#### CIS 160

#### Week 7 Team Project

Purpose: Create programs given more freedom than in labs. Exercise teamwork and communication as related to technical projects.

Task: Work in teams of two to create a C# program for a company.

Your project must meet the following requirements:

1. Each team will create a single C# program.
2. The names of the team members must be in a comment at the top of the C# file(s) for the program.
3. Use the technologies we've covered in weeks 1-4, including:
   1. Data types
   2. Variables
   3. Relational operators
   4. Selection structures
   5. Loops
   6. Methods
   7. GUI controls
   8. Data validation and/or exception handling
4. You are welcome to find example C# code on-line. If you include nontrivial code from someone else's example, please include the URL in a comment in the program.  
   *You will still receive full credit for your project if you cite your source.*  
   Code included without citation can have serious academic consequences.

For instance:

// Code from <http://www.java-examples.com/while-loop/example>

int i = 0;

while(i < 5) {

System.out.println("i is : " + i);

i++;

}

1. The program must work!
2. Creativity and extra effort will be rewarded.
3. When the project is complete, please zip all project files together and turn in the zip file to the professor on-line. The program must run correctly on the instructor computer when run from Visual Studio.
   1. I strongly urge you to try running your project on a different computer than you usually use, to be sure you’ve included all required files.

8. Please paste your source code and a screenshot of a successful program run into a Word document for submission. A sample submission is listed later in this document. If you have several C# files, please paste the code from each into Word sequentially so I can reconstruct your project if needed.

9. You will submit a .zip file with both:

9.1. The source folder for your project

9.2. A Word document with your source code and screenshots.

10. Your team will demonstrate your completed program to the class according to the Canvas calendar.

* 1. **Projects must be submitted by end-of-day the day before the presentation.**
  2. Only one team member needs to submit the project.
  3. You can only present using what is submitted to Canvas.
  4. The presentation structure: introduce yourselves and your project, run the program,   
     show off one thing in the source code, give us a minute for questions
  5. Presentations should run five minutes or less
  6. I recommend you practice your presentation before the actual event!

CHECKLIST FOR SUBMISSION

* Team member names and program topic are posted to the team project discussion thread
* The team has created a C# program
* All team members have their names in the comments of the C# file(s)
* All required technologies are used: see item 3 for list
* Programs with input must work with *any* suitable input, not just what you demo/test
* Your submission includes:
  + The source folder with all required files to run on the instructor machine
  + The Word project submission document with screenshots and copy/pasted source code
* Submission is in by end-of-day before the presentation

RUBRIC

Student name and date is in a comment on the first line of the program: -5 points if fails

All required technologies used: 50 points

(that’s about 6 points per technology listed in #3)

