|  |
| --- |
| College LaSalle |
| Project - Oriented Object Programming User and Technical Manual |
|  |
| Presented to: Mihai Maftei. |

|  |
| --- |
| Your name: Queen Sarah Anumu Bih  7/11/2023  2023/07/18  2023/07/25  Version 1, 2,3 |

1. **Start by adding a short description of your project, and the languages (technologies) used:**
2. C# Language
3. Used tool(s): Visual Studio version 2022.

//Description of the Dashboard: A dashboard windows form application which comprises of several other windows form applications which are: An IP Validator Application, A Simple Calculator Application, A Money Exchange & Temperature Convertion Application, and Lotto Max & Lotto 649 Application.

//Description of the IP Validator: An IPv4 and IPv6 address Validator which writes, reads and displays its valid address in a binary file followed by the current date and time.

//Description of the LottoMax: A LottoMax Windows form that has two buttons (which writes, reads, and display the random

//numbers in and from a text file) and two textboxes (one displaying random numbers from 1-50 and the other

//from 0-9).

//Description of the Lotto649: A LottoMax Windows form that has two buttons (which writes, reads, and display the random

//numbers in and from a text file) and two textboxes (one displaying random numbers from 1-50 and the other

//from 0-9).

//Name: Queen Sarah Anumu Bih

//Description of the Simple Calculator: A Simple Calculator form that has sevaral buttons which consists of numbers (0-9), a dot (for decimal),

//operator (+, -, \*, /), an equal to (=), a clear button, and an exit button. It performs all the operations with both integer numbers.

//and real numbers with the help of the class Calculator and displays them in the text boxes.

//Date: 07/18/2023

//Name: Queen Sarah Anumu Bih

//Description of the class Calculator: This is the class of the calculator having some private fields, a constructor, a property, and several methods which help

//the calculator to function properly.

//Date: 07/18/2023

//Name: Queen Sarah Anumu Bih

//Description of the Money Exchange: A money conversion application that converts money from one currency to several currencies.

//The app also tells the user how long he/she has spent on it before the user exits the application.

//Date: 07/25/2023

//Name: Queen Sarah Anumu Bih

//Description of the Temperature Convertor: A temperature conversion application that converts degree Celsius to Fahrenheit and Fahrenheit to degree Celsius.

//Date: 07/25/2023

1. **Present the print screens of yours forms and have a detailed description of the functionalities (step by step).**

**Dashboard**

A screenshot of a computer

Description automatically generated A screenshot of a calculator

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

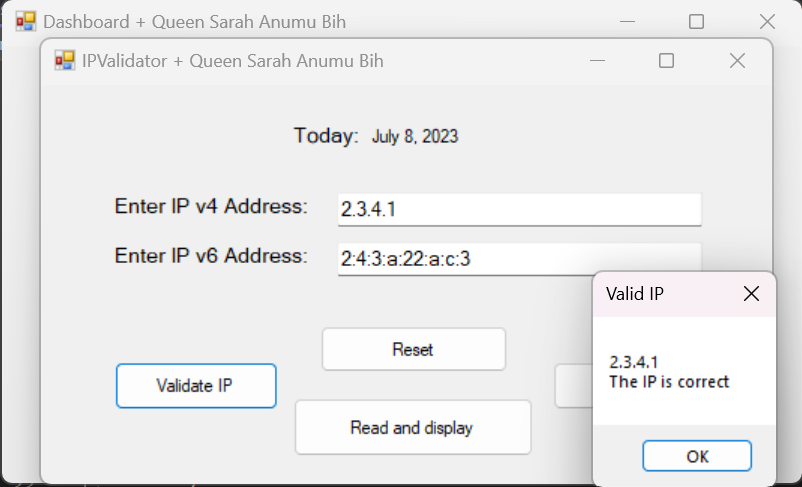
Description automatically generated

A screenshot of a computer

Description automatically generated

1. If you click on tab IP v4 & v6 validator, you will see a button having the image of an IP validator and when you click on it, it takes you to form 1 which is the IPv4 & IPv6 Validator Application.
2. If you click on tab Simple Calculator, you will see a button having the image of a simple calculator and when you click on it, it takes you to form 2 which is the Simple Calculator Application.
3. If you click on the tab Conversions, you will see two buttons in a group box, conversions having the image of Money Exchange and Temperature Convert which takes you to form 5, the Money Exchange Application and form 6, the Temperature Application when you click on them respectively.
4. If you click on the tab Generated Numbers, you will see two buttons in a group box, Lotto having the image of Lotto Max and Lotto 649 which takes you to form 3, the Lotto Max Application and form 4, the Lotto 649 Application when you click on them respectively.
5. If you click on the Exit button, the application asks you if you really want to exit.

**IP Validator**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated A screenshot of a computer error

Description automatically generated

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

A screenshot of a chat

Description automatically generated

1. The IP validator displays the current date at the top of the form.
2. The Validate button is used to validate and write just the valid IP addresses of IPv4 and IPv6 inputted in their text boxes. It writes their addresses in a binary file alongside the current date and time. Regular expression is used to validate the valid address and if the address is not valid, an error message is generated.
3. The Read and Display button is used to read the binary file written and displays it.
4. The Reset button is used to set back all the values in the text boxes to null allowing the user to re-enter other IP addresses.
5. The exit button is used to leave the form and ensure if the user is sure of quitting.

