PROG1017

Memo

|  |  |
| --- | --- |
| To: | Cathy Burchill |
| From: | Heather Taylor |
| Date: | December 6, 2022 |
| Re: | Lab # 5 |

This lab went much more smoothly than the last one. Using a form load event sure helps!

I spent about 6-7 hours total on this lab, which was mostly rearranging code so that it makes sense and is functional.

The biggest barrier I faced was with the login feature. I didn’t understand how to access the randomly generated number but I asked and it was made clear to just compare the labels. Also, increasing the counter was giving me trouble as I had it within the login button click event, not outside. This made it re-initialize to 0 each time the button was clicked.

Functions are in blue and Events are in yellow.

public Form1()

{

InitializeComponent();

}

/\*Intro to Programming Lab #5

\*For Cathy Burchill by Heather Taylor

\*December 6, 2022

\*This lab generates random numbers in order to have a user log in.

\*Once logged in they may choose between analyzing 2 inputted strings or a randomly generated list of numbers.\*/

const string PROGRAMMER = "Heather Taylor";

/\*Name: GetRandom

\*Send: min (int), max(int)

\*Return: int

\*This function generates a random number between 100,000 and 200,000 for an authentication code.\*/

private int GetRandom(int min, int max)

{

int authentication;

Random rand = new Random();

authentication = rand.Next(min, max);

return authentication;

}

//Form Load - Hides all grp except grpLogin, creates authentication code

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

{

this.Text += " " + PROGRAMMER;

txtCode.Focus();

grpChoose.Hide();

grpStats.Hide();

grpText.Hide();

int authenticationCode = GetRandom(100000, 200001);

lblCode.Text = authenticationCode.ToString();

}

/\*Name: ResetTextGrp

\*Send: none

\*Return: none

\*This function resets the Text Group Box.\*/

private void ResetTextGrp()

{

txtString1.Text = "";

txtString2.Text = "";

lblResults.Text = "";

txtString1.Focus();

chkSwap.Checked = false;

this.AcceptButton = btnJoin;

this.CancelButton = btnReset;

}

/\*Name: ResetStatsGrp

\*Send: none

\*Return: none

\*This function resets the Stats Group Box.\*/

private void ResetStatsGrp()

{

nudHowMany.Value = 10;

this.AcceptButton = btnGenerate;

this.CancelButton = btnClear;

lblSum.Text = "";

lblMean.Text = "";

lblOdd.Text = "";

lstNumbers.Items.Clear();

}

/\*Name: SetupOption

\*Send: none

\*Return: none

\*This will show or hide the proper group box based on the selected radio button\*/

private void SetupOption()

{

if(radText.Checked)

{

grpText.Show();

grpStats.Hide();

ResetTextGrp();

}

else

{

grpStats.Show();

grpText.Hide();

ResetStatsGrp();

}

}

//Counter variable for login attempts

int tries = 0;

//login button is clicked

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

{

tries++;

if (txtCode.Text != lblCode.Text)

{

if (tries < 3)

{

string warning = tries + " incorrect code(s) entered\nTry again - only 3 attempts allowed";

MessageBox.Show(warning, PROGRAMMER);

txtCode.SelectAll();

txtCode.Focus();

}

if (tries >= 3)

{

MessageBox.Show("3 attempts to login\nAccount locked - Closing program", PROGRAMMER);

this.Close();

}

}

else

{

grpChoose.Show();

radText.Checked = true;

grpLogin.Enabled = false;

}

}

//radText is selected

private void radText\_CheckedChanged(object sender, EventArgs e)

{

SetupOption();

}

//radStats is selected

private void radStats\_CheckedChanged(object sender, EventArgs e)

{

SetupOption();

}

/\*Name: Swap

\*Send: 2 Strings

\*Return: None

\*This Function swaps the inputs from txtString1 and txtString2\*/

private void Swap(ref string stringOne, ref string stringTwo)

{

string temporary;

if (chkSwap.Checked)

{

temporary = stringOne;

stringOne = stringTwo;

stringTwo = temporary;

txtString1.Text = stringOne;

txtString2.Text = stringTwo;

lblResults.Text = "Strings have been swapped!";

}

else

lblResults.Text = "";

}

/\*Name: CheckInput

\*Send: None

\*Return: bool

\*This checks that both txtString1 and txtString2 have value. \*/

private bool CheckInput()

{

bool validInput;

if(txtString2.Text != "" && txtString1.Text != "")

validInput = true;

else

validInput = false;

return validInput;

}

//chkSwap is checked. It will swap strings 1 and 2 by calling the Swap function

private void chkSwap\_CheckedChanged(object sender, EventArgs e)

{

bool validInput = CheckInput();

string stringOne = txtString1.Text;

string stringTwo = txtString2.Text;

if(validInput)

Swap(ref stringOne, ref stringTwo);

}

//Joins the strings together in lblResults

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

{

bool validInput = CheckInput();

string stringOne = txtString1.Text;

string stringTwo = txtString2.Text;

lblResults.Text = "";

if (validInput)

{

lblResults.Text += "First string = " + stringOne + "\nSecond String = " + stringTwo +

"\nJoined = " + stringOne + "-->" + stringTwo;

}

}

//Analyzes the number of characters in each string and displays data to the user

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

{

bool validInput = CheckInput();

string stringOne = txtString1.Text;

string stringTwo = txtString2.Text;

if(validInput)

{

lblResults.Text = "";

int length1 = stringOne.Length;

int length2 = stringTwo.Length;

lblResults.Text += "First string = " + stringOne + "\n Characters = " + length1 +

"\nSecond string = " + stringTwo + "\n Characters = " + length2;

}

}

//Resets grpText

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

{

ResetTextGrp();

}

/\*Name:AddList

\*Send: none

\*Return: int

\*This function sums the values of the numbers in lstNumbers\*/

private int AddList()

{

int counter = 0;

int sum = 0;

while(counter < nudHowMany.Value)

{

int value = Convert.ToInt32(lstNumbers.Items[counter]);

sum += value;

counter++;

}

return sum;

}

/\*Name: CountOdd

\*Send: None

\*Return: int

\*This function count the number of items in the list that are odd.\*/

private int CountOdd()

{

int counter = 0;

int numOdd = 0;

do

{

int value = Convert.ToInt32(lstNumbers.Items[counter]);

int oddCheck = value % 2;

if (oddCheck >= 1)

{

numOdd++;

counter++;

}

else

{

counter++;

}

} while (counter < nudHowMany.Value);

return numOdd;

}

//Generates random numbers in the list box, displays sum in lblSum, mean in lblMean and number of odd numbers in lblOdd

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

{

Random rand = new Random(733);

lstNumbers.Items.Clear();

for(int generation = 0; generation < nudHowMany.Value; generation++)

{

int listNum = rand.Next(1000, 5001);

lstNumbers.Items.Add(listNum);

}

double sum = AddList();

lblSum.Text = sum.ToString("n0");

double mean = sum / lstNumbers.Items.Count;

lblMean.Text = mean.ToString("n");

int numOdd = CountOdd();

lblOdd.Text = numOdd.ToString();

}

//Resets grpStats

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

{

ResetStatsGrp();

}

}

}