Laboratory Practical Report

of

**Visual Programming with C#**

**(ICT ED 465)**

Submitted To

**TRIBHUVAN UNIVERSITY**

In Partial Fulfillment of the Requirements of the course

**B.Ed. ICTE 6th Semester**

Submitted By

Sanam Tamang

Symbol No.: 76214020

T.U. Regd. No.: 9-2-214-54-2019

Under the guidance of

**Er. Santosh Dahal**

Lecturer

Sukuna Multiple Campus

**SUKUNA MULTIPLE CAMPUS**

Sundarharaincha-12, Morang, Nepal

2080

**CERTIFICATE**

This is to certify that the Laboratory Practical Report

of

**Visual Programming with C#**

**(ICT ED 465)**

In Partial Fulfillment of the Requirements of the course

**B.Ed. ICTE 6th Semester**

Submitted By

Sanam Tamang

Symbol No.: 76214020

T.U. Regd. No.: 9-2-214-54-2019

is a bonafide record of experiments carried out by him/her under by guidance.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Er. Santosh Dahal

Lecturer

Sukuna Multiple Campus

Sundarharaincha-12, Morang

(Internal Examiner)

Submitted for the Final Examination on: 2080/08/20

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Lecturer

(External Examiner

**Table of Contents**

[**Introduction to C# Application** 1](#_Toc152571036)

[**Building a Calculator in C# with Windows Forms step by step process** 1](#_Toc152571037)

[**Getting Started** 2](#_Toc152571038)

[**Program.cs** 2](#_Toc152571039)

[**Explaining C# Code** 3](#_Toc152571040)

[**Form1.cs** 4](#_Toc152571041)

[**Explaining C# Code** 7](#_Toc152571042)

[**Output:** 8](#_Toc152571043)

# **Introduction to C# Application**

C# is a general-purpose, object-oriented programming language developed by Microsoft. It's known for its:

Ease of learning: C# syntax is similar to familiar languages like Java and C++, making it approachable for beginners.

Versatility: C# can create various applications, from desktop programs like your calculator to web services and mobile apps.

Object-oriented: C# revolves around objects, representing real-world entities with their own data and behavior. This promotes modularity and code reuse.

Powerful libraries: The .NET framework provides a vast ecosystem of libraries for tasks like database access, networking, and graphics.

# **Building a Calculator in C# with Windows Forms step by step process**

Creating a calculator with C# and Windows Forms involves several key steps:

1. Setting Up the Project:

* Open your Visual Studio and create a new Windows Forms Application project.
* Name it "Calculator" or something relevant.
* This creates a basic form for your calculator interface.

2. Designing the Interface:

* Drag and drop buttons from the Toolbox onto the form for numbers, operators ( +, -, \*, / ), and other functions (C, ., =).
* Add a text box to display calculations and results.
* Arrange the elements for a user-friendly layout.

3. Writing the Code:

* Declare variables to store operands (numbers) and the operator used.
* Write event handlers for button clicks:
  + Number buttons: Append the clicked number to the text box.
  + Operator buttons: Store the operator and prepare for the next operand.
  + Equal button: Perform the calculation based on the stored operator and operands. Use C#'s built-in operators like "+" or functions like Math.Pow() for exponentiation.
  + C button: Clear the text box and stored values.
  + Decimal button: Allow entering decimal numbers.

4. Handling Errors:

* Implement checks to avoid invalid inputs like dividing by zero.
* Display appropriate error messages in the text box for users.

5. Adding Functionality:

* Memory buttons (M+, M-, MR, MC) to store and recall values.

