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

|  |
| --- |
| Gabriel Paz Paiva  7/19/2021 |

1. **Start by adding a short description of your project, and the languages (technologies) used:**

This project’s objective was to create a Windows Form with several applications and functionalities linked each other by a main dashboard to help the user to do some operations and have a log of them. The operations are validation of IPV4, a calculator tool, temperature converter, money exchange converter, and two lottery number generators.

1. Language C#.
2. tool Visual Studio version 2019.
3. **Present the print screens of yours forms, and have a detailed description of the functionalities (step by step).**

Dashboard

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo, Word

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

1. The dashboard is divided in 4 areas, allowing the user to transit between them by clicking.
2. After selecting one of the tools, the system creates an instance of the selected functionality.
3. The system allows the user to create many instances of the selected functionality.
4. If the user clicks on the Exit button, a new message box will pop up asking if the application should be closed. All instances are closed if the user elects “yes”, otherwise nothing happens.

LottoMax

Interface gráfica do usuário, Aplicativo, Word

Descrição gerada automaticamenteInterface gráfica do usuário, Texto

Descrição gerada automaticamenteInterface gráfica do usuário, Texto

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

1. After selecting the LottoMax button, the user has 3 options.
2. If the user clicks on Read before writing anything before, a message warning about it will pop up.
3. If the user clicks at “Generate” button, the program will generate 8 unique numbers between them and will record it in a file with the name of the lottery, date, and time.
4. If the user clicks at “Read File” button, the program will read the file created.
5. If the user clicks on the Exit button, a new message box will pop up asking if the instance should be closed. Only the current instance is closed if the user elects “yes”, otherwise nothing happens.

Lotto649

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo, Word

Descrição gerada automaticamenteInterface gráfica do usuário, Texto

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

1. After selecting the LottoMax button, the user has 3 options.
2. If the user clicks on Read before writing anything before, a message warning about it will pop up.
3. If the user clicks at “Generate” button, the program will generate 7 unique numbers between them and will record it in a file with the name of the lottery, date, and time.
4. If the user clicks at “Read File” button, the program will read the file created. 649Lotto and LottoMax shares the same file.
5. If the user clicks on the Exit button, a new message box will pop up asking if the instance should be closed. Only the current instance is closed if the user elects “yes”, otherwise nothing happens.

Money Exchange

Interface gráfica do usuário

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

1. After selecting the Money Exchange button, a new instance of Money Exchange will open.
2. If the user clicks on Read before writing anything before, a message warning about it will pop up.
3. The user should select one money currency “From” and “To” from the radio buttons. By default, one country of the “To” is selected when the form load.
4. If the user tries to convert anything before selecting a country for “To”, then a message warning about the issue is pop up. This will also happen if the user tries to input some invalid character in the input textbox, such as a letter.
5. If the user clicks “Convert” and all the data is valid, then the program will perform the conversion by comparing the strings of the checked radio buttons and a dictionary, and register it in a file with the current date, and time.
6. By clicking “Read File”, the program will display the information inside the file mentioned before.
7. If the user clicks on the Exit button, a new message box will pop up asking if the instance should be closed. Only the current instance is closed if the user elects “yes”, otherwise nothing happens.

Temp App

Interface gráfica do usuário

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Texto

Descrição gerada automaticamente com confiança médiaInterface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

1. After selecting the Temperature Convert button, a new instance of Temp App will open.
2. If the user clicks on Read before writing anything before, a message warning about it will pop up.
3. By default, the first radio button is checked. If the user changes it, then the labels under the text boxes will also change their places.
4. If the user tries to convert an invalid data, the program will display a message notifying about the issue.
5. If the user tries to convert a valid data, then the program will convert the number and check if the result matches with one of the specific conditions. If it matches, then the program will display the message at the read only text box under the label “Message”, otherwise it will not display anything. In addition, the operation is registered in a specific file with the current date and time.
6. By clicking “Read File”, the program will display the information inside the file mentioned before.
7. If the user clicks on the Exit button, a new message box will pop up asking if the instance should be closed. Only the current instance is closed if the user elects “yes”, otherwise nothing happens.

