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

|  |
| --- |
| Aafrin Yusuf Sayani  6/12/2021 |

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

1. Language – C#
2. tool Visual Studio version (2017 / 2019)

**Screenshots of the Console App:**

* **Global Menu:**

**Graphical user interface, text, application

Description automatically generated**

**1) Generate Numbers:**

**Graphical user interface, text, application

Description automatically generated**

* When User Generates a winning numbers

Text

Description automatically generated

* When User Reads the winning numbers

**­­**Text

Description automatically generated

**2) Option 2: Conversions**

Graphical user interface, text, application

Description automatically generated

* Conversions - Currency options

Graphical user interface, text, application

Description automatically generated

* Convert currency From CAD to GBP:

Text

Description automatically generated

* Convert the currency from USD to EUR:

Text

Description automatically generated

* First the converted values are being read from the classes and printing the result after getting result from the classes the function will write the value to the file and read the value from the file in order to print all values.
* Option 2: Temperature Conversions :

Graphical user interface, text, application

Description automatically generated

* When the user converts the temperature from Celsius to Fahrenheit:

Graphical user interface, application

Description automatically generated

* Convert Temperature from Celsius to Fahrenheit and Fahrenheit to Celsius

Text

Description automatically generated with medium confidenceText

Description automatically generated with low confidence

* First the converted values are being read from the classes and printing the result after getting result from the classes the function will write the value to the file and read the value from the file in order to print all values.
* 3) Calculator

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generatedText, letter

Description automatically generatedText

Description automatically generated

* 4) IP validator

Graphical user interface, text, application

Description automatically generated

Text, letter

Description automatically generated

* **Present the code of your application (Console).**

Code:

using System;

using System.Linq;

using System.IO;

/\*

Name: Aafrin Sayani

Project Description: This is a Charp Project which will be allowing you to do the followings

1. Generate and read/print Numbers

2. Conversion for Money/Temerature

3. Calculator for addition/ substracting/multiplying/division

4. IPV4 Validator

Date: 3 December, 2021

\*/

namespace c\_sharp\_project

{

class Program

{

static void Main(string[] args)

{

byte choice = 0, opt = 0, optNum = 0, optMoneyFrom = 0, optMoneyTo = 0, optTemp = 0;

double temp = 0, cad = 0, usd= 0, eur = 0,gbp= 0;

string ip;

Numbers obj\_num = new Numbers();

//ConvertArea obj = new ConvertArea();

Calculator calc\_obj = new Calculator(); //object (instance) creation

Conversion convert\_obj = new Conversion();

IPValidator ip\_obj = new IPValidator();

do

{

top1:

Console.Clear();

Console.WriteLine("Enter the option to choose:\n1) " + "Generate Numbers\n2) Conversions\n3) calculator\n4) " + "IPV4 Validator\n0) to quit");

Console.Write("Please Enter the options: ");

try

{

choice = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto top1;

}

switch (choice)

{

case 1: // lotto max and 649

{

do

{

lbl1:

Console.Clear();

Console.WriteLine("Enter the option:\n1) " + "Lotto max\n2) Lotto 649 \n0) to quit");

Console.Write("Please Enter the options: ");

try

{

opt = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl1;

}

switch (opt)

{

case 1: // lotto max

{

lbl11:

Console.Clear();

Console.WriteLine("Enter the option:\n1) " + "Generate Winning Numbers\n2) Read Numbers\n0) to quit");

Console.Write("Please Enter the options: ");

try

{

optNum = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl11;

}

switch (optNum)

{

case 1:

{

int check = 0;

string genNumbers = "";

Random rand = new Random();

int[] Lottery = new int[8];

genNumbers = "loto " + DateTime.Now.ToShortDateString() + " " + DateTime.Now.ToShortTimeString() + " ";

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

{

check = rand.Next(1, 50);

Lottery[i] = check;

Console.WriteLine(check);

if (i< (Lottery.Length - 1))

{

genNumbers += check.ToString() + ",";

}

else

{

genNumbers += "Bonus: " + check.ToString();

}

}

FileStream fs = new FileStream(@"./file.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(genNumbers);

objW.Close();