# **Getting Started**

# **Program.cs**

﻿using System;

using System.Windows.Forms;

namespace Advanced\_Calculator

{

internal static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

# **Explaining C# Code**

* System: This is the most fundamental namespace in C# and provides basic types like int, string, and bool, as well as essential functionality like file access and console input/output.
* System.Collections.Generic: This namespace provides classes like List and Dictionary for storing and managing collections of data in a flexible way.
* System.Linq: This namespace offers powerful extension methods for querying and manipulating collections, making it easier to filter, sort, and aggregate data.
* System.Threading.Tasks: This namespace provides tools for working with asynchronous tasks, which allows your program to perform multiple operations concurrently without blocking the main thread.
* namespace Advanced\_Calculator: This line declares that the following code belongs to a namespace named Advanced\_Calculator. This helps organize your code and prevents naming conflicts.
* internal static class Program: This defines a class called Program with an internal access modifier. This means the class is only accessible within the current assembly (your program).
* static void Main(): This is the main entry point of your program. When you run the program, this method is called first.
* Application.EnableVisualStyles(): This line enables the Windows Forms visual styles, which makes your application look more modern and consistent with the operating system theme.
* Application.SetCompatibleTextRenderingDefault(false): This disables the default text rendering behavior, which can sometimes lead to blurry text on high-resolution displays.
* Application.Run(new Form1()): This line starts the main message loop of your application. It creates a new instance of the Form1 class and runs it. This is where your calculator's user interface and functionality will be implemented.

# **Form1.cs**

﻿using System;

using System.Drawing;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Advanced\_Calculator{

public partial class Form1 : Form {

decimal fstNum, secNum = 0.0m;

string result;

string operation = "";

[DllImport("dwmapi", PreserveSig = false)]

static extern void DwmSetWindowAttribute(IntPtr hwnd, int dwAttribute, in bool pvAttribute, int cbAttribute);

protected override void OnHandleCreated(EventArgs e) {

const int DWMWA\_USE\_IMMERSIVE\_DARK\_MODE = 20;

DwmSetWindowAttribute(Handle, DWMWA\_USE\_IMMERSIVE\_DARK\_MODE, true, Marshal.SizeOf<bool>());

}

public Form1() {

InitializeComponent();

}

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

{

if (!txtBox.Text.Contains("."))

{

txtBox.Text += ".";

} }

private void BtnNums\_Click(object sender, EventArgs e) {

if (txtBox.Text == result)

txtBox.Text = "0";

btnPlus.Enabled = true;

btnMinus.Enabled = true;

btnMultiply.Enabled = true;

btnDivide.Enabled = true;

Button button = (Button)sender;

if (txtBox.Text == "0")

{

txtBox.Text = button.Text;

}

else

{

txtBox.Text += button.Text; //txtBox.Text += button.Text ==> txtBox.Text = txtBox.Text + button.Text

}

}

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

{

if (txtBox.Text == "Infinity" || txtBox.Text == "∞")

{

txtBox.Text = "0";

MessageBox.Show("Can not operate with infinity");

btnC.PerformClick();

}

else {

try

{

btnEquals.PerformClick();

Button button = (Button)sender;

fstNum = decimal.Parse(txtBox.Text);

operation = button.Text;

txtBox.Text = "0";

txtDis.Text = fstNum.ToString() + " " + operation;

btnPlus.Enabled = false;

btnMinus.Enabled = false;

btnMultiply.Enabled = false;

btnDivide.Enabled = false;

}

catch (System.OverflowException)

{

MessageBox.Show("The Number Is Too Long.");

}}}

private void BtnEquals\_Click(object sender, EventArgs e) {

secNum = decimal.Parse(txtBox.Text);

txtDis.Text = $"{txtDis.Text} {txtBox.Text} =";

btnPlus.Enabled = true;

btnMinus.Enabled = true;

btnMultiply.Enabled = true;

btnDivide.Enabled = true;

try{

switch (operation){

case "+":

result = txtBox.Text = (fstNum + secNum).ToString();

break;

case "−":

result = txtBox.Text = (fstNum - secNum).ToString();

break;

case "×":

result = txtBox.Text = (fstNum \* secNum).ToString();

break;

case "÷":

try{

result = txtBox.Text = (fstNum / secNum).ToString();}

catch (System.DivideByZeroException){

result = txtBox.Text = "Infinity"; }

break;

default:

txtDis.Text = $"{txtBox.Text} =";

break;