**Generated Numbers**

**LottoMax and Lotto649**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

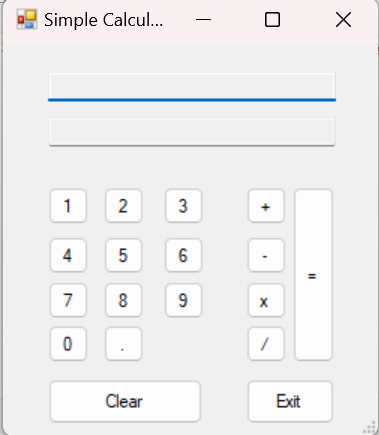
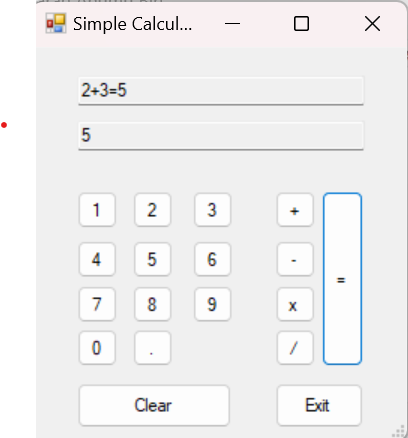
Description automatically generated**

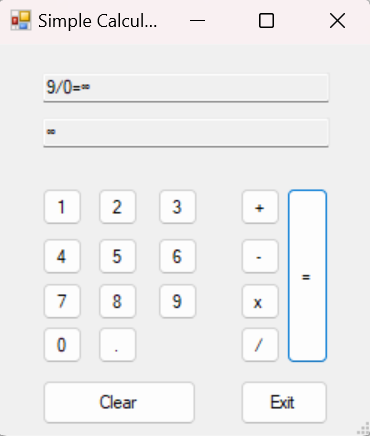
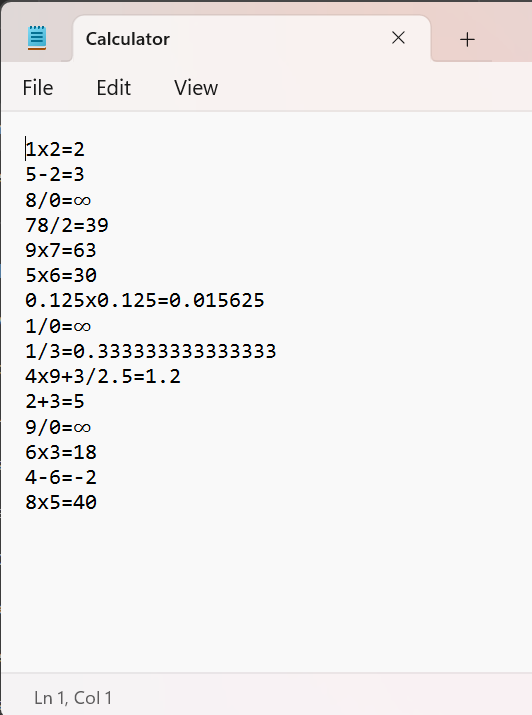
**A screenshot of a computer

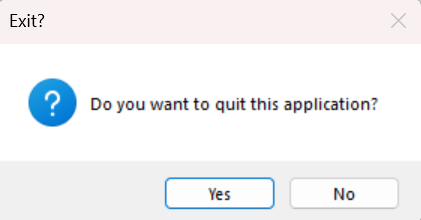
Description automatically generated**

1. The Lotto Max and Lotto 649 have two buttons whose functions are to display generated numbers and the other to save the generated numbers in a text file.
2. The display generate button in Lotto Max displays 7 plus one out of 1-50 random unique numbers in the vertical read only text box and at the same time displays 7 out of 0-9 random numbers. This button also writes these generated numbers in a text file as well as the current date and time.
3. The display generate button in Lotto 649 displays 6 plus one out of 1-49 random unique numbers in the vertical read only text box and at the same time displays 7 out of 0-9 random numbers. This button also writes these generated numbers in a text file as well as the current date and time.
4. The Read and Display button in both Lottos is used to read the written text file and display the generated numbers of both Lotto Max and Lotto 649. It also displays the current date and time and the bonus (plus one) of each generated unique number and the extra (0-9 random numbers).
5. The exit button is used to leave the applications.

**Simple Calculator**



1. The buttons in this calculator are to be pressed to display the operation(s) you want to carry out.
2. The first text box is where the full operation is displayed together with the answer and the second text box is where only the answer of the operation is displayed as shown in the image above.
3. The calculator also catches the exception of dividing a number by zero as shown in the image above.
4. The clear button is used to erase the operation recently carried out. The user can as well just start a new operation without necessarily clearing the old one. This is so because a flag is used in this Calculator to automatically erase the previous operation if you want to carry out another operation.
5. All operations carried out in this application are saved in a text file by name Calculator.txt; with each operation on a new line.
6. The exit button is used to leave the application.

**Conversions**

**Money Exchange**

**A screenshot of a computer

Description automatically generated**A screenshot of a computer

Description automatically generated

A screenshot of a phone

Description automatically generated

A screenshot of a computer

Description automatically generated

1. The convert from button is used to display the amount of the other various currencies when the user inputs an amount in the text box and chooses from which country the amount is in using the radio buttons. This button also writes all the converted currencies in a text file.
2. The read and display button reads all the conversions that are written into the text file and displays all of them in a message box as shown in the image above. It also displays the current date and time.
3. The exit button asks the user if he/she is sure of quitting the application and tells the user how long he/she has spent on the app.

**Temperature**

**A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

1. The radio buttons allow the user to choose from which temperature he/she wants to convert.
2. The button convert is used to display a short and specific description of the temperature in the message box when the user inputs a temperature in the text box.
3. The color of the temperatures also changes depending on its message description. The button convert also writes the temperatures imputed and converted in a text file.
4. The read file is used to read the temperature written in the text file and displays all of them in a message box as well as the current date and time with their specific descriptions.
5. The exit button allows the user to exit the application by first asking if he/she is sure of leaving.

**Note:** The second text box at the right is a read only textbox.

1. **Present the code of your application (forms).**

**Dashboard**

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;

//Name: Queen Sarah Anumu Bih

//Description: A dashboard windows form application which comprises of several other windows form applications which are:

//An IP Validator Application, A Simple Calculator Application, A Money Exchange & Temperature Convertion Application, and

//A Lotto Max & Lotto 649 Application.