Calculator

Tela de computador com fundo branco

Descrição gerada automaticamente com confiança médiaInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário

Descrição gerada automaticamenteInterface gráfica do usuário

Descrição gerada automaticamente

1. After selecting the calculator image button, a new instance of the calculator will open.
2. The user has an option to select the decimal separator as a point or comma. This will solve any problems with operations done in different countries formats.
3. If the user tries to convert an invalid data, the program will display a message notifying about the issue.
4. To use the calculator the user should insert a number, either by typing the value or pressing the buttons, choose an operation, select a second value, and then press equal to see the result.
5. For each operation selected the system stored the value and the operation in variables.
6. The user has the option to reuse the last value calculated and use it as the first number inserted.
7. For each number inserted, the calculator will clean the display.
8. If the user wants to clear all values and operations stored inside the program, the user should press “Clear” button.
9. If the user clicks on the Exit button, a new message box will pop up asking if the instance should be closed. Only the current instance is closed if the user elects “yes”, otherwise nothing happens.

IP4Validator

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo, Word

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo, Teams

Descrição gerada automaticamente

1. After selecting the IP image button, a new instance of the IP Validator will open.
2. The program will get the current date and write inside the label when the forms load.
3. If the user wants to check and IP address format, then the user should insert in the text box the IP address to be validated and then press “Validate IP” button.
4. The system will break the string inserted in 4 parts and verifies its content, and then it will display the result.
5. If the user tries to convert an invalid data, the program will display a message notifying about the issue.
6. If the user wants to clean the textbox, then the user should press “Reset” button.
7. If the user clicks on the Exit button, a new message box will pop up asking if the instance should be closed. Only the current instance is closed if the user elects “yes”, otherwise nothing happens.
8. **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;

/\* Student: Gabriel Paz Paiva

\* Description: Oriented Object Programming Project - Summer 2021

\* Starting Date: 24 June 2021.

\*Version 1.0 - 29/06/2021 - Layout creation, Temp App, and IP application.

Version 2.0 - 08/07/2021 - MoneyConversion created.

Version 3.0 - 16/07/2021 - Started to Create the LottoMax layout, and finished IP, Money Exchange and Temp apps.

Version 4.0 - 17/07/2024 - Finished LottoMax and Lott649, add ReadFiles functions, added some exeptions clauses, and revied everything.\*/

namespace WFOOPProject

{

public partial class DashboardForm : Form

{

public DashboardForm()

{

InitializeComponent();

}

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

{

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

{

Application.Exit();

}

}

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

{

SimpleCalc Calculator = new SimpleCalc();

Calculator.Show();

}

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

{

IP4Validator IpVal = new IP4Validator();

IpVal.Show();

}

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

{

TempApp TempInst = new TempApp();

TempInst.Show();

}

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

{

MoneyConversion MoneyInst = new MoneyConversion();

MoneyInst.Show();

}

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

{

LottoMax LottoMaxInst = new LottoMax();

LottoMaxInst.Show();

}

private void btnLotto649\_Click(object sender, EventArgs e)

{

Lotto649 Lotto649Inst = new Lotto649();

Lotto649Inst.Show();

}

}

}

IP4Validator

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 WFOOPProject

{

public partial class IP4Validator : Form

{

public IP4Validator()

{

InitializeComponent();

}

private void IP4Validator\_Load(object sender, EventArgs e)

{

labelToday.Text = "Today: "+ DateTime.Now.ToLongDateString();

}

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

{

if (MessageBox.Show("Do you want to quit the application IP4 Validator?", "Exit ?", MessageBoxButtons.YesNo).ToString() == "Yes")

{

this.Close();