}

break;

case 2:

{

FileStream fsR = new FileStream(@"./file.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 0:

{

Console.WriteLine("Back to main menu");

}

break;

}

}

break;

case 2: // lotto 649

{

//double num = 0;

lbl11:

Console.Clear();

Console.WriteLine("Enter the option to choose:\n1) " + "Generate Winning Numbers\n2) Read Numbers\n0) to quit");

Console.Write("Please Enter the options: ");

try

{

optNum = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl11;

}

switch (optNum)

{

case 1:

{

int check = 0;

string genNumbers = "";

Random rand = new Random();

int[] Lottery = new int[7];

genNumbers = "649 " + DateTime.Now.ToShortDateString() + " " + DateTime.Now.ToShortTimeString() + " ";

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

{

check = rand.Next(1, 49);

Lottery[i] = check;

Console.WriteLine(check);

if (i < (Lottery.Length - 1))

{

genNumbers += check.ToString() + ", ";

}

else { genNumbers += "Bonus: " + check.ToString(); }

}

FileStream fs = new FileStream(@"./file1.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(genNumbers);

objW.Close();

}

break;

case 2:

{

FileStream fsR = new FileStream(@"./file1.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 0:

{

Console.WriteLine("Back to main menu");

}

break;

}

}

break;

case 0:

{

Console.WriteLine("Back to main menu");

}

break;

//default : { Console.WriteLine("Enter 1, 2 or 3 to quit"); }

} //end switch

Console.ReadKey();

} while (opt != 0);

}

break;

case 2: // conversions money and temperature

{

do

{

lbl2:

Console.Clear();

Console.WriteLine("Enter the option to convert:\n1) " + "MONEY\n2) Temperature\n0) to quit");

Console.Write("Please Enter the options: ");

try

{

opt = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2;

}

switch (opt)

{

case 1: // Money opt-1

{

//double num = 0;

lbl21:

Console.Clear();

Console.WriteLine("Choose currency to convert from :\n1) " + "CAD\n2) USD\n3) EUR\n4) GBP\n0) to quit");

Console.Write("Please Enter the options: ");

try

{

optMoneyFrom = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl21;

}

switch (optMoneyFrom)

{

case 1: // CAD to other currency

{

lbl211:

Console.Clear();

Console.WriteLine("Choose currency to convert into :\n1) " + "USD\n2) EUR\n3) GBP\n0) to quit");

try

{

optMoneyTo = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl211;

}

switch (optMoneyTo)

{

case 1: // CAD to USD

//case 1: // CAD to USD

{

lbl2111:

Console.WriteLine("Enter amount: ");

try

{

cad = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2111;

}

convert\_obj.Val = cad;

Console.WriteLine("{0} CAD = {1} USD", cad, convert\_obj.cadToUSD());

//var result = convert\_obj.cadToUSD();

string result = "";

result += cad.ToString() + " CAD = " + convert\_obj.cadToUSD().ToString() + " USD \n";

FileStream fs = new FileStream(@"./filectou.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./filectou.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

//{

// cad = Convert.ToDouble(Console.ReadLine());

// convert\_obj.Val = cad;

// Console.WriteLine("{0} CAD = {1} USD", cad, convert\_obj.cadToUSD());

// //var result = convert\_obj.cadToUSD();

// string result= "";

// result += convert\_obj.cadToUSD().ToString() + ", ";

// //var result += result1.ToString() + ", ";

// FileStream fs = new FileStream(@"./filectou.txt", FileMode.Append, FileAccess.Write);

// StreamWriter objW = new StreamWriter(fs);

// objW.WriteLine(result);

// objW.Close();

// FileStream fsR = new FileStream(@"./filectou.txt", FileMode.Open, FileAccess.Read);

// StreamReader objR = new StreamReader(fsR);

// string tempo = objR.ReadToEnd();

// Console.WriteLine(tempo);

// objR.Close();

//}

break;

case 2: // CAD to EUR

{

lbl2112:

Console.WriteLine("Enter amount: ");

try

{

cad = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2112;