//Date: 07/11/2023

namespace FinalProject\_2023

{

public partial class frm23Dashboard : Form

{

public frm23Dashboard()

{

InitializeComponent();

}

private void btn\_IP\_Validator\_Click(object sender, EventArgs e)

{

IP\_Validator obj = new IP\_Validator();

obj.ShowDialog();

}

private void btn\_Simple\_Calculator\_Click(object sender, EventArgs e)

{

Simple\_Calculator obj = new Simple\_Calculator();

obj.ShowDialog();

}

private void btn\_Lotto\_Max\_Click(object sender, EventArgs e)

{

LottoMax obj = new LottoMax();

obj.ShowDialog();

}

private void btn\_Lotto\_649\_Click\_1(object sender, EventArgs e)

{

Lotto649 obj = new Lotto649();

obj.ShowDialog();

}

private void btn\_Money\_Exchange\_Click(object sender, EventArgs e)

{

MoneyExchange obj = new MoneyExchange();

obj.ShowDialog();

}

private void btn\_Temperature\_Convert\_Click(object sender, EventArgs e)

{

Temperature obj = new Temperature();

obj.ShowDialog();

}

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

{

if (MessageBox.Show("Are you sure you want to exit this application?", "Exit?", MessageBoxButtons.YesNo, MessageBoxIcon.Question).ToString() == "Yes")

{

Application.Exit();

}

}

}

}

**IPv4 & IPv6 Validator**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Net;

using System.Text;

using System.Text.RegularExpressions;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

//Name: Queen Sarah Anumu Bih

//Description: An IPv4 and IPv6 address Validator which writes, reads and displays its valid address in a binary file

//followed by the current date and time.

//Date:07/11/2023

namespace FinalProject\_2023

{

public partial class IP\_Validator : Form

{

public IP\_Validator()

{

InitializeComponent();

}

Regex obj;

private void IP\_Validator\_Load(object sender, EventArgs e)

{

DateTime obj = DateTime.Today;

lblDate.Text = obj.ToLongDateString(); //.ToString();

if (!Directory.Exists(dirPath1))

Directory.CreateDirectory(dirPath1);

}

//string dir = @"C:\Files\";

static string dirPath1 = @"C:.\Files\";

// static string dirPath2 = @"C:..\Files\";

static string path = dirPath1 + "IPv4B.txt";

FileStream fs = null;

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

{

DateTime d = DateTime.Now;

obj = new Regex(@"^((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$");

if (obj.IsMatch(txtIPv4.Text))

{

try

{

fs = new FileStream(path, FileMode.Append, FileAccess.Write);

BinaryWriter bw = new BinaryWriter(fs);

bw.Write(txtIPv4.Text);

bw.Write(d.ToShortDateString() + " " + d.ToShortTimeString());

bw.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(path + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath1 + " not found.", "Directory Not Found");

}

catch (IOException ex)

{ MessageBox.Show(ex.Message, "IOException"); }

finally { if (fs != null) fs.Close(); }

MessageBox.Show($"{txtIPv4.Text} \n" +

$"The IP is correct", "Valid IP");

}

else

{

MessageBox.Show($"{txtIPv4.Text}\n"+

"The IP must have 4bytes\ninteger numbers between 0 to 255\n" +

"separated by a dot (255.255.255.255)", "Error");

txtIPv4.Focus();

}

Regex obj2 = new Regex(@"^(?:[0-9a-fA-F]{1,4}:){7}[0-9a-fA-F]{1,4}$");

if (obj2.IsMatch(txtIPv6.Text))

{

try

{

fs = new FileStream(path, FileMode.Append, FileAccess.Write);

BinaryWriter bw = new BinaryWriter(fs);

bw.Write(txtIPv6.Text);

bw.Write(d.ToShortDateString() + " " + d.ToShortTimeString());

bw.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(path + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath1 + " not found.", "Directory Not Found");

}

catch (IOException ex)

{ MessageBox.Show(ex.Message, "IOException"); }

finally { if (fs != null) fs.Close(); }

MessageBox.Show($"{txtIPv6.Text}\n" +

$"The IP is correct", "Valid IP");

}

else

{

MessageBox.Show($"{txtIPv6.Text}\n" +

"The IP must have 8bytes\ninteger numbers and letters (a-f or A-F)\n" +

"separated by a column (2001:0Db8:0000:0000:0000:ff00:0042:8329)", "Error");

txtIPv6.Focus();

}

}

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

{

string ipToPrint = "IP Address \t\t\t Date \n";

string ipAddress, date;

try

{

fs = new FileStream(path, FileMode.OpenOrCreate, FileAccess.Read);

BinaryReader br = new BinaryReader(fs);

while (br.PeekChar() != -1) //not finding the end of string

{

ipAddress = br.ReadString();

date = br.ReadString();

if(ipAddress.Length < 15)

{

ipToPrint += ipAddress + "\t\t\t\t" + date + "\n";

}

else

{

ipToPrint += ipAddress + "\t\t\t" + date + "\n" ;

}

}

MessageBox.Show(ipToPrint);

}

catch (IOException ex)

{

MessageBox.Show(ex.Message, "IOException");

}

finally

{

if (fs != null) fs.Close();

}

}

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

{

txtIPv4.ReadOnly = false;

txtIPv4.Text = null;

txtIPv4.Focus();

txtIPv6.Text = "";

}

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

{

if (MessageBox.Show("Do you want to quit?", "Exit?", MessageBoxButtons.YesNo, MessageBoxIcon.Question).ToString() == "Yes")

{

this.Close();

}

}

}

}

**LottoMax**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

//Name: Queen Sarah Anumu Bih

//Description: A LottoMax Windows form that has two buttons (which writes, reads and display the random

//numbers in and from a text file) and two textboxes (one displaying random numbers from 1-50 and the other

//from 0-9).

//Date:07/18/2023

namespace FinalProject\_2023

{

public partial class LottoMax : Form

{

public LottoMax()

{

InitializeComponent();

}

//string dir = @"C:\Files\";

static string dirPath = @"C:.\Files\";

string filePath = dirPath + "LottoNbrs.txt";

FileStream fs = null;

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

{

if (!Directory.Exists(dirPath))

Directory.CreateDirectory(dirPath);

}

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

{

string genNums = "";

int[] results = new int[8];

Random random = new Random();

int randomNumber;

for (int i = 0; i < results.Length; i++)

{

do

{

randomNumber = random.Next(1, 50);

}

while (results.Contains(randomNumber));

results[i] = randomNumber;

genNums = genNums + results[i].ToString() + "\t";

}

txtGeneratedN.Text = genNums;

string extraNums = "";

int[] extra = new int[7];

Random random2 = new Random();

int randomExtra;

for (int i = 0; i < extra.Length; i++)