}

}

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

{

string[] arIptriad;

byte temp;

try

{

arIptriad = txtIpAddress.Text.Trim().Split('.'); //Separates each triad

if (arIptriad.Length != 4) throw new Exception(); //Verify if the string has 4 '.'s, and if its not it calls exception

for (byte i = 0; i < arIptriad.Length; i++) //Verify if the number its a byte

{temp = Convert.ToByte(arIptriad[i]);}

MessageBox.Show(txtIpAddress.Text + "\nThe IP is correct.", "Valid IP");

}

catch (Exception)

{

MessageBox.Show(txtIpAddress.Text + "\nThe IP must have 4 bytes\ninteger number between 0 to 255\nseparated by a dot (255.255.255.255)", "Error");

txtIpAddress.Focus();

}

}

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

{

txtIpAddress.Text = "";

}

}

}

Lotto 649

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;

using System.IO;

using System.Text.RegularExpressions;

namespace WFOOPProject

{

public partial class Lotto649 : Form

{

public Lotto649()

{

InitializeComponent();

}

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

{

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

{

this.Close();

}

}

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

{

Random random = new Random();

string numberslist = txtNumbers.Text = "";

for (int i = 1; i < 8; i++)

{

int randomNumber;

do { randomNumber = random.Next(1, 50); }

while (Regex.IsMatch(numberslist, @"(" + randomNumber.ToString() + ")")); //Verifies if its reppeated and if yes, then it generates a new number.

txtNumbers.Text += randomNumber.ToString() + Environment.NewLine;

if (i == 7) numberslist += " Bonus " + randomNumber.ToString(); // verifies if its the last number and add Bonus if yes

else if (i == 6) numberslist += randomNumber.ToString(); // verifies if its the penultimate number and removes comma if yes (as required)

else numberslist += randomNumber.ToString() + ", ";

}

//Starting exportation

DateTime dateodnow = DateTime.Now;

FileStream directory = new FileStream(@"../../txtfiles/LottoNbrs.txt", FileMode.Append, FileAccess.Write);

StreamWriter file = new StreamWriter(directory);

file.WriteLine("649," + dateodnow.ToShortDateString() + " " + dateodnow.ToLongTimeString() + ", " + numberslist); ;

file.Close();

directory.Close();

}

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

{

try

{

if (!File.Exists(@"../../txtfiles/LottoNbrs.txt")) throw new Exception("File not created yet. Please generate some numbers first.");

FileStream directory = new FileStream(@"../../txtfiles/LottoNbrs.txt", FileMode.Open, FileAccess.Read);

StreamReader file = new StreamReader(directory);

string textoprint = file.ReadToEnd();

file.Close();

directory.Close();

MessageBox.Show(textoprint, "LottoNbrs.txt");

}

catch (Exception ex)

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

}

}

}

Lotto Max

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;

using System.IO;

using System.Text.RegularExpressions;

namespace WFOOPProject

{

public partial class LottoMax : Form

{

public LottoMax()

{

InitializeComponent();

}

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

{

}

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

{

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

{

this.Close();

}

}

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

{

Random random = new Random();

string numberslist = txtNumbers.Text = "";

for (int i = 1; i < 9; i++)

{

int randomNumber;

do { randomNumber = random.Next(1, 50); }

while (Regex.IsMatch(numberslist, @"(" + randomNumber.ToString() + ")")); //Verifies if its reppeated and if yes, then it generates a new number.

txtNumbers.Text += randomNumber.ToString() + Environment.NewLine;

if (i == 8) numberslist += " Bonus " + randomNumber.ToString(); // verifies if its the last number and add Bonus if yes

else if (i == 7) numberslist += randomNumber.ToString(); // verifies if its the penultimate number and removes comma if yes (as required)

else numberslist += randomNumber.ToString() + ", ";

}

//Starting exportation

DateTime dateodnow = DateTime.Now;