} }

catch (System.OverflowException) {

MessageBox.Show("The Number Is Too Long.");

} }

private void BtnBack\_Click(object sender, EventArgs e) {

if (txtBox.Text.Length > 0) {

txtBox.Text = txtBox.Text.Remove(txtBox.Text.Length - 1, 1);

}

if (txtBox.Text == "")

txtBox.Text = txtBox.Text = "0"; }

private void BtnCE\_Click(object sender, EventArgs e) {

txtBox.Text = "0"; }

private void BtnOperation\_Click(object sender, EventArgs e) {

Button button = (Button)sender;

operation = button.Text;

switch (operation) {

case "√x":

txtDis.Text = $"√({txtBox.Text})";

result = txtBox.Text = Convert.ToString(Math.Sqrt(Convert.ToDouble(txtBox.Text)));

break;

case "^2":

txtDis.Text = $"({txtBox.Text})^2";

result = txtBox.Text = Convert.ToString(Convert.ToDouble(txtBox.Text) \* (Convert.ToDouble(txtBox.Text)));

break;

case "¹/x":

txtDis.Text = $"¹/({txtBox.Text})";

result = txtBox.Text = Convert.ToString(1.0 / Convert.ToDouble(txtBox.Text));

break;

case "%":

txtDis.Text = $"%({txtBox.Text})";

result = txtBox.Text = Convert.ToString(Convert.ToDouble(txtBox.Text) / Convert.ToDouble(100));

break;

default:

break; }}

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

Point form\_pt = new Point(this.Left, this.Top);

Info info = new Info{

StartPosition = FormStartPosition.Manual,

Location = form\_p; }

info.Show(); }

private void BtnC\_Click(object sender, EventArgs e) {

Point form\_pt = new Point(this.Left, this.Top);

Form1 frm = new Form{

StartPosition = FormStartPosition.Manual,

Location = form\_pt};

frm.Show();

this.Hide();

frm.Closed += (s, args) => this.Close();

}

private void BtnMP\_Click(object sender, EventArgs e){

txtBox.Text = Convert.ToString(-1 \* double.Parse(txtBox.Text));

}

}

}

## **Explaining C# Code**

Overall structure:

* The code defines a class called Form1 which represents the main window of the calculator.
* It uses various variables to store the current state (operands, operation, result, etc.).
* Several functions handle different user interactions like clicking buttons, entering numbers, and performing calculations.

Functions:

* BtnDot\_Click: Adds a decimal point to the textbox only if there isn't one already.
* BtnNums\_Click: Handles clicking number buttons. It replaces the textbox content with the clicked number if it's 0, otherwise appends it.
* Operation\_Click: Handles clicking operation buttons (+, -, \*, /). It performs the previous operation first if there's one pending, then stores the chosen operation and enables/disables other buttons.
* BtnEquals\_Click: Performs the chosen operation with the second number entered, updates the result and display, and resets the state for further calculations.
* BtnBack\_Click: Deletes the last character from the textbox.
* BtnCE\_Click: Clears the textbox and resets to displaying 0.
* BtnOperation\_Click: Handles clicking special operation buttons like square root, power, reciprocal, and percentage. It performs the operation directly on the current number and updates the display.
* BtnInfo\_Click: Opens a separate information window.
* BtnC\_Click: Creates a new instance of the same form and hides the current one, essentially restarting the calculator.
* BtnMP\_Click: Negates the current number displayed in the textbox.

Additional details:

* The code uses try-catch blocks to handle potential errors like dividing by zero or entering too large numbers.
* It also utilizes some Windows Forms functionalities like DwmSetWindowAttribute for dark mode and MessageBox.Show for displaying messages.

Overall, this code demonstrates a well-structured and functional calculator application with basic and advanced operations.

# **Output:**