{

randomExtra = random2.Next(0, 9);

extra[i] = randomExtra;

extraNums = extraNums + extra[i].ToString();

}

txtExtraMax.Text = extraNums;

try

{

fs = new FileStream(filePath, FileMode.Append, FileAccess.Write);

// create the output stream for a text file that exists

StreamWriter textOut = new StreamWriter(fs);

// write the fields into text file

textOut.Write("Max, " + DateTime.Now.ToString("yyyy/MM/dd hh:mm:ss tt") + ", ");

for (int i = 0; i < results.Length; i++)

{

if (i < 6)

{

textOut.Write(results[i] + ", ");

}

if (i == 6)

{

textOut.Write(results[i] + " ");

}

if (i == 7)

{

textOut.Write(" Bonus: " + results[i] + " ");

}

}

textOut.Write(" Extra: " + extraNums + "\n");

// close the output stream for the text file

textOut.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(filePath + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath + " not found.", "Directory Not Found");

}

catch (IOException ex)

{

MessageBox.Show(ex.Message, "IOException");

}

finally

{

if (fs != null) fs.Close();

}

}

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

{

try

{

fs = new FileStream(filePath, FileMode.Open, FileAccess.Read);

// create the object for the input stream for a text file

StreamReader textIn = new StreamReader(fs);

string textToPrint = "";

// read the data from the file and store it in the list

while (textIn.Peek() != -1)

{

string row = textIn.ReadLine().Trim();

textToPrint += row + "\n";

}

MessageBox.Show(textToPrint);

// close the input stream for the text file

textIn.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(filePath + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath + " not found.", "Directory Not Found");

}

catch (IOException ex)

{

MessageBox.Show(ex.Message, "IOException");

}

finally

{

if (fs != null) fs.Close();

}

}

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

{

if (MessageBox.Show("Do you want to quit this application?", "Exit?", MessageBoxButtons.YesNo, MessageBoxIcon.Question).ToString() == "Yes")

{

this.Close();

}

}

}

}

**Lotto649**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

//Name: Queen Sarah Anumu Bih

//Description: A Lotto649 Windows form that has two buttons (which writes, reads and display the random

//numbers in and from a text file) and two textboxes (one displaying random numbers from 1-49 and the other

//from 0-9).

//Date:07/18/2023

namespace FinalProject\_2023

{

public partial class Lotto649 : Form

{

public Lotto649()

{

InitializeComponent();

}

//string dir = @"C:\Files\";

static string dirPath = @"C:.\Files\";

string filePath = dirPath + "LottoNbrs.txt";

FileStream fs = null;

private void Lotto649\_Load(object sender, EventArgs e)

{

if (!Directory.Exists(dirPath))

Directory.CreateDirectory(dirPath);

}

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

{

string genNums = "";

int[] results = new int[7];

Random random = new Random();

int randomNumber;

for (int i = 0; i < results.Length; i++)

{

do

{

randomNumber = random.Next(1, 49);

}

while (results.Contains(randomNumber));

results[i] = randomNumber;

genNums = genNums + results[i].ToString() + "\t";

}

txtGeneratedN.Text = genNums;

string extraNums = "";

int[] extra = new int[7];

Random random2 = new Random();

int randomExtra;

for (int i = 0; i < extra.Length; i++)

{

randomExtra = random2.Next(0, 9);

extra[i] = randomExtra;

extraNums = extraNums + extra[i].ToString();

}

txtExtra.Text = extraNums;

try

{

fs = new FileStream(filePath, FileMode.Append, FileAccess.Write);

// create the output stream for a text file that exists

StreamWriter textOut = new StreamWriter(fs);

// write the fields into text file

textOut.Write("649, " + DateTime.Now.ToString("yyyy/MM/dd hh:mm:ss tt") + ", ");

for (int i = 0; i < results.Length; i++)

{

if (i < 5)

{

textOut.Write(results[i] + ", ");

}

if (i == 5)

{

textOut.Write(results[i] + " ");

}

if (i == 6)

{

textOut.Write(" Bonus: " + results[i] + " ");

}

}

textOut.Write(" Extra: " + extraNums + "\n");

// close the output stream for the text file

textOut.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(filePath + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath + " not found.", "Directory Not Found");

}

catch (IOException ex)

{

MessageBox.Show(ex.Message, "IOException");

}

finally

{

if (fs != null) fs.Close();

}

}

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

{

try

{

fs = new FileStream(filePath, FileMode.Open, FileAccess.Read);

// create the object for the input stream for a text file

StreamReader textIn = new StreamReader(fs);

string textToPrint = "";

// read the data from the file and store it in the list

while (textIn.Peek() != -1)

{

string row = textIn.ReadLine().Trim();

textToPrint += row + "\n";

}

MessageBox.Show(textToPrint);

// close the input stream for the text file

textIn.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(filePath + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath + " not found.", "Directory Not Found");

}

catch (IOException ex)

{

MessageBox.Show(ex.Message, "IOException");

}

finally

{

if (fs != null) fs.Close();

}

}

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

{

if (MessageBox.Show("Do you want to quit this application?", "Exit?", MessageBoxButtons.YesNo, MessageBoxIcon.Question).ToString() == "Yes")

{

this.Close();

}

}

}

}

**Simple Calculator**

sing System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

//Name: Queen Sarah Anumu Bih

//Description: A Simple Calculator form that has sevaral buttons which consists of numbers(0-9), a dot(for decimal),

//operator (+, -, \*, /), an equals to(=), a clear button, and an exit button. It performs all the operations with both integer numbers

//and real numbers with the help of the class Calculator and displays them in the text boxes.