}

convert\_obj.Val = cad;

Console.WriteLine("{0} CAD = {1} EUR", cad, convert\_obj.cadToEUR());

string result = "";

result += cad.ToString() + " CAD = " + convert\_obj.cadToEUR().ToString() + " EUR \n";

FileStream fs = new FileStream(@"./filectoE.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./filectoE.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 3: // CAD to GBP

{

lbl2113:

Console.WriteLine("Enter amount: ");

try

{

cad = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2113;

}

convert\_obj.Val = cad;

Console.WriteLine("{0} CAD = {1} GBP", cad, convert\_obj.cadToGBP());

string result = "";

result += cad.ToString() + " CAD = " + convert\_obj.cadToGBP().ToString() + " GBP \n";

FileStream fs = new FileStream(@"./filectog.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./filectog.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 0:

{

Console.WriteLine("quit");

}

break;

}

}

break;

case 2: // USD to other currency

{

lbl211:

Console.Clear();

Console.WriteLine("Choose currency to convert into :\n1) " + "CAD\n2) EUR\n3) GBP\n0) to quit");

Console.Write("Please Enter the options: ");

try

{

optMoneyTo = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl211;

}

switch (optMoneyTo)

{

case 1: // to CAD

{

lbl2111:

Console.WriteLine("Enter amount: ");

try

{

usd = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2111;

}

convert\_obj.Val = usd;

Console.WriteLine("{0} USD = {1} CAD", usd, convert\_obj.usdToCAD());

string result = "";

result += usd.ToString() + " USD = " + convert\_obj.usdToCAD().ToString() + " CAD \n";

FileStream fs = new FileStream(@"./fileutoc.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./fileutoc.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 2: // to EUR

{

lbl2112:

Console.WriteLine("Enter amount: ");

try

{

usd = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2112;

}

convert\_obj.Val = usd;

Console.WriteLine("{0} USD = {1} EUR", usd, convert\_obj.usdToEUR());

string result = "";

result += usd.ToString() + " USD = " + convert\_obj.usdToEUR().ToString() + " EUR \n";

FileStream fs = new FileStream(@"./fileutoe.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./fileutoe.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 3: // to GBP

{

lbl2113:

Console.WriteLine("Enter amount: ");

try

{

usd = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2113;

}

convert\_obj.Val = usd;

Console.WriteLine("{0} USD = {1} GBP", usd, convert\_obj.usdToGBP());

string result = "";

result += usd.ToString() + " USD = " + convert\_obj.usdToGBP().ToString() + " GBP \n";

FileStream fs = new FileStream(@"./fileutog.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./fileutog.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 0:

{

Console.WriteLine("Quiting..");

}

break;

}

}

break;

case 3: // EUR to other currency

{

lbl211:

Console.Clear();

Console.WriteLine("Choose currency to convert into :\n1) " + "CAD\n2) USD\n3) GBP\n0) to quit");

Console.Write("Please Enter the options: ");

try

{

optMoneyTo = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl211;

}

switch (optMoneyTo)

{

case 1: // to CAD

{

lbl2111:

Console.WriteLine("Enter amount: ");

try

{

eur = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2111;

}

convert\_obj.Val = eur;

Console.WriteLine("{0} EUR = {1} CAD", eur, convert\_obj.eurToCAD());

string result = "";

result += eur.ToString() + " EUR = " + convert\_obj.eurToCAD().ToString() + " CAD \n";

FileStream fs = new FileStream(@"./fileetoc.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./fileetoc.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 2: // to USD

{

lbl2112:

Console.WriteLine("Enter amount: ");

try

{

eur = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2112;

}

convert\_obj.Val = eur;

Console.WriteLine("{0} EUR = {1} USD", eur, convert\_obj.eurToUSD());

string result = "";

result += eur.ToString() + " EUR = " + convert\_obj.eurToUSD().ToString() + " USD \n";

FileStream fs = new FileStream(@"./fileetou.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./fileetou.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 3: // to GBP

{

lbl2113:

Console.WriteLine("Enter amount: ");

try

{

eur = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2113;

}

convert\_obj.Val = eur;