Program works correctly: 10 points

Program works from files submitted: 10 points

Presentation: 20 points

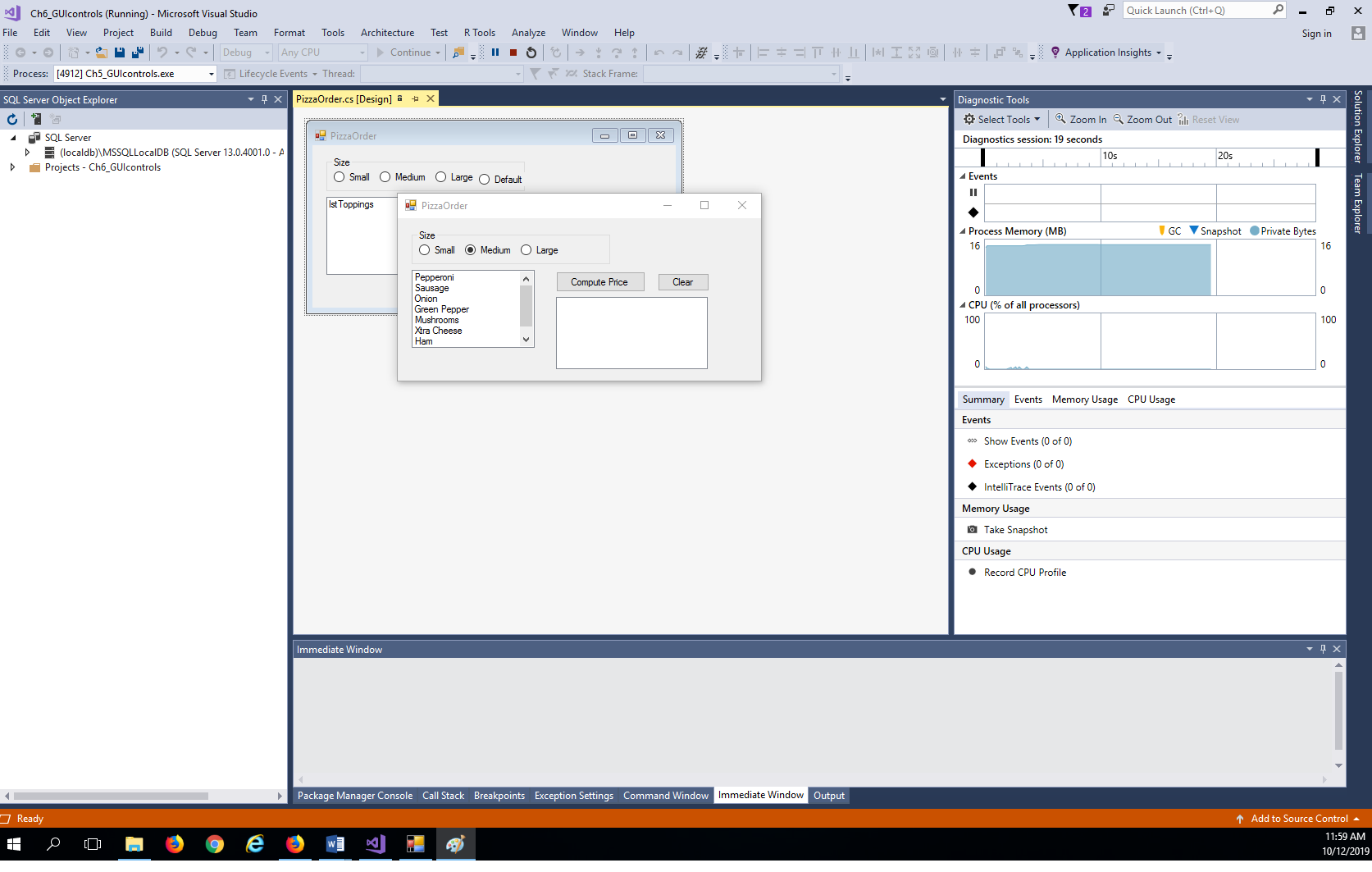
Creativity: 10 points

SAMPLE C# FILE COMMENT

// Program by Miranda Cartel and Joseph Leoman 12/1/2025

# SAMPLE CIS 160 WEEK 7 TEAM PROJECT SUBMISSION PAGE

# By Miranda Cartel and Joseph Leoman



// Program by Miranda Cartel and Joseph Leoman 12/1/2025

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Ch5\_GUIcontrols

{

public partial class PizzaOrder : Form

{

public PizzaOrder()

{

InitializeComponent();

}

// This form uses a ListBox to let users select multiple items from a list:

private void PizzaOrder\_Load(object sender, EventArgs e)

{

string[] toppings = {"Pepperoni","Sausage", "Onion", "Green Pepper",

"Mushrooms", "Xtra Cheese", "Ham", "Anchovies"};

foreach (string topping in toppings) // Add toppings to ListBox

lstToppings.Items.Add(topping);

rdbDefault.Checked = true;

}

private void btnCompute\_Click(object sender, EventArgs e)

{

// Verify that the user selected size:

if (rdbDefault.Checked == true)

{

MessageBox.Show("Please select size");

return;