//Date: 07/18/2023

namespace FinalProject\_2023

{

public partial class Simple\_Calculator : Form

{

private Calculator calculator;

private string currentInput;

public Simple\_Calculator()

{

InitializeComponent();

calculator = new Calculator();

currentInput = string.Empty; //originally currentValue. Used that name in order not to get confused with that in class calc

}

static string dirPath = @"C:.\Files\";

string filePath = dirPath + "Calculator.txt";

FileStream fs = null;

private void Simple\_Calculator\_Load(object sender, EventArgs e)

{

if (!Directory.Exists(dirPath))

Directory.CreateDirectory(dirPath);

}

bool isCreatedCal = false; //flag

private void btn1\_Click(object sender, EventArgs e)

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btn1.Text;

txtCalculate.Text += btn1.Text;

}

else

{

currentInput += btn1.Text;

txtCalculate.Text += btn1.Text;

}

}

private void btn2\_Click(object sender, EventArgs e)

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btn2.Text;

txtCalculate.Text += btn2.Text;

}

else

{

currentInput += btn2.Text;

txtCalculate.Text += btn2.Text;

}

}

private void btn3\_Click(object sender, EventArgs e)

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btn3.Text;

txtCalculate.Text += btn3.Text;

}

else

{

currentInput += btn3.Text;

txtCalculate.Text += btn3.Text;

}

}

private void btn4\_Click(object sender, EventArgs e)

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btn4.Text;

txtCalculate.Text += btn4.Text;

}

else

{

currentInput += btn4.Text;

txtCalculate.Text += btn4.Text;

}

}

private void btn5\_Click(object sender, EventArgs e)

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btn5.Text;

txtCalculate.Text += btn5.Text;

}

else

{

currentInput += btn5.Text;

txtCalculate.Text += btn5.Text;

}

}

private void btn6\_Click(object sender, EventArgs e)

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btn6.Text;

txtCalculate.Text += btn6.Text;

}

else

{

currentInput += btn6.Text;

txtCalculate.Text += btn6.Text;

}

}

private void btn7\_Click(object sender, EventArgs e)

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btn7.Text;

txtCalculate.Text += btn7.Text;

}

else

{

currentInput += btn7.Text;

txtCalculate.Text += btn7.Text;

}

}

private void btn8\_Click(object sender, EventArgs e)

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btn8.Text;

txtCalculate.Text += btn8.Text;

}

else

{

currentInput += btn8.Text;

txtCalculate.Text += btn8.Text;

}

}

private void btn9\_Click(object sender, EventArgs e)

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btn9.Text;

txtCalculate.Text += btn9.Text;

}

else

{

currentInput += btn9.Text;

txtCalculate.Text += btn9.Text;

}

}

private void btn0\_Click(object sender, EventArgs e)

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btn0.Text;

txtCalculate.Text += btn0.Text;

}

else

{

currentInput += btn0.Text;

txtCalculate.Text += btn0.Text;

}

}

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

{

if (isCreatedCal)

{

txtCalculate.Text = "";

txtAnswer.Text = "";

isCreatedCal = false;

calculator.Clear();

currentInput += btnDecimal.Text;

txtCalculate.Text += btnDecimal.Text;

}

else

{

currentInput += btnDecimal.Text;

txtCalculate.Text += btnDecimal.Text;

}

}

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

{

string operation = btnAdd.Text;

if (!string.IsNullOrEmpty(currentInput))

{

calculator.Add(double.Parse(currentInput));

calculator.SetOperator(operation);

currentInput = string.Empty;

txtCalculate.Text += operation;

}

}

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

{

string operation = btnSubtract.Text;

if (!string.IsNullOrEmpty(currentInput))

{

calculator.Subtract(double.Parse(currentInput));

calculator.SetOperator(operation);

currentInput = string.Empty;

txtCalculate.Text += operation;

}

}

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

{

string operation = btnMultiply.Text;

if (!string.IsNullOrEmpty(currentInput))

{

calculator.Multiply(double.Parse(currentInput));

calculator.SetOperator(operation);

currentInput = string.Empty;

txtCalculate.Text += operation;

}

}

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

{

string operation = btnDivide.Text;

if (!string.IsNullOrEmpty(currentInput))

{

calculator.Divide(double.Parse(currentInput));

calculator.SetOperator(operation);

currentInput = string.Empty;

txtCalculate.Text += operation;

}

}

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

{

Button button = (Button)sender;

string operation = button.Text;

txtCalculate.Text += btnEquals.Text;

if (!string.IsNullOrEmpty(currentInput))

{

calculator.Equals(double.Parse(currentInput));

currentInput= string.Empty;

txtCalculate.Text += calculator.Value.ToString();

txtAnswer.Text = calculator.Value.ToString();

//changes the condition of flag after clicking button '='

isCreatedCal = true;

}

//used else to handle event: after pressing '=', stop appending text in textfields

else

{

txtCalculate.Text = "";

txtAnswer.Text= "";

}

try

{

fs = new FileStream(filePath, FileMode.Append, FileAccess.Write);

// create the output stream for a text file that exists

StreamWriter textOut = new StreamWriter(fs);

// write the fields into text file

textOut.Write(txtCalculate.Text + "\n");

// close the output stream for the text file

textOut.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(filePath + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath + " not found.", "Directory Not Found");

}

catch (IOException ex)

{

MessageBox.Show(ex.Message, "IOException");

}

finally

{

if (fs != null) fs.Close();

}

}

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

{

calculator.Clear();

currentInput = string.Empty;

txtCalculate.Text = string.Empty;

txtAnswer.Text = string.Empty;

}

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

{

if (MessageBox.Show("Do you want to quit this application?", "Exit?", MessageBoxButtons.YesNo, MessageBoxIcon.Question).ToString() == "Yes")

{

this.Close();

}

}

}

}

**Class Calculator**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.Cryptography.X509Certificates;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

//Name: Queen Sarah Anumu Bih

//Description: This is the class of the calculator having some private fields, a constructor, a property and several methods which help

//the calculator to function properly.