Console.WriteLine("{0} EUR = {1} GBP", eur, convert\_obj.eurToGBP());

string result = "";

result += eur.ToString() + " EUR = " + convert\_obj.eurToGBP().ToString() + " GBP \n";

FileStream fs = new FileStream(@"./fileetog.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./fileetog.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 0:

{

Console.WriteLine("Quiting..");

}

break;

}

}

break;

case 4: // GBP to other currency

{

lbl211:

Console.Clear();

Console.WriteLine("Choose currency to convert into :\n1) " + "CAD\n2) USD\n3) EUR\n0) to quit");

Console.Write("Please Enter the options: ");

try

{

optMoneyTo = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl211;

}

switch (optMoneyTo)

{

case 1: // to CAD

{

lbl2111:

Console.WriteLine("Enter amount: ");

try

{

gbp = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2111;

}

convert\_obj.Val = gbp;

Console.WriteLine("{0} GBP = {1} CAD", gbp, convert\_obj.gbpToCAD());

string result = "";

result += gbp.ToString() + " GBP = " + convert\_obj.gbpToCAD().ToString() + " CAD \n";

FileStream fs = new FileStream(@"./filegtoc.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./filegtoc.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 2: // to USD

{

lbl2112:

Console.WriteLine("Enter amount: ");

try

{

gbp = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2112;

}

convert\_obj.Val = eur;

Console.WriteLine("{0} GBP = {1} USD", eur, convert\_obj.gbpToUSD());

string result = "";

result += gbp.ToString() + " GBP = " + convert\_obj.gbpToUSD().ToString() + " USD \n";

FileStream fs = new FileStream(@"./filegtou.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./filegtou.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 3: // to EUR

{

lbl2113:

Console.WriteLine("Enter amount: ");

try

{

gbp = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl2113;

}

convert\_obj.Val = gbp;

Console.WriteLine("{0} GBP = {1} EUR", gbp, convert\_obj.gbpToEUR());

string result = "";

result += gbp.ToString() + " GBP = " + convert\_obj.gbpToEUR().ToString() + " EUR \n";

FileStream fs = new FileStream(@"./filegtoe.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./filegtoe.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 0:

{

Console.WriteLine("Quiting..");

}

break;

}

}

break;

case 0:

{

Console.WriteLine("Back to main menu");

}

break;

}

}

break;

case 2: // Temperature main

{

//double num = 0;

lbl22:

Console.Clear();

Console.Write("Enter the option to convert temperature: \n1) C to F\n2) F to C\n0) to quit\n");

Console.Write("Please Enter the options: ");

try

{

optTemp = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl22;

}

switch (optTemp)

{

case 1: // C to F

{

lbl221:

Console.Clear();

Console.Write("Enter the value for Temp in C: ");

//double convertedToC = 0;

//double convertedTof = 0;

try

{

temp = Convert.ToDouble(Console.ReadLine());

convert\_obj.Val = temp;

double convertedTof = convert\_obj.cToF();

Console.WriteLine("{0} C = {1} F", temp, convertedTof);

string result = "";

result += temp.ToString() + " C = " + convert\_obj.cToF().ToString() + " F \n";

FileStream fs = new FileStream(@"./filectof.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./filectof.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

//double convertedTof = convert\_obj.cToF();

if (convertedTof == 212)

{

Console.WriteLine("Water Boils!");

}

else if (convertedTof == 104)

{

Console.WriteLine("Hot Bath!");

}

else if (convertedTof == 98.6)

{

Console.WriteLine("Body Temperature!");

}

else if (convertedTof == 86)

{

Console.WriteLine("Beach Weather!");

}

else if (convertedTof <= 70 && convertedTof > 50)

{

Console.WriteLine("Room Temperature!");

}

else if (convertedTof == 50)

{

Console.WriteLine("Cool Day!");

}

else if (convertedTof == 32)

{

Console.WriteLine("Freezing Point of water!");

}

else if (convertedTof > -40 && convertedTof <= 0)

{

Console.WriteLine("Very Cold Day!");

}

else if (convertedTof == -40)

{

Console.WriteLine("Extremely Cold Day!");

}

else

{

Console.WriteLine("Good Bye!");

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

goto lbl221;

}

}

break;

case 2: // F to C

{

lbl222:

Console.Clear();

Console.Write("Enter the value for F: ");

try

{

temp = Convert.ToDouble(Console.ReadLine());

convert\_obj.Val = temp;

double convertedToC = convert\_obj.fToC();

Console.WriteLine("{0} F = {1} C", temp, convertedToC);

string result = "";

result += temp.ToString() + " F = " + convert\_obj.fToC().ToString() + " C \n";

FileStream fs = new FileStream(@"./fileftoc.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./fileftoc.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

if (convertedToC == 100)

{

Console.WriteLine("Water Boils!");

}

else if (convertedToC == 40)

{

Console.WriteLine("Hot Bath!");

}

else if (convertedToC == 37)

{

Console.WriteLine("Body Temperature!");

}

else if (convertedToC == 30)

{

Console.WriteLine("Beach Weather!");

}

else if (convertedToC >= 10 && convertedToC <= 21)

{

Console.WriteLine("Room Temperature!");

}

else if (convertedToC == 10)

{

Console.WriteLine("Cool Day!");

}

else if (convertedToC == 0)

{

Console.WriteLine("Freezing Point of water!");

}

else if (convertedToC >= -18 && convertedToC <=0)

{

Console.WriteLine("Very Cold Day!");

}

else if (convertedToC == -40)

{

Console.WriteLine("Extremely Cold Day!");

}

else

{

Console.WriteLine("temperature type");

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

goto lbl222;

}

}

break;

case 0: // default

{

Console.Write("Back to main menu");

}

break;

}

}

break;

case 0:

{

Console.WriteLine("Back to main menu");

}

break;

//default : { Console.WriteLine("Enter 1, 2 or 3 to quit"); }

}//end switch

Console.ReadKey();

} while (opt != 0);

}//end case 2

break;

case 3: // Calculator

{

do

{

lbl3:

Console.Clear();

Console.WriteLine("Enter the value to select actions:\n1) " + "Addition(+)\n2) Substraction(-)\n3) Multiplication(x)\n4) Division(/)\n0) to quit");

Console.Write("Please Enter the options: ");

try

{

opt = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message);

goto lbl3;

}

switch (opt)

{

case 1: // Addition

{

double num1 = 0, num2 = 0;

lbl31:

Console.Write("Enter the first number: ");

try

{

num1 = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

goto lbl31;

}

Console.Write("Enter the second number: ");

try

{

num2 = Convert.ToDouble(Console.ReadLine());

calc\_obj.Calc\_val1 = num1;

calc\_obj.Calc\_val2 = num2;

Console.WriteLine("{0} + {1} = {2}", num1, num2, calc\_obj.Add());

string result = "";

result += num1.ToString() + " + " + num2.ToString() + " = " + calc\_obj.Add().ToString() + "\n";

FileStream fs = new FileStream(@"./fileadd.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./fileadd.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

goto lbl31;

}

}

break;

case 2: // Substraction

{

double num1 = 0, num2 = 0;

lbl32:

Console.Write("Enter the first number: ");

try

{

num1 = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

goto lbl32;

}

Console.Write("Enter the second number: ");

try

{

num2 = Convert.ToDouble(Console.ReadLine());

calc\_obj.Calc\_val1 = num1;

calc\_obj.Calc\_val2 = num2;

Console.WriteLine("{0} - {1} = {2}", num1, num2, calc\_obj.Sub());

string result = "";

result += num1.ToString() + " - " + num2.ToString() + " = " + calc\_obj.Sub().ToString() + "\n";

FileStream fs = new FileStream(@"./filesub.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./filesub.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

goto lbl32;

}

}

break;

case 3: // Multiplications

{

double num1 = 0, num2 = 0;

lbl33:

Console.Write("Enter the first number: ");

try

{

num1 = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

goto lbl33;

}

Console.Write("Enter the second number: ");

try

{

num2 = Convert.ToDouble(Console.ReadLine());

calc\_obj.Calc\_val1 = num1;

calc\_obj.Calc\_val2 = num2;