}

// Invoke UnitPrice to get base price and cost per topping

var unitPrice = UnitPrice();

float basePrice = unitPrice.basePrice;

float costPerTopping = unitPrice.costPerTopping;

// Compute and display price:

float price = basePrice + costPerTopping \* lstToppings.SelectedItems.Count;

// SelectedItems is a collection of all selected items

lstDisplay.Items.Clear();

lstDisplay.Items.Add ("Price: " + price.ToString("C2"));

// Display Toppings:

lstDisplay.Items.Add("");

lstDisplay.Items.Add ("Toppings");

if (lstToppings.SelectedItems.Count == 0) // if no item is selected

lstDisplay.Items.Add(" None");

else

{

foreach (string topping in lstToppings.SelectedItems)

lstDisplay.Items.Add(" " + topping);

}

}

private (float basePrice, float costPerTopping) UnitPrice()

{

float basePrice = 0, costPerTopping = 0;

if (rdbSmall.Checked)

{

basePrice = 6.95F;

costPerTopping = 1.5F;

}

else if (rdbMedium.Checked)

{

basePrice = 8.95F;

costPerTopping = 2.0F;

}

else if (rdbLarge.Checked)

{

basePrice = 11.95F;

costPerTopping = 2.5F;

}

return (basePrice, costPerTopping);

}

private void btnClear\_Click(object sender, EventArgs e)

{

rdbDefault.Checked = true;

lstToppings.SelectedIndex = -1;

lstDisplay.Items.Clear();

}

}

}

HOW TO APPROACH THIS PROJECT

Day 1: Team up, form project plan, start initial coding

* + Form your team
  + Choose from the instructor-provided list of programs, or select your own and get instructor approval
  + Post to the team project thread!
  + Sketch out the big picture of what coding is required and how it can be split up
  + Start coding your individual pieces
  + Make sure to start with something small that works and build up!

Development days: Coordinate with your team

* + Make final adjustments to individual code
  + Combine your programs
  + Debug any issues raised by the new combination
  + Verify that the program loads correctly from the extracted .zip file you will submit for the assignment

Last day: Show off!

* + The file must be submitted to the instructor by midnight the morning of the presentations. Files submitted after this time *will not be accepted for presentation*.
  + The instructor will copy and unzip your team’s submissions to the instructor workstation.
  + Present using the overall structure outline above.

WEEK 7 TEAM PROJECT PROGRAM SUGGESTIONS

1. Calculator  
Create a calculator that allows the user to add/subtract/multiple/divide, using windows buttons and forms controls. Be sure to validate user input! Some creativity points will be given for whimsy, using numeric UpDown controls, or clever/attractive form design.

1a. Calculator (advanced, more creativity/effort points)  
Create a calculator as in #1, but include buttons that allow the user to change the data type of the input and output.

2. Bar Chart  
Write a program that asks the user to enter today’s sales for five stores. The program should display a bar chart comparing each store’s sales. Create each bar in the bar chart by displaying a row of asterisks. Each asterisk should represent $100 of sales. Here is an example of the program’s input/output, although you will want to do something with Windows forms controls for formatting Feel free to use multiple labels or textboxes to display the bar chart.

Input:

Enter today's sales for store 1: 1000

Enter today's sales for store 2: 1200

Enter today's sales for store 3: 1800

Enter today's sales for store 4: 800

Enter today's sales for store 5: 1900

Output:

SALES BAR CHART

Store 1: \*\*\*\*\*\*\*\*\*\*

Store 2: \*\*\*\*\*\*\*\*\*\*\*\*

Store 3: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Store 4: \*\*\*\*\*\*\*\*

Store 5: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2a. Bar chart (advanced, more creativity/effort points)  
Create a bar chart as in #2, but use a multiline textbox to display the output all together.