//Date: 07/18/2023

namespace FinalProject\_2023

{

internal class Calculator

{

private double currentValue;

private double operand1;

private double operand2;

private string op;

public double Value { set { currentValue = value; } get { return currentValue; } }

public Calculator()

{

currentValue = 0;

operand1 = 0;

operand2 = 0;

op = null;

}

public void Add(double displayValue)

{

operand1 = displayValue;

currentValue = displayValue;

op = "+";

}

public void Subtract(double displayValue)

{

operand1 = displayValue;

currentValue = displayValue;

op = "-";

}

public void Multiply(double displayValue)

{

operand1 = displayValue;

currentValue = displayValue;

op = "\*";

}

public void Divide(double displayValue)

{

operand1 = displayValue;

currentValue = displayValue;

op = "/";

}

public void SetOperator (string operation)

{

op = operation;

}

public void Equals()

{

switch (op)

{

case "+":

currentValue = operand1 + operand2;

break;

case "-":

currentValue = operand1 - operand2;

break;

case "x":

currentValue = operand1 \* operand2;

break;

case "/":

if (operand2 != 0)

{

try

{

currentValue = operand1 / operand2;

}

catch(Exception ex1)

{

MessageBox.Show(ex1.Message);

}

}

else

{

currentValue = operand1 / operand2;

}

break;

}

operand1 = currentValue;

operand2 = 0;

op = null;

}

public void Equals(double displayValue)

{

operand2 += displayValue;

Equals();

}

public void Clear()

{

currentValue = 0;

operand1 = 0;

operand2 = 0;

op = null;

}

}

}

**Money Exchange**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.Button;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

//Name: Queen Sarah Anumu Bih

//Description: A money conversion application that converts money from one currency to several currencies.

//The app also tells the user how long he/she has spent on it before the user exits the application.

//Date: 07/25/2023

namespace FinalProject\_2023

{

public partial class MoneyExchange : Form

{

public MoneyExchange()

{

InitializeComponent();

}

static string dirPath = @"C:.\Files\";

string filePath = dirPath + "MoneyConversions.txt";

FileStream fs = null;

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

{

if (!Directory.Exists(dirPath))

Directory.CreateDirectory(dirPath);

startTime = DateTime.Now;

}

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

{

try

{

double inputValue = double.Parse(txtAmount.Text);

double cadToUsd = 0.7353;

double cadToEur = 0.6757;

double cadToGbp = 0.5952;

double cadToCfa = 441.74;

double cadToInr = 62.32;

double usdToCad = 1.32;

double usdToEur = 0.89;

double usdToGbp = 0.77;

double usdToCfa = 581.50;

double usdToInr = 82.07;

double eurToCad = 1.48;

double eurToUsd = 1.12;

double eurToGbp = 0.87;

double eurToCfa = 651.43;

double eurToInr = 91.92;

double gbpToCad = 1.70;

double gbpToUsd = 1.29;

double gbpToEur = 1.67;

double gbpToCfa = 752.23;

double gbpToInr = 106.18;

double cfaToCad = 0.0023;

double cfaToUsd = 0.00171038;

double cfaToEur = 0.0015;

double cfaToGbp = 0.0013;

double cfaToInr = 0.14;

double inrToCad = 0.016;

double inrToUsd = 0.012;

double inrToEur = 0.011;

double inrToGbp = 0.0094;

double inrToCfa = 7.09;

double usd, cad, eur, gbp, cfa, inr;

string labelValue;

if (radioButtonCAD.Checked)

{

labelValue = radioButtonCAD.Text;

cad = inputValue;

usd = cad \* cadToUsd;

eur = cad \* cadToEur;

gbp = cad \* cadToGbp;

cfa = cad \* cadToCfa;

inr = cad \* cadToInr;

}

else if (radioButtonUSD.Checked)

{

labelValue = radioButtonUSD.Text;

usd = inputValue;

cad = usd \* usdToCad;

eur = usd \* usdToEur;

gbp = usd \* usdToGbp;

cfa = usd \* usdToCfa;

inr = usd \* usdToInr;

}

else if (radioButtonEUR.Checked)

{

labelValue = radioButtonEUR.Text;

eur = inputValue;

cad = eur \* eurToCad;

usd = eur \* eurToUsd;

gbp = eur \* eurToGbp;

cfa = eur \* eurToCfa;

inr = eur \* eurToInr;

}

else if (radioButtonGBP.Checked)

{

labelValue = radioButtonGBP.Text;

gbp = inputValue;

cad = gbp \* gbpToCad;

usd = gbp \* gbpToUsd;

eur = gbp \* gbpToEur;

cfa = gbp \* gbpToCfa;

inr = gbp \* gbpToInr;

}

else if (radioButtonCFA.Checked)

{

labelValue = radioButtonCFA.Text;

cfa = inputValue;

cad = cfa \* cfaToCad;

usd = cfa \* cfaToUsd;

eur = cfa \* cfaToEur;

gbp = cfa \* cfaToGbp;

inr = cfa \* cfaToInr;

}

else // radioButtonINR.Checked

{

labelValue = radioButtonINR.Text;

inr = inputValue;

cad = inr \* inrToCad;

usd = inr \* inrToUsd;

eur = inr \* inrToEur;

gbp = inr \* inrToGbp;

cfa = inr \* inrToCfa;

}

txtCad.Text = cad.ToString();

txtUsd.Text = usd.ToString();

txtEur.Text = eur.ToString();

txtGbp.Text = gbp.ToString();

txtCfa.Text = cfa.ToString();

txtInr.Text = inr.ToString();

try

{

fs = new FileStream(filePath, FileMode.Append, FileAccess.Write);

StreamWriter textOut = new StreamWriter(fs);

DateTime currentDate = DateTime.Now;

string formattedDate = currentDate.ToString("dd MM yyyy");

string time = currentDate.ToString("HH:mm:ss tt");

string[] dateFile = formattedDate.Split(' ');

string day = dateFile[0];

string month = dateFile[1];

string year = dateFile[2];

textOut.Write(year + "/" + month + "/" + day + " " + time + "\n" + inputValue + " " +labelValue +

" = " + cad + " CAD; " + usd + " USD; " + eur + " EUR; " + gbp + " GBP; " + cfa + " CFA; " + inr + " INR\n");

textOut.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(filePath + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath + " not found.", "Directory Not Found");

}

catch (IOException ex)

{ MessageBox.Show(ex.Message, "IOException"); }

finally

{ if (fs != null) fs.Close(); }

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Conversion Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

try

{

fs = new FileStream(filePath, FileMode.Open, FileAccess.Read);

StreamReader textIn = new StreamReader(fs);

string textToPrint = textIn.ReadToEnd();

MessageBox.Show(textToPrint + "\n");

textIn.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(filePath + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath + " not found.", "Directory Not Found");

}

catch (IOException ex)