Console.WriteLine("{0} x {1} = {2}", num1, num2, calc\_obj.Mul());

string result = "";

result += num1.ToString() + " x " + num2.ToString() + " = " + calc\_obj.Mul().ToString() + "\n";

FileStream fs = new FileStream(@"./filemul.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./filemul.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

goto lbl33;

}

}

break;

case 4: // Division

{

double num1 = 0, num2 = 0;

lbl34:

Console.Write("Enter the first number: ");

try

{

num1 = Convert.ToDouble(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

goto lbl34;

}

Console.Write("Enter the second number: ");

try

{

num2 = Convert.ToDouble(Console.ReadLine());

calc\_obj.Calc\_val1 = num1;

calc\_obj.Calc\_val2 = num2;

Console.WriteLine("{0} / {1} = {2}", num1, num2, calc\_obj.Div());

string result = "";

result += num1.ToString() + " / " + num2.ToString() + " = " + calc\_obj.Div().ToString() + "\n";

FileStream fs = new FileStream(@"./filediv.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./filediv.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

goto lbl34;

}

}

break;

case 0:

{

Console.WriteLine("Back to main menu");

}

break;

//default : { Console.WriteLine("Enter 1, 2 or 3 to quit"); }

}//end switch

Console.ReadKey();

} while (opt != 0);

}//end case 3

break;

case 4: // validate IPv4

{

do

{

lbl4:

Console.Clear();

Console.WriteLine("Enter the IPV4:\n1) " + "validate IP\n2) reset\n0) to quit");

Console.Write("Please Enter the options: ");

try

{

opt = Convert.ToByte(Console.ReadLine());

}

catch (Exception ex)

{

Console.WriteLine("Error\n" + ex.Message); goto lbl4;

}

switch (opt)

{

case 1:

{

Console.Write("Enter the IPV4 to validate : ");

ip = Console.ReadLine();

ip\_obj.IP = ip;

Console.WriteLine("{0} ip validation is {1}", ip, ip\_obj.ipValidator());

string result = "";

result += ip.ToString() + " ip validation is " + ip\_obj.ipValidator().ToString() + "\n";

FileStream fs = new FileStream(@"./fileip.txt", FileMode.Append, FileAccess.Write);

StreamWriter objW = new StreamWriter(fs);

objW.WriteLine(result);

objW.Close();

FileStream fsR = new FileStream(@"./fileip.txt", FileMode.Open, FileAccess.Read);

StreamReader objR = new StreamReader(fsR);

string tempo = objR.ReadToEnd();

Console.WriteLine(tempo);

objR.Close();

}

break;

case 2:

{

Console.Clear();

}

break;

case 0:

{

Console.WriteLine("Back to main menu");

}

break;

//default : { Console.WriteLine("Enter 1, 2 or 3 to quit"); }

}//end switch

Console.ReadKey();

} while (opt != 0);

}//end case 3

break;

case 0:

{

Console.WriteLine("Bye");

}

break;

}//end switch

} while (choice != 0);

Console.ReadKey();

}

}

//obj.Calc\_val1 = num1;

//obj.Calc\_val2 = num2;

//Console.WriteLine("{0} + {1} = {2}", num1, num2, obj.Add());

// Class generate numbers

class Numbers

{

private int num;

public int Num

{

set

{

num = value;

}

get

{

return num;

}

}

public Numbers() { } // default constructor

// constructor with one parameter

public Numbers(int val)

{

this.Num = num;

}

}

// Class Conversion

class Conversion

{

private double val;

public double Val

{

set

{

val = value;

// write into private field

}

get

{

return val;

}

}

public Conversion() { } // default constructor

// constructor with two parameter

public Conversion(double val)

{

this.Val = val;

}

public double cToF()

{

return (Val \* 9/5)+ 32;

}

public double fToC()

{

return (Val - 32)\* 5/9 ;

}

public double cadToUSD()

{

return Val \* 0.80;

}

public double cadToEUR()

{

return Val \* 0.70;

}

public double cadToGBP()