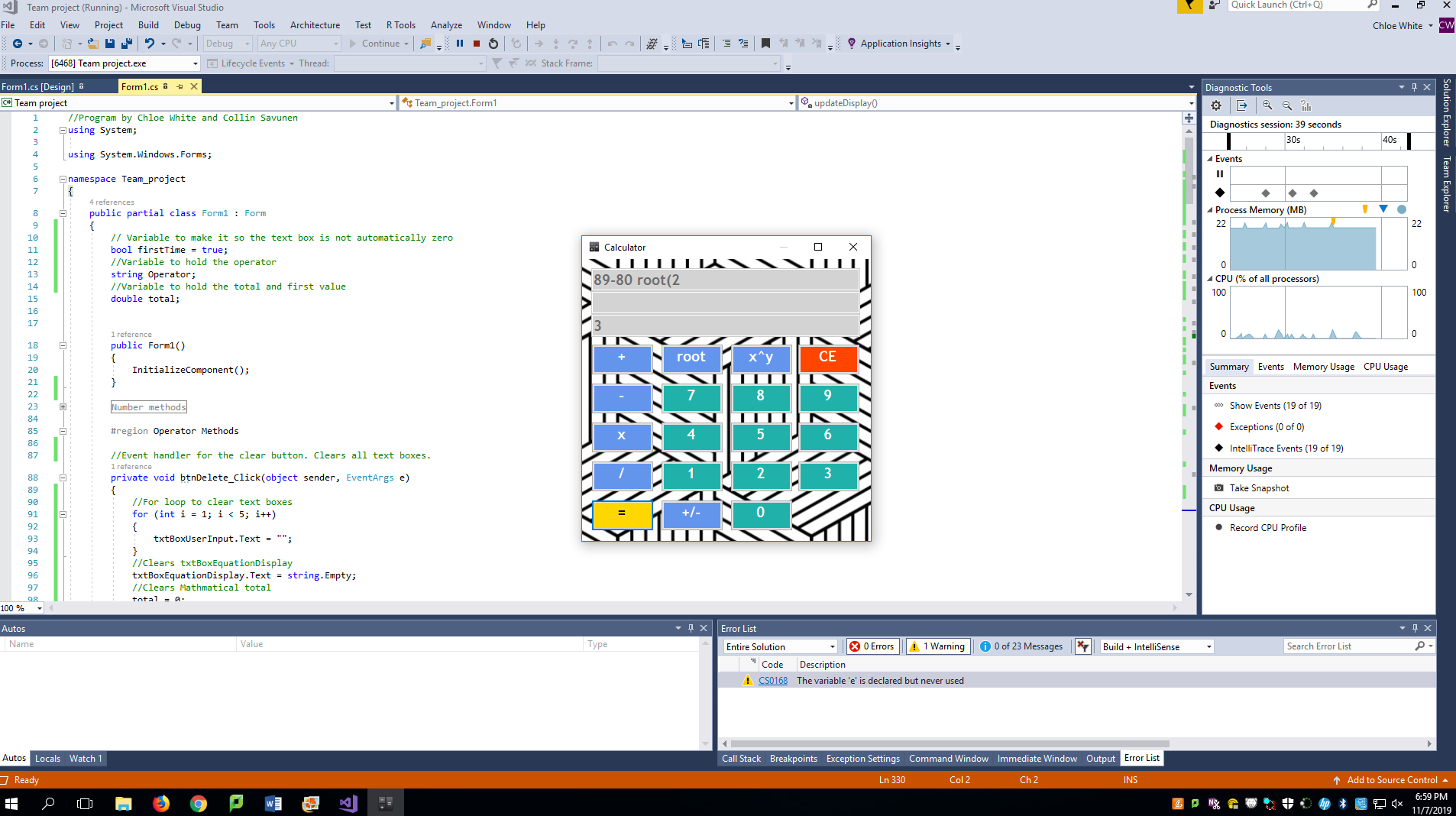
3. Write a program that reads in five test scores. The program should display a letter grade for each score and the average test score. Write the following methods in the program:

• calcAverage—This method should accept five test scores as arguments and return the average of the scores.

• determineGrade—This method should accept a test score as an argument and return a letter grade for the score, based on the following grading scale:

90–100 A, 80–89 B, 70–79 C, 60–69 D, Below 60 F

4. Additional student ideas will be accepted within the first hour of team formation. Please ask the instructor to approve your idea before moving forward. In practice almost all student ideas get approved, but professor input can help keep the project scope reasonable.



//Program by Chloe White and Collin Savunen

using System;

using System.Windows.Forms;

namespace Team\_project

{

public partial class Form1 : Form

{

// Variable to make it so the text box is not automatically zero

bool firstTime = true;

//Variable to hold the operator

string Operator;

//Variable to hold the total and first value

double total;

public Form1()

{

InitializeComponent();

}

#region Number methods

private void btnNum0\_Click(object sender, EventArgs e)

{

inputValue("0");

inputFocus();

}

private void btnNum1\_Click(object sender, EventArgs e)

{

inputValue("1");

inputFocus();

}

private void btnNum2\_Click(object sender, EventArgs e)

{

inputValue("2");

inputFocus();

}

private void btnNum3\_Click(object sender, EventArgs e)

{

inputValue("3");

inputFocus();

}

private void btnNum4\_Click(object sender, EventArgs e)

{

inputValue("4");

inputFocus();

}

private void btnNum5\_Click(object sender, EventArgs e)

{

inputValue("5");

inputFocus();

}

private void btnNum6\_Click(object sender, EventArgs e)

{

inputValue("6");

inputFocus();

}

private void btnNum7\_Click(object sender, EventArgs e)

{

inputValue("7");

inputFocus();

}

private void btnNum8\_Click(object sender, EventArgs e)

{

inputValue("8");

inputFocus();

}

private void btnNum9\_Click(object sender, EventArgs e)

{

inputValue("9");

inputFocus();

}

#endregion

#region Operator Methods

//Event handler for the clear button. Clears all text boxes.

private void btnDelete\_Click(object sender, EventArgs e)

{

//For loop to clear text boxes

for (int i = 1; i < 5; i++)

{

txtBoxUserInput.Text = "";

}

//Clears txtBoxEquationDisplay

txtBoxEquationDisplay.Text = string.Empty;

//Clears Mathmatical total

total = 0;

//resets firsTime to true in order to make textbox not automatically 0

firstTime = true;

//refocouses on txtBoxUserInput

txtBoxUserInput.Focus();

//Clears txtBoxAns

txtBoxAns.Text = "";

}

//Method to do division

private void btnDiv\_Click(object sender, EventArgs e)

{

inputValue("/");

ButtonPress();

Operator = "/";

}

//Method to do Multiplication

private void btnMult\_Click(object sender, EventArgs e)

{

inputValue("x");

ButtonPress();

Operator = "x";

}

//Method to do Subtraction

private void btnMinus\_Click(object sender, EventArgs e)

{

inputValue("-");

ButtonPress();

Operator = "-";

}

//Method to do Addition

private void btnAdd\_Click(object sender, EventArgs e)

{

inputValue("+");

ButtonPress();

Operator = "+";

}

//Method to do Power operations

private void btnPower\_Click(object sender, EventArgs e)

{

inputValue("^");

ButtonPress();

Operator = "^";

}

//Method to do Root operations

private void btnSqrt\_Click(object sender, EventArgs e)

{

inputValue(" root(");

ButtonPress();

Operator = "(sqrt)";

}

//Method to invert the sign of the currentNumber

private void btnNegative\_Click(object sender, EventArgs e)

{

var currentNumber = Convert.ToDouble(txtBoxUserInput.Text) \* -1;

txtBoxUserInput.Text = currentNumber.ToString();

}

#endregion

#region Helper Methods

//Method to change focus to txtBoxUserInput

private void inputFocus()

{

txtBoxUserInput.Focus();

}

//Method to input values in txtBoxEquationDisplay

private void inputValue(string value)

{

double e;

if (value == "0" || value == "1" || value == "2" || value == "3" || value == "4" || value == "5" || value == "6" || value == "7" || value == "8" || value == "9")

{

//insert value at end of text in txtBoxEquationDisplay

txtBoxEquationDisplay.Text = txtBoxEquationDisplay.Text.Insert(txtBoxEquationDisplay.Text.Length, value);

//inserts value in txtBoxUserInput

txtBoxUserInput.Text = txtBoxUserInput.Text + value;

}

else

//insert value at end of text

txtBoxEquationDisplay.Text = txtBoxEquationDisplay.Text.Insert(txtBoxEquationDisplay.Text.Length, value);

}

//Method to update display of txtBoxUserInput

private void updateDisplay()

{

txtBoxAns.Text = total.ToString();

txtBoxUserInput.Text = "";

txtBoxUserInput.Focus();

}

private void ButtonPress()

{

if (firstTime)

{

firstTime = false;

total = Convert.ToDouble(txtBoxUserInput.Text);

txtBoxUserInput.Text = "";

txtBoxUserInput.Focus();

return;

}

calculate();

updateDisplay();

}

#endregion

#region calculate

private void calculate()

{

double currentNumber = Convert.ToDouble(txtBoxUserInput.Text);

try

{

if (Operator == "+")

{

total = total + currentNumber;

}

else if (Operator == "-")

{

total = total - currentNumber;

}

else if (Operator == "/")

{

total = total / currentNumber;

}

else if (Operator == "x")

{

total = total \* currentNumber;

}

else if (Operator == "^")

{

total = Math.Pow(total, currentNumber);

}

else if (Operator == "(sqrt)")

{

total = Math.Pow(total, (1 / currentNumber));

}

}

catch (FormatException)

{

MessageBox.Show("Please input a number or an operator!");

}

catch (OverflowException)

{

MessageBox.Show("An OverflowException occured!");

}

catch (DivideByZeroException)

{

MessageBox.Show("You cannot divide by zero!");

}

catch (Exception ex)

{

MessageBox.Show("An error occured!");

}

//Displays total

txtBoxAns.Text = Convert.ToString(total);

}

private void btnEqual\_Click(object sender, EventArgs e)

{

calculate();

updateDisplay();

}

#endregion

#region Noted out code

//From btnEqual\_Click

//double secondNumber;

//double Result;

//var x = txtBoxUserInput.Text;

//secondNumber = Convert.ToDouble(txtBoxUserInput.Text);

//if (Operator == "+")

//{

// Result = (firstNumber + secondNumber);

// //txtBoxUserInput.Text = "";

// txtBoxAns.Text = Convert.ToString(Result);

// firstNumber = Result;

//}

//if (Operator == "-")

//{

// Result = (firstNumber - secondNumber);

// txtBoxAns.Text = Convert.ToString(Result);

//}

//if (Operator == "/")

//{

// Result = (firstNumber / secondNumber);

// txtBoxAns.Text = Convert.ToString(Result);

//}

//if (Operator == "x")

//{

// Result = (firstNumber \* secondNumber);

// txtBoxAns.Text = Convert.ToString(Result);

//}

//From Varible declaration in public partial class Form1 : Form

//double firstNumber;

//double secondNumber;

//from operator methods

// private void btnLParenthesis\_Click(object sender, EventArgs e)

// {

// txtBoxUserInput.Text += btnLParenthesis.Text;

// }

// private void btnRParenthesis\_Click(object sender, EventArgs e)

// {

// txtBoxUserInput.Text += btnRParenthesis.Text;

// }

//From btnAdd\_Click

//firstNumber = Convert.ToDouble(txtBoxUserInput.Text);

//txtBoxUserInput.Text += btnAdd.Text;

//txtBoxUserInput.Text += btnAdd.Text;

#endregion

private void Form1\_Load(object sender, EventArgs e)

{

}

}

}