{ MessageBox.Show(ex.Message, "IOException"); }

finally

{ if (fs != null) fs.Close(); }

}

DateTime startTime;

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

{

DateTime startTime2 = DateTime.Now;

DateTime startTime3 = DateTime.Now;

TimeSpan elapsedTime = startTime2 - startTime;

TimeSpan elapsedTime2 = startTime3 - startTime;

string lblElapsedTime = elapsedTime.ToString(@"mm");

string lblElapsedTime2 = elapsedTime2.ToString(@"ss");

if (MessageBox.Show("Do you want to quit this app? You have been here for " + lblElapsedTime + " minutes " +

lblElapsedTime2 + " sec", "Exit?", MessageBoxButtons.YesNo, MessageBoxIcon.Question).ToString() == "Yes")

{

this.Close();

}

}

}

}

**Temperature**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.Button;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

//Name: Queen Sarah Anumu Bih

//Description: A temperature conversion application that converts degree celsius to Fahrenheit and Fahrenheit to degree celsius.

//Date: 07/25/2023

namespace FinalProject\_2023

{

public partial class Temperature : Form

{

public Temperature()

{

InitializeComponent();

}

static string dirPath = @"C:.\Files\";

string filePath = dirPath + "TempConversions.txt";

FileStream fs = null;

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

{

if (!Directory.Exists(dirPath))

Directory.CreateDirectory(dirPath);

}

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

{

lblC.Text = "C";

lblF.Text = "F";

}

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

{

lblC.Text = "F";

lblF.Text = "C";

}

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

{

double celsius = 0;

double farenhait = 0;

if (radioButtonCtoF.Checked)

{

celsius = double.Parse(txtConvert.Text);

if (celsius == 100)

{

txtConvert.ForeColor = Color.Red;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

farenhait = 212;

txtResult.Text = farenhait.ToString();

txtResult.ForeColor = Color.Red;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Water boils";

}

else if (celsius == 40)

{

txtConvert.ForeColor = Color.Red;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

farenhait = 104;

txtResult.Text = farenhait.ToString();

txtResult.ForeColor = Color.Red;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Hot Bath";

}

else if (celsius == 37)

{

txtConvert.ForeColor = Color.Orange;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

farenhait = 98.6;

txtResult.Text = farenhait.ToString();

txtResult.ForeColor = Color.Orange;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Body temprature";

}

else if (celsius == 30)

{

txtConvert.ForeColor = Color.Orange;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

farenhait = 86;

txtResult.Text = farenhait.ToString();

txtResult.ForeColor = Color.Orange;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Beach weather";

}

else if (celsius == 21)

{

txtConvert.ForeColor = Color.Green;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Regular);

farenhait = 70;

txtResult.Text = farenhait.ToString();

txtResult.ForeColor = Color.Green;

txtResult.Font = new Font(txtResult.Font, FontStyle.Regular);

txtMessage.Text = "Room temperature";

}

else if (celsius == 10)

{

txtConvert.ForeColor = Color.Blue;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

farenhait = 50;

txtResult.Text = farenhait.ToString();

txtResult.ForeColor = Color.Blue;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Cool Day";

}

else if (celsius == 0)

{

txtConvert.ForeColor = Color.Blue;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

farenhait = 32;

txtResult.Text = farenhait.ToString();

txtResult.ForeColor = Color.Blue;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Freezing Point of Water";

}

else if (celsius == -18)

{

txtConvert.ForeColor = Color.Black;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Regular);

farenhait = 0;

txtResult.Text = farenhait.ToString();

txtResult.ForeColor = Color.Black;

txtResult.Font = new Font(txtResult.Font, FontStyle.Regular);

txtMessage.Text = "Very Cold Day";

}

else if (celsius == -40)

{

txtConvert.ForeColor = Color.Black;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

farenhait = -40;

txtResult.Text = farenhait.ToString();

txtResult.ForeColor = Color.Black;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Extremely Cold Day";

}

else

{

txtConvert.ForeColor = Color.Black;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Regular);

farenhait = (celsius \* 9 / 5) + 32;

txtResult.Text = farenhait.ToString();

txtResult.ForeColor = Color.Black;

txtResult.Font = new Font(txtResult.Font, FontStyle.Regular);

txtMessage.Text = "No specific description.";

}

}

if (radioButtonFtoC.Checked)

{

farenhait = double.Parse(txtConvert.Text);

if (farenhait == 212)

{

txtConvert.ForeColor = Color.Red;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

celsius = 100;

txtResult.Text = celsius.ToString();

txtResult.ForeColor = Color.Red;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Water boils";

}

else if (farenhait == 104)

{

txtConvert.ForeColor = Color.Red;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

celsius = 40;

txtResult.Text = celsius.ToString();

txtResult.ForeColor = Color.Red;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Hot Bath";

}

else if (farenhait == 98.6)

{

txtConvert.ForeColor = Color.Orange;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

celsius = 37;

txtResult.Text = celsius.ToString();

txtResult.ForeColor = Color.Orange;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Body temprature";

}

else if (farenhait == 86)

{

txtConvert.ForeColor = Color.Orange;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

celsius = 30;

txtResult.Text = celsius.ToString();

txtResult.ForeColor = Color.Orange;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Beach weather";

}

else if (farenhait == 70)

{

txtConvert.ForeColor = Color.Green;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Regular);

celsius = 21;

txtResult.Text = celsius.ToString();

txtResult.ForeColor = Color.Green;

txtResult.Font = new Font(txtResult.Font, FontStyle.Regular);

txtMessage.Text = "Room temperature";

}

else if (farenhait == 50)

{

txtConvert.ForeColor = Color.Blue;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

celsius = 10;

txtResult.Text = celsius.ToString();

txtResult.ForeColor = Color.Blue;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Cool Day";

}

else if (farenhait == 32)

{

txtConvert.ForeColor = Color.Blue;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

celsius = 0;

txtResult.Text = celsius.ToString();

txtResult.ForeColor = Color.Blue;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Freezing Point of Water";

}

else if (farenhait == 0)

{

txtConvert.ForeColor = Color.Black;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Regular);

celsius = -18;

txtResult.Text = celsius.ToString();

txtResult.ForeColor = Color.Black;

txtResult.Font = new Font(txtResult.Font, FontStyle.Regular);

txtMessage.Text = "Very Cold Day";

}

else if (farenhait == -40)

{

txtConvert.ForeColor = Color.Black;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Bold);

celsius = -40;

txtResult.Text = celsius.ToString();

txtResult.ForeColor = Color.Black;

txtResult.Font = new Font(txtResult.Font, FontStyle.Bold);

txtMessage.Text = "Extremely Cold Day";

}

else

{

txtConvert.ForeColor = Color.Black;

txtConvert.Font = new Font(txtConvert.Font, FontStyle.Regular);

celsius = (farenhait - 32) \* 5 / 9;

txtResult.Text = celsius.ToString();

txtResult.ForeColor = Color.Black;

txtResult.Font = new Font(txtResult.Font, FontStyle.Regular);

txtMessage.Text = "No specific description.";

}

}

try

{

fs = new FileStream(filePath, FileMode.Append, FileAccess.Write);

StreamWriter textOut = new StreamWriter(fs);

DateTime currentDate = DateTime.Now;

string formattedDate = currentDate.ToString("dd MM yyyy");

string time = currentDate.ToString("HH:mm:ss tt");

string[] dateFile = formattedDate.Split(' ');

string day = dateFile[0];

string month = dateFile[1];

string year = dateFile[2];

if (radioButtonCtoF.Checked)

{

textOut.WriteLine(celsius.ToString() + " C = " + farenhait.ToString() + " F, " + year + "/" + month + "/" + day + " " + time + " " + txtMessage.Text, "\n");

}

else

{

textOut.WriteLine(farenhait.ToString() + " F = " + celsius.ToString() + " C, " + year + "/" + month + "/" + day + " " + time + " " + txtMessage.Text, "\n");

}

textOut.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(filePath + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath + " not found.", "Directory Not Found");

}

catch (IOException ex)

{ MessageBox.Show(ex.Message, "IOException"); }

finally

{ if (fs != null) fs.Close(); }

}

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

{

try

{

fs = new FileStream(filePath, FileMode.Open, FileAccess.Read);

StreamReader textIn = new StreamReader(fs);

string textToPrint = textIn.ReadToEnd();

MessageBox.Show(textToPrint + "\n");

textIn.Close();

}

catch (FileNotFoundException)

{

MessageBox.Show(filePath + " not found.", "File Not Found");

}

catch (DirectoryNotFoundException)

{

MessageBox.Show(dirPath + " not found.", "Directory Not Found");

}

catch (IOException ex)

{ MessageBox.Show(ex.Message, "IOException"); }

finally

{ if (fs != null) fs.Close(); }

}

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

{

if (MessageBox.Show("Do you want to quit this application?", "Exit?", MessageBoxButtons.YesNo, MessageBoxIcon.Question).ToString() == "Yes")

{

this.Close();

}

}

}

}

1. **Present the classes and/or methods that you create, or you did use in the project.**

|  |  |
| --- | --- |
| **Class/Method Name** | **Description** |
| 1. **Class Calc** 2. **Private Fields**   currentValue  operand1  operand2  op   1. **Property**   Value   1. **Constructor**   Calculator()   1. **Methods of Calc**   Add(displayValue)  Subtract(displayValue)  Multiply(displayValue)  Divide(displayValue)  SetOperator(operation)  Equals()  Equals(displayValue)  Clear() | It is designed to simulate a simple calculator.  Holds the current value displayed on the calculator.  Stores the first operand of the operation.  Stores the second operand of the operation.  Stores the operator symbols (+, -, \*, /) for the arithmetic operation.  Allows getting and setting the currentValue, providing access to the currentValue (currentInput) displayed on the calculator.  It initializes all the fields to their default values (zero for numeric fields and null for the op, string field).  Used to handle the addition operation.  Used to handle the subtraction operation.  Used to handle the multiplication operation.  Used to handle the division operation.  Used to set the operator (op) directly.  Used to perform the arithmetic operation based on the operator (op).  Used to handle chained arithmetic operations. It updates the second operand with the value of displayValue and then calls the Equals () method to calculate the result.  Used to reset all the fields to their default values, effectively clearing the calculator. |
| 1. **Global Classes**   Regex  Directory  FileStream  BinaryReader  BinaryWriter  StreamWriter  StreamReader  Random  String  MessageBox | Short description of the classes:  Regular expressions, checking if a given input matches with the given pattern.  Used to manipulate the directory structure.  Used for reading and writing text and binary files.  Used to handle binary data.  Used to write binary information into stream.  Used to write to string builder class.  Used to read data written by a StringWriter class.  Used for generating random numbers based on a seed value.  Represents a sequence of characters.  It represents a message to users. |
| 1. **Global Methods**   ReadLine()  WriteLine()  ToString()  isMatch()  Write()  Show()  Close()  Next()  ToShortDateString()  ToShortTimeString()  ReadString()  IsNullOrEmpty()  String.Empty  Exists()  CreateDirectory()  PeekChar()  Peek()  Contains()  Focus() | Short description of the methods:  Reads the complete string until the user presses the enter key or a new line character is found.  Prints data along with printing new line.  Converts an object to its string representation for suitable display.  Used to test whether a text string matches a pattern of ordinary numbers or regular expressions.  Used to print data without printing a new line.  It displays a message box in a small window in the center of the form.  Used to close the form on some buttons click event.  Returns a non-negative(positive) random integer.  Converts the value of the current DateTime object to its equivalent short date string representation.  Converts the value of the current DateTime object to its equivalent short time string representation.  Used to read a string from the current stream.  Used to indicate whether a specified string is null or empty.  Initializes a string variable to an empty string.  Used to determine whether the specified file exist or not.  Creates all directories and subdirectories in the specified path with the specified permissions unless they already exist.  Returns the next available binary character.  Returns the object at the top of the stack without removing it.  Used to return a value indicating whether the specified substring occurs within this string or not.  Gives the specified element keyboard focus. |
|  |  |
|  |  |

1. **Present the difficulties that you have, what was the hardest and the easiest part of your project.**

The easiest part of the project was to design the application. The hardest part for me was the calculator because it had a lot of work and to create the flag for new calculations to overwrite the old one. But in the end everything works fine.