{

return Val \* 0.59;

}

public double usdToCAD()

{

return Val \* 1.26;

}

public double usdToEUR()

{

return Val \* 0.88;

}

public double usdToGBP()

{

return Val \* 0.74;

}

public double eurToCAD()

{

return Val \* 1.42;

}

public double eurToUSD()

{

return Val \* 1.13;

}

public double eurToGBP()

{

return Val \* 0.84;

}

public double gbpToCAD()

{

return Val \* 1.69;

}

public double gbpToUSD()

{

return Val \* 1.34;

}

public double gbpToEUR()

{

return Val \* 1.19;

}

}

// Class calculator

class Calculator

{

// private field

private double calc\_val1;

private double calc\_val2;

// public property Calc\_val1

public double Calc\_val1

{

set

{

calc\_val1 = value;

//calc\_val2 = value2;

// write into private field

}

get

{

return calc\_val1;

//return calc\_val2;

// read private field value

}

}

// public property Calc\_val2

public double Calc\_val2

{

set

{

calc\_val2 = value;

// write into private field

}

get

{

return calc\_val2;

// read private field value

}

}

public Calculator() { } // default constructor

// constructor with two parameter

public Calculator(double calc\_val1, double calc\_val2)

{

this.Calc\_val1 = calc\_val1;

this.Calc\_val2 = calc\_val2;

}

public double Add()

{

return Calc\_val1 + Calc\_val2;

}

public double Sub()

{

return Calc\_val1 - Calc\_val2;

}

public double Mul()

{

return Calc\_val1 \* Calc\_val2;

}

public double Div()

{

return Calc\_val1 / Calc\_val2;

}

}

class IPValidator

{

private string ip;

public string IP

{

set

{

ip = value; // private field

}

get

{

return ip; // read

}

}

public IPValidator() { }

public IPValidator(string ip)

{

this.IP = ip;

}

public bool ipValidator()

{

if (String.IsNullOrWhiteSpace(IP))

{

Console.WriteLine("Please remove spaces");

return false;

}

string[] splitValues = IP.Split('.');

if (splitValues.Length != 4)

{

Console.WriteLine("IP length doesn't match with IPv4.");

return false;

}

byte tempForParsing;

return splitValues.All(r => byte.TryParse(r, out tempForParsing));

}

}

// OLD

/\*

class ConvertArea

{

//private field

private double val;

//public Propriety

public double Val

{

set

{

val = value;

} //write into private field

get

{

return val;

} //read private field value

}

//default constructor

public ConvertArea() { }

//constructor with one parameter

public ConvertArea(double val)

{

this.Val = val;

}

//public methods

//public double sqCmtosqIn()

//{

// return Val \* 0.155;

//}

//public double sqMmtosqIn()

//{

// return Val / 100 \* 0.155;

//}

//public double sqMtosqYd()

//{

// return Val \* 1.196;

//}

//public double sqCmtosqYd()

//{

// return Val / 10000 \* 1.196;

//}

//public double HatoAcres()

//{

// return Val \* 2.4711;

//}

//public double sqMtoAcres()

//{

// return Val / 10000 \* 2.4711;

//}

//public double sqKmtosqMile()

//{

// return Val \* 0.3861;

//}

//public double HatosqMile()

//{

// return Val / 100 \* 0.3861;

//}

}

\*/

}

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

|  |  |
| --- | --- |
| **Class/Method Name** | **Description** |
| class Numbers | This class has all the methods which will be used for generating random winning numbers and reading values using setter and getter |
| class Conversions | Conversion class contains all the different methods like “cadToUsd()” to convert one currency from the other by using formulas in all methods and returning back all the values from the class. |
| class Calculator | Calculator class has all the methods related to the functions of the calculator like adding substracting and multiplying etc to compute values and returning back the result. |
| class IPValidator | This class has the authentication for validating the ip address input by user and validates the input and returns back the result. |

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

The Whole journey in making this project was roller coaster I learnt so many things from this project and in the end was amazed by the result because in the beginning It was difficult for me to deal with the classes but now learned so many new things from it, so thank you for this project.