FileStream directory = new FileStream(@"../../txtfiles/LottoNbrs.txt", FileMode.Append, FileAccess.Write);

StreamWriter file = new StreamWriter(directory);

file.WriteLine("Max,"+dateodnow.ToShortDateString() + " " + dateodnow.ToLongTimeString() + ", " + numberslist); ;

file.Close();

directory.Close();

}

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

{

try

{

if (!File.Exists(@"../../txtfiles/LottoNbrs.txt")) throw new Exception("File not created yet. Please generate some numbers first.");

FileStream directory = new FileStream(@"../../txtfiles/LottoNbrs.txt", FileMode.Open, FileAccess.Read);

StreamReader file = new StreamReader(directory);

string textoprint = file.ReadToEnd();

file.Close();

directory.Close();

MessageBox.Show(textoprint, "LottoNbrs.txt");

}

catch (Exception ex)

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

}

}

}

Money Exchange

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;

using System.IO;

/\*Flags credits: https://www.flaticon.com/br/packs/countrys-flags \*/

namespace WFOOPProject

{

public partial class MoneyConversion : Form

{

public MoneyConversion()

{

InitializeComponent();

}

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

{

if (MessageBox.Show("Do you want to quit the application Money Exchange?", "Exit ?", MessageBoxButtons.YesNo).ToString() == "Yes")

{

this.Close();

}

}

public string ConvertValues (string Fromopt, string Toopt)

{

//The app will verify if the option chosed are the same

if(Fromopt == Toopt)

{ return (Convert.ToDouble(txtFrom.Text)).ToString(); } //Just to remove if there is 0 before

else

{

//If its not, then itt will get the strings togheter and it will compare it with the strings from the dictionaty to retrive the exchange factor

Dictionary<string, double> ExchangeFactor = new Dictionary<string, double>();

ExchangeFactor.Add("CADUSD", 0.797171);

ExchangeFactor.Add("USDCAD", 1.25444);

ExchangeFactor.Add("CADEUR", 1.48608);

ExchangeFactor.Add("EURCAD", 0.672911);

ExchangeFactor.Add("CADGBP", 1.72956);

ExchangeFactor.Add("GBPCAD", 0.578180);

ExchangeFactor.Add("CADBRL", 4.19256);

ExchangeFactor.Add("BRLCAD", 0.238518);

ExchangeFactor.Add("USDEUR", 0.844274);

ExchangeFactor.Add("EURUSD", 1.18445);

ExchangeFactor.Add("USDGBP", 0.725285);

ExchangeFactor.Add("GBPUSD", 1.37874);

ExchangeFactor.Add("BRLUSD", 0.190117);

ExchangeFactor.Add("USDBRL", 5.25992);

ExchangeFactor.Add("EURGBP", 0.859034);

ExchangeFactor.Add("GBPEUR", 1.16410);

ExchangeFactor.Add("EURBRL", 6.22808);

ExchangeFactor.Add("BRLEUR", 0.160561);

ExchangeFactor.Add("GBPBRL", 7.25179);

ExchangeFactor.Add("BRLGBP", 0.137889);

return (Convert.ToDouble(txtFrom.Text) \* ExchangeFactor[Fromopt+ Toopt]).ToString("0.00");

}

}

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

{

string FromOption = "", ToOption = "";

try

{

//Verifies the From option

foreach (RadioButton rbtnFrom in groupBox1.Controls.OfType<RadioButton>())

{ if (rbtnFrom.Checked) { FromOption = rbtnFrom.Text; break; } }

//Verifies the To option

foreach (RadioButton rbtnTo in groupBox2.Controls.OfType<RadioButton>())

{ if (rbtnTo.Checked) { ToOption = rbtnTo.Text; break; } }

if (ToOption == "") { throw new InvalidOperationException("Please select one 'To' radiobox option."); }

txtTo.Text = this.ConvertValues(FromOption, ToOption);

