GLA UNIVERSITY
ESTD. 2010
ऋते ज्ञानान्न मुक्तिः

GLA
UNIVERSITY
MATHURA
Recognised by UGC Under Section 2(f)

Accredited with A+ Grade by NAAC
3.46 Score

12-B Status from UGC

(SESSION 2023-2024)

MCA –IIIrd SEMESTER

PRACTICAL FILE : .NET FRAMEWORK

SUBMITTED TO:

DR.SACHENDRA SINGH

(ASSISTANT PROFESSOR)

DEPT. OF C.E.A.

SUBMITTED BY:

NAME : MOHIT KUMAR

SECTION : A 2

UNIV. ROLL No : 2284200121

INDEX

Que1. Installation of visual studio step by step

1. **Download Visual Studio:**

- Visit https://visualstudio.microsoft.com/.
- Select your preferred edition (e.g., Community).
- Download the installer.

2. **Run Installer:**

- Execute the downloaded installer.

3. **Customize Installation:**

- Choose development workloads (e.g., ".NET desktop," "ASP.NET," etc.).

- Optionally select individual components based on your preferences.

4. **Start Installation:**

- Click "Install" to begin the installation process.
- Wait for the installation to complete.

5. **Launch and Setup:**

- Launch Visual Studio.
- Sign in or create a Microsoft account (optional).
- Choose default development settings.
- Start using Visual Studio for your projects.

These five steps cover the essential stages of downloading, installing, customizing, and setting up Visual Studio for your development needs.

Que2. Creating a simple .net platform program with classes library and functions.

Step 1: Create a Class Library

1. **Open Visual Studio:**

    - Launch Visual Studio.

2. **Create a Class Library Project:**

- Select "Create a new project."
- Choose "Class Library" under the "Class Library" category in the language of your choice (e.g., C#).
- