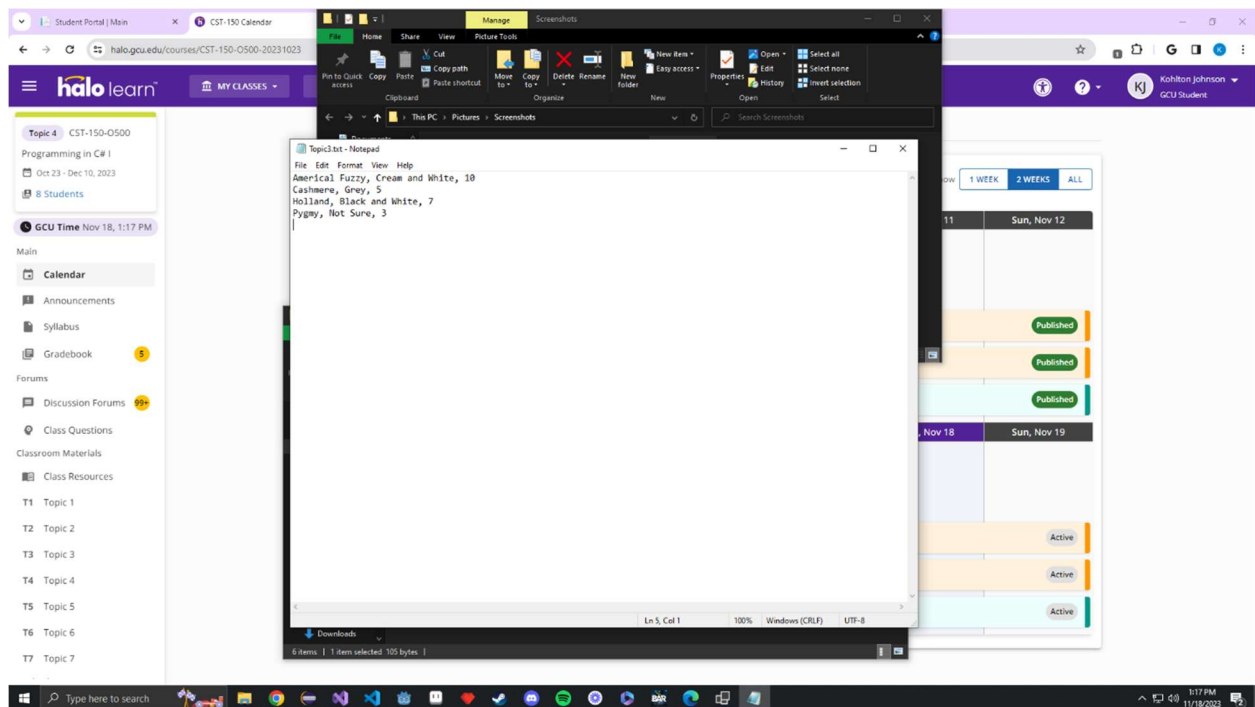**Original Text Document**

The original text document before being edited.
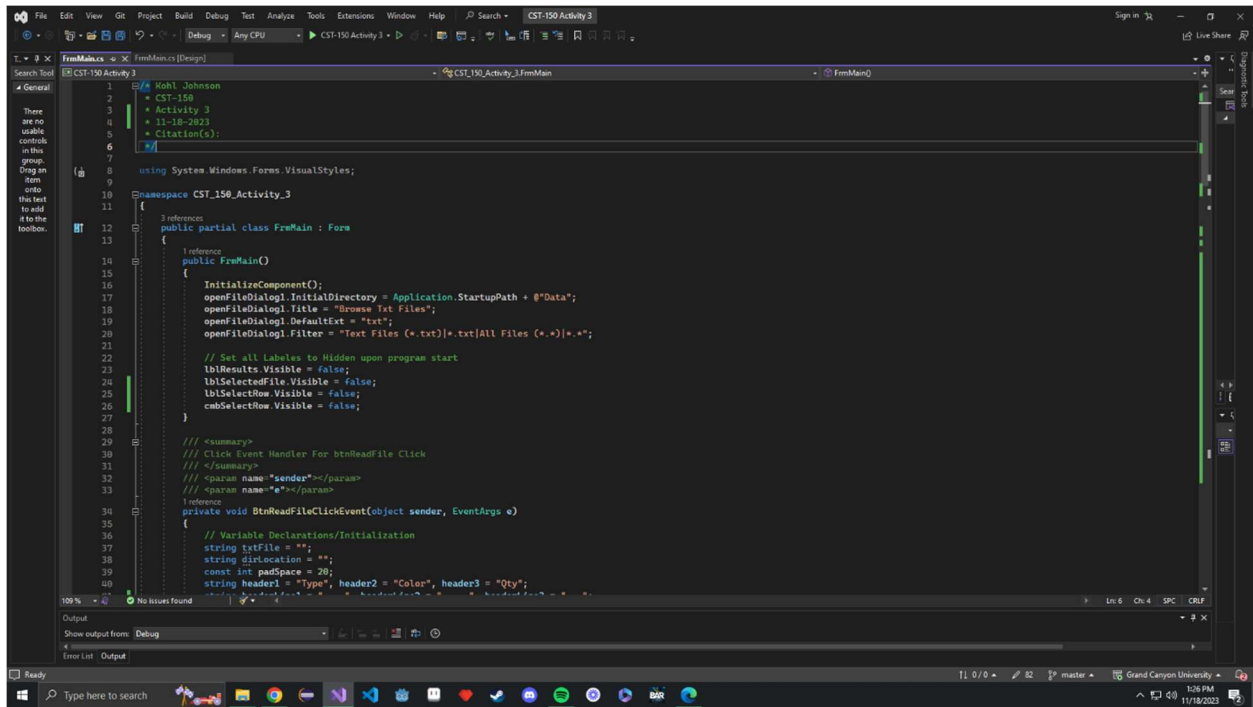
Post-Form Screenshot

Shows the form and inventory being updated after row 4 is selected in the "Select Row" dropdown box.
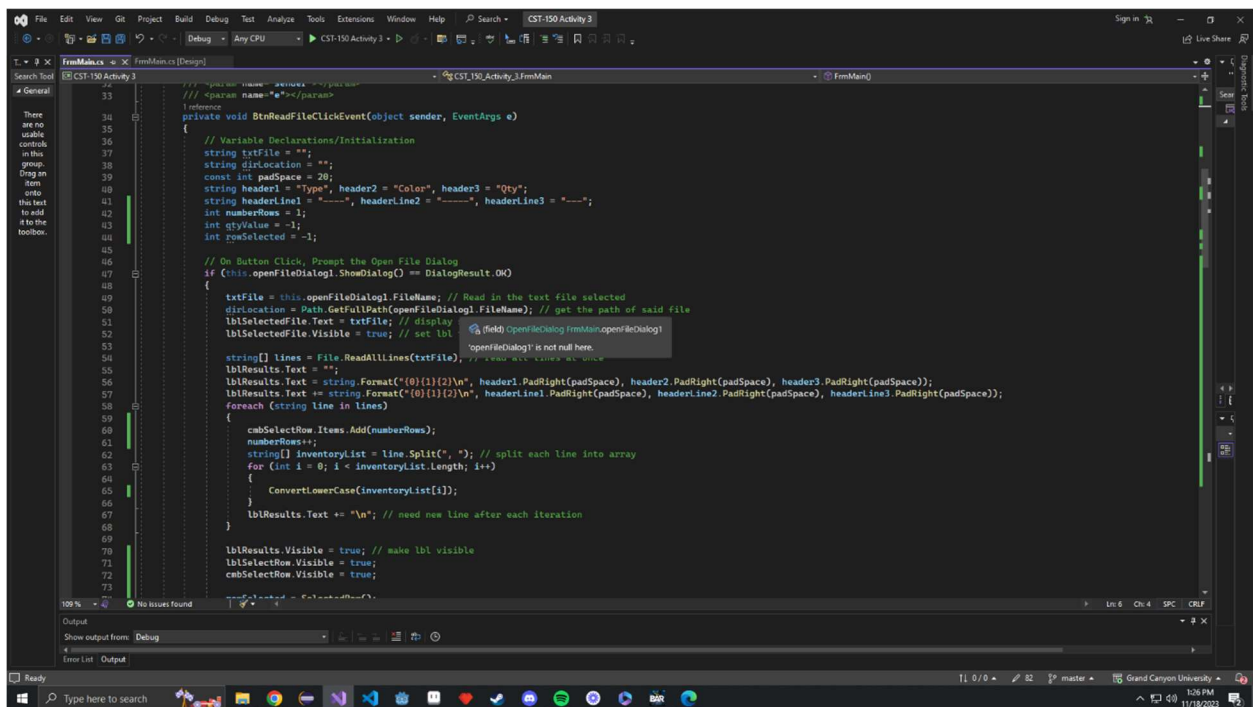
Edited Text Document

Shows the updated quantity. For some reason shows "3" when the original was "1", might be because I did the "Read File" dialog twice.
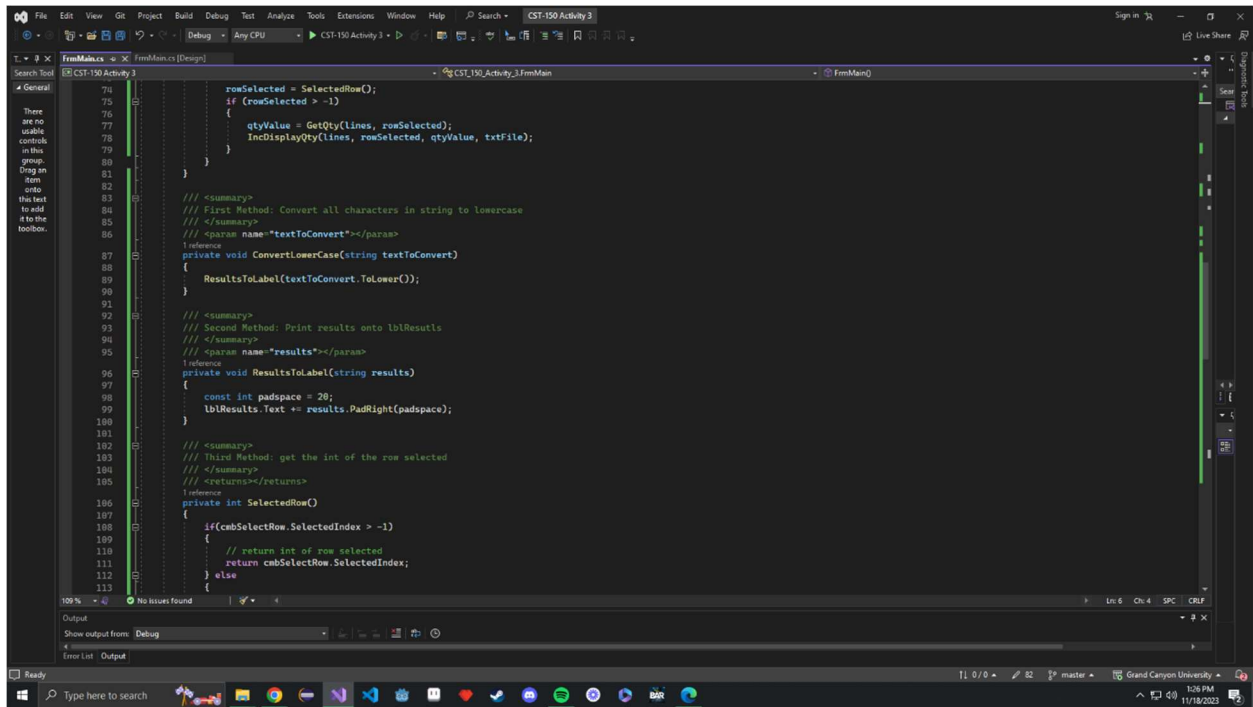
Code Screenshot 1


```csharp
/*  Kohl Johnson
 *  CST-150
 *  Activity 3
 *  11-18-2023
 *  Citation(s):
 */

using System.Windows.Forms.VisualStyles;

namespace CST_150_Activity_3
{

    public partial class FrmMain : Form
    {

        public FrmMain()
        {
            InitializeComponent();
            openFileDialog1.InitialDirectory = Application.StartupPath + @"Data";
            openFileDialog1.Title = "Browse Txt Files";
            openFileDialog1.DefaultExt = "txt";
            openFileDialog1.Filter = "Text Files (*.txt)|*.txt|All Files (*.*)|*.*";

            // Set all Labels to Hidden upon program start
            lblResults.Visible = false;
            lblSelectedFile.Visible = false;
            lblSelectRow.Visible = false;
            cmbSelectRow.Visible = false;
        }

        /// <summary>
        /// Click Event Handler For btnReadFile Click
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void BtnReadFileClickEvent(object sender, EventArgs e)
        {
            // Variable Declarations/Initialization
            string txtFile = "";
            string dirLocation = "";
            const int padSpace = 20;
            string header1 = "Type", header2 = "Color", header3 = "Qty";
```

Code Screenshot 2


```csharp
            /// <param name="sender"></param>
            /// <param name="e"></param>
            private void BtnReadFileClickEvent(object sender, EventArgs e)
            {
                // Variable Declarations/Initialization
                string txtFile = "";
                string dirLocation = "";
                const int padSpace = 20;
                string header1 = "Type", header2 = "Color", header3 = "Qty";
                string headerLine1 = "----", headerLine2 = "-----", headerLine3 = "---";
                int numberRows = 1;
                int qtyValue = -1;
                int rowSelected = -1;

                // On Button Click, Prompt the Open File Dialog
                if (this.openFileDialog1.ShowDialog() == DialogResult.OK)
                {
                    txtFile = this.openFileDialog1.FileName; // Read in the text file selected
                    dirLocation = Path.GetFullPath(openFileDialog1.FileName); // get the path of said file
                    lblSelectedFile.Text = txtFile; // display
                    lblSelectedFile.Visible = true; // set lbl
                                                      // (field) OpenFileDialog FrmMain.openFileDialog1
                    string[] lines = File.ReadAllLines(txtFile); // 'openFileDialog1' is not null here.
                    lblResults.Text = "";
                    lblResults.Text = string.Format("{0}{1}{2}\n", header1.PadRight(padSpace), header2.PadRight(padSpace), header3.PadRight(padSpace));
                    lblResults.Text += string.Format("{0}{1}{2}\n", headerLine1.PadRight(padSpace), headerLine2.PadRight(padSpace), headerLine3.PadRight(padSpace));
                    foreach (string line in lines)
                    {
                        cmbSelectRow.Items.Add(numberRows);
                        numberRows++;
                        string[] inventoryList = line.Split(", "); // split each line into array
                        for (int i = 0; i < inventoryList.Length; i++)
                        {
                            ConvertLowerCase(inventoryList[i]);
                        }
                        lblResults.Text += "\n"; // need new line after each iteration
                    }

                    lblResults.Visible = true; // make lbl visible
                    lblSelectRow.Visible = true;
                    cmbSelectRow.Visible = true;
```

Code Screenshot 3

```csharp
            rowSelected = SelectedRow();
            if (rowSelected > -1)
            {
                qtyValue = GetQty(lines, rowSelected);
                IncDisplayQty(lines, rowSelected, qtyValue, txtFile);
            }
        }
    }

    /// <summary>
    /// First Method: Convert all characters in string to lowercase
    /// </summary>
    /// <param name="textToConvert"></param>
    private void ConvertLowerCase(string textToConvert)
    {
        ResultsToLabel(textToConvert.ToLower());
    }

    /// <summary>
    /// Second Method: Print results onto lblResutls
    /// </summary>
    /// <param name="results"></param>
    private void ResultsToLabel(string results)
    {
        const int padspace = 20;
        lblResults.Text += results.PadRight(padspace);
    }

    /// <summary>
    /// Third Method: get the int of the row selected
    /// </summary>
    /// <returns></returns>
    private int SelectedRow()
    {
        if(cmbSelectRow.SelectedIndex > -1)
        {
            // return int of row selected
            return cmbSelectRow.SelectedIndex;
        } else
        {
```
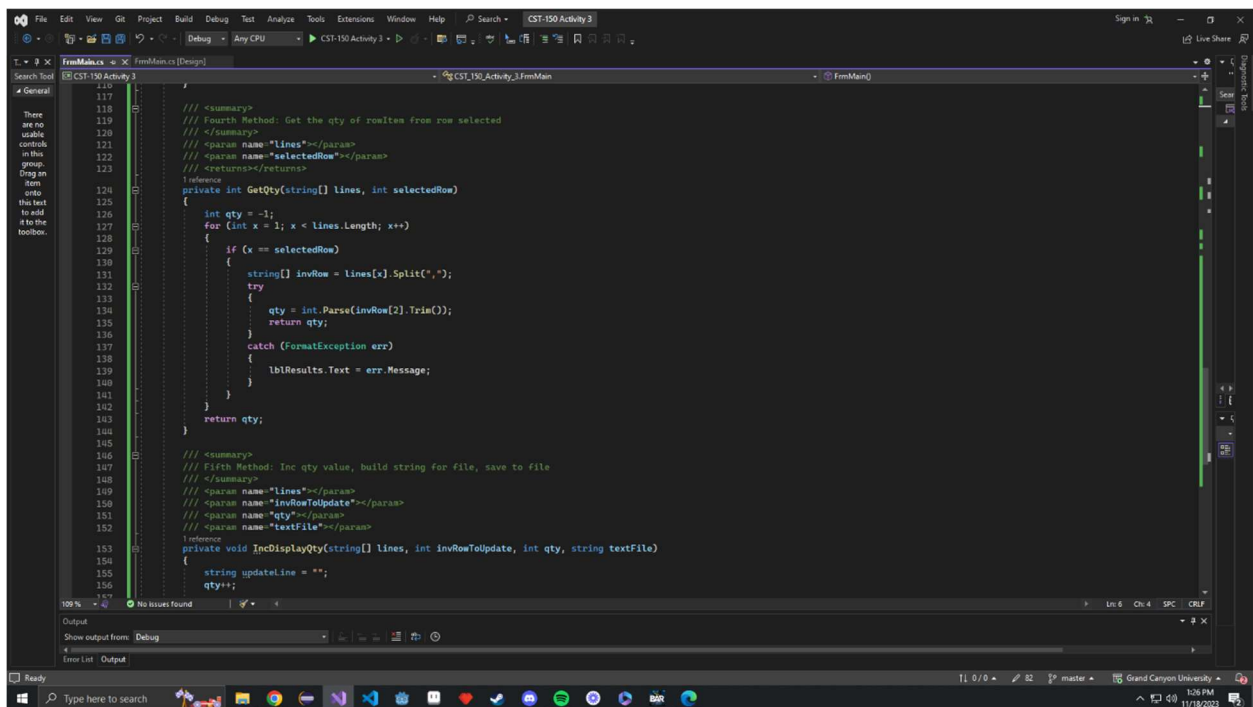
Code Screenshot 4

```csharp
    /// <summary>
    /// Fourth Method: Get the qty of rowItem from row selected
    /// </summary>
    /// <param name="lines"></param>
    /// <param name="selectedRow"></param>
    /// <returns></returns>
    private int GetQty(string[] lines, int selectedRow)
    {
        int qty = -1;
        for (int x = 1; x < lines.Length; x++)
        {
            if (x == selectedRow)
            {
                string[] invRow = lines[x].Split(",");
                try
                {
                    qty = int.Parse(invRow[2].Trim());
                    return qty;
                }
                catch (FormatException err)
                {
                    lblResults.Text = err.Message;
                }
            }
        }
        return qty;
    }

    /// <summary>
    /// Fifth Method: Inc qty value, build string for file, save to file
    /// </summary>
    /// <param name="lines"></param>
    /// <param name="invRowToUpdate"></param>
    /// <param name="qty"></param>
    /// <param name="textFile"></param>
    private void IncDisplayQty(string[] lines, int invRowToUpdate, int qty, string textFile)
    {
        string updateLine = "";
        qty++;
```

Code Screenshot 5


```
            }
        }
    }
    return qty;
}

/// <summary>
/// Fifth Method: Inc qty value, build string for file, save to file
/// </summary>
/// <param name="lines"></param>
/// <param name="invRowToUpdate"></param>
/// <param name="qty"></param>
/// <param name="textFile"></param>
1 reference
private void IncDisplayQty(string[] lines, int invRowToUpdate, int qty, string textFile)
{
    string updateLine = "";
    qty++;

    string[] invRow = lines[invRowToUpdate].Split(",");
    invRow[2] = qty.ToString();
    updateLine = invRow[0].Trim() + ", " + invRow[1].Trim() + ", " + invRow[2].Trim();
    lines[invRowToUpdate] = updateLine;
    File.WriteAllLines(textFile, lines);
}
```