//Starting exportation

DateTime dateodnow = DateTime.Now;

FileStream directory = new FileStream(@"../../txtfiles/MoneyConversions.txt", FileMode.Append,FileAccess.Write);

StreamWriter file = new StreamWriter(directory);

file.WriteLine(txtFrom.Text + " " + FromOption + " = " + txtTo.Text + " " + ToOption + ", " + dateodnow.ToShortDateString() + " " + dateodnow.ToLongTimeString()); ;

file.Close();

directory.Close();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

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

{

try

{

if (!File.Exists(@"../../txtfiles/MoneyConversions.txt")) throw new Exception("File not created yet. Please do at least one convertion first.");

FileStream directory = new FileStream(@"../../txtfiles/MoneyConversions.txt", FileMode.Open, FileAccess.Read);

StreamReader file = new StreamReader(directory);

string textoprint = "FROM\tTO\t\tDATE\n" + file.ReadToEnd();

file.Close();

directory.Close();

MessageBox.Show(textoprint, "MoneyConversions.txt");

}

catch (Exception ex)

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

}

}

}

Calculator

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 WFOOPProject

{

public partial class SimpleCalc : Form

{

public SimpleCalc()

{

InitializeComponent();

}

Calculator Calc = new Calculator();

public void AddtoVisor(string valuetoadd)

{

txtVisor.Text += valuetoadd;

}

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

{

if (MessageBox.Show("Do you want to quit the application Simple Calculator?", "Exit ?", MessageBoxButtons.YesNo).ToString() == "Yes")

{

this.Close();

}

}

private void button1\_Click(object sender, EventArgs e)

{

AddtoVisor(button1.Text);

}

private void button2\_Click(object sender, EventArgs e)

{

AddtoVisor(button2.Text);

}

private void button3\_Click(object sender, EventArgs e)

{

AddtoVisor(button3.Text);

}

private void button4\_Click(object sender, EventArgs e)

{

AddtoVisor(button4.Text);

}

private void button5\_Click(object sender, EventArgs e)

{

AddtoVisor(button5.Text);

}

private void button6\_Click(object sender, EventArgs e)

{

AddtoVisor(button6.Text);

}

private void button7\_Click(object sender, EventArgs e)

{

AddtoVisor(button7.Text);

}

private void button8\_Click(object sender, EventArgs e)

{

AddtoVisor(button8.Text);

}

private void button9\_Click(object sender, EventArgs e)

{

AddtoVisor(button9.Text);

}

private void button0\_Click(object sender, EventArgs e)

{

AddtoVisor(button0.Text);

}

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

{

AddtoVisor(btndot.Text);

}

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

{

txtVisor.Text = "";

Calc.Clear();

}

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

{

try

{

Calc.Add(Convert.ToDecimal(txtVisor.Text));

txtVisor.Text = "";

}

catch (Exception ex)

{

MessageBox.Show("Please only numbers are valid. Ex: 12.34" + ex.Message);

txtVisor.Focus();

}

}

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

{

try

{

Calc.Subtract(Convert.ToDecimal(txtVisor.Text));

txtVisor.Text = "";

//txtVisor.Text = Convert.ToString((float)Calc.CurrentValue);

}

catch (Exception ex)

{

MessageBox.Show("Please only numbers are valid. Ex: 12.34" + ex.Message);

txtVisor.Focus();

}

}

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

{

try

{

Calc.Multiply(Convert.ToDecimal(txtVisor.Text));

txtVisor.Text = "";

//txtVisor.Text = Convert.ToString((float)Calc.CurrentValue);

}

catch (Exception ex)

{

MessageBox.Show("Please only numbers are valid. Ex: 12.34" + ex.Message);

txtVisor.Focus();

}

}

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

{

try

{

Calc.Divide(Convert.ToDecimal(txtVisor.Text));

txtVisor.Text = "";

//txtVisor.Text = Convert.ToString((float)Calc.CurrentValue);

}

catch (Exception ex)

{

MessageBox.Show("Please only numbers are valid. Ex: 12.34" + ex.Message);

txtVisor.Focus();

}

}

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

{

try

{

Calc.Equals(Convert.ToDecimal(txtVisor.Text));

txtVisor.Text = Convert.ToString((float)Calc.CurrentValue);

}

catch (Exception ex)

{

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

txtVisor.Focus();

}

}

private void checkBox1\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox1.Checked) btndot.Text = ",";

else btndot.Text = ".";

}

}

class Calculator

{

private string op = null;

private decimal operand1 = 0;

private decimal operand2 = 0;

private decimal currentValue = 0;

public decimal CurrentValue { get; set; }

public void Clear()

{

operand1 = operand2 = CurrentValue = 0;

op = null;

}

public void Add(decimal displayValue)

{

op = "+";

operand1 = CurrentValue = displayValue;

}

public void Subtract(decimal displayValue)

{

op = "-";

operand1 = CurrentValue = displayValue;

}

public void Multiply(decimal displayValue)

{

op = "\*";

operand1 = CurrentValue = displayValue;

}

public void Divide(decimal displayValue)

{

op = "/";

operand1 = CurrentValue = displayValue;

}

public void Equals()

{

switch (op)

{

case "+":

operand1 += operand2;

break;

case "-":

operand1 -= operand2;

break;

case "\*":

operand1 \*= operand2;

break;

case "/":

operand1 /= operand2;

break;

}

CurrentValue = operand1;

}

public void Equals (decimal displayValue)

{

operand2 = CurrentValue = displayValue;

Equals();

}

}

}

Temperature Converter

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;

using System.IO;

namespace WFOOPProject

{

public partial class TempApp : Form

{

public TempApp()

{

InitializeComponent();

}

Tempcalc Cal1 = new Tempcalc();

//True = C to F

//False = F to C

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

{

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

{

this.Close();

}

}

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

{

labelResult.Text = "C";

labelInput.Text = "F";

txtInput.Text = txtResult.Text = "";

Cal1.Convertstatus = false;

}

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

{

labelResult.Text = "F";

labelInput.Text = "C";

txtInput.Text = txtResult.Text = "";

Cal1.Convertstatus = true;

}

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

{

try

{

Cal1.Initval = Convert.ToDouble(txtInput.Text);

if (Cal1.Convertstatus)

Cal1.Finalval = ((Cal1.Initval \* 9 / 5) + 32);

else

Cal1.Finalval = (Cal1.Initval - 32) \* 5 / 9;

txtResult.Text = Cal1.Finalval.ToString();

txtboxMessage.Text = Cal1.Message();

//Starting exportation

DateTime dateodnow = DateTime.Now;

FileStream directory = new FileStream(@"../../txtfiles/TempConversions.txt", FileMode.Append, FileAccess.Write);

StreamWriter file = new StreamWriter(directory);

if (Cal1.Convertstatus)

file.WriteLine(txtInput.Text + " C = " + txtResult.Text + " " + " F , " + dateodnow.ToShortDateString() + " " + dateodnow.ToLongTimeString());

else

file.WriteLine(txtInput.Text + " F = " + txtResult.Text + " " + " C , " + dateodnow.ToShortDateString() + " " + dateodnow.ToLongTimeString());

file.Close();

directory.Close();

}

catch (Exception ex)

{

MessageBox.Show("Error :\n" + ex.Message + "\nPlease insert only numbers.", "Error");

txtInput.Focus();

}

}

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

{

try

{

if (!File.Exists(@"../../txtfiles/TempConversions.txt")) throw new Exception("File not created yet. Please do at least one convertion first.");

FileStream directory = new FileStream(@"../../txtfiles/TempConversions.txt", FileMode.Open, FileAccess.Read);

StreamReader file = new StreamReader(directory);

string textoprint = "FROM\tTO\t\tDATE\n" + file.ReadToEnd();

file.Close();

directory.Close();

MessageBox.Show(textoprint, "TempConversions.txt");

}

catch (Exception ex)

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

}

}

public class Tempcalc

{

private bool convertstatus;

private double initval, finalval;

public bool Convertstatus { get; set; }

public double Initval { get; set; }

public double Finalval { get; set; }

public string Message()

{

if ((Convertstatus && Initval == 100) || (!Convertstatus && Initval == 212))

{

return "Water boils";

} else if ((Convertstatus && Initval == 40) || (!Convertstatus && Initval == 104))

{

return "Hot Bath";

}else if ((Convertstatus && Initval == 37) || (!Convertstatus && Initval == 98.6))

{

return "Body temperature";

}else if ((Convertstatus && Initval == 30) || (!Convertstatus && Initval == 86))

{

return "Beach weather";

}else if ((Convertstatus && Initval == 21) || (!Convertstatus && Initval == 70))

{

return "Room temperature";

}else if ((Convertstatus && Initval == 10) || (!Convertstatus && Initval == 50))

{

return "Cool Day";

}else if ((Convertstatus&& Initval == 0) || (!Convertstatus && Initval == 32))

{

return "Freezing point of water";

}else if ((Convertstatus && Initval == -18) || (!Convertstatus && Initval == 0))

{

return "Very Cold Day";

}else if (Initval == -40)

{

return "Extremely Cold Day\n(and the same number!)";

}else { return ""; }

}

}

}

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

|  |  |
| --- | --- |
| **Class/Method Name** | **Description** |
| 1. Dashboard | Used for controlling the user chooses from the dashboard. |
| 1. IP4Validator | Used for controlling the user chooses from the IP4Validator Form. |
| 1. Lotto649 | Used for controlling the user chooses from the Lotto649 Form. |
| 1. LottoMax | Used for controlling the user chooses from the LottoMax Form. |
| 1. MoneyConversion | Used for controlling the user chooses from the Money Conversion Form. |
| 1. ConvertValues (string Fromopt, string Toopt) | Uses the strings Fromopt (means From option) and Toopt (means To option) and try to match with one of the dictionary created. |
| 1. Simple Calc | Used for controlling the user chooses from the Calculator Form. |
| 1. Calculator | Class used to perform operations of the calculator. |
| 1. Add(decimal displayValue) | Set the operand 1 and current value to the displayValue at the textbox, and set the operator to “+”. |
| 1. Subtract(decimal displayValue) | Set the operand 1 and current value to the displayValue at the textbox, and set the operator to “-”. |
| 1. Multiply(decimal displayValue) | Set the operand 1 and current value to the displayValue at the textbox, and set the operator to “\*”. |
| 1. Divide(decimal displayValue) | Set the operand 1 and current value to the displayValue at the textbox, and set the operator to “/”. |
| 1. Equals() | Perform a operation set in “op” in operand1 and operand2, stores the result in operand1, and sets the current value to the operand1. |
| 1. Equals (decimal displayValue) | Set the operand 2 and current value to the displayValue at the textbox, and execute Equals(). |
| 1. Clear() | Set all attributes to their default values. |
| 1. TempApp | Used for controlling the user chooses from the Temperature Converter Form. |
| 1. Tempcalc | Class used to perform operations of the temperature converter. |
| 1. Message() | Compares the temperature converter results with one of the strings reserved. |

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

Hardest part – The hardest part was to find an efficient way to deal with the Money exchange possibilities. I did not want to use many “if” clauses.

Easiest part – Build the design and make the “Exit” and “Read File” buttons.

One difficulties was to use the RegEx, I was not sure where to use since I was already using “try and catch”, but then I realize I could use it to verify if the number is repeated in Lotteries and also to create a new “throw” in try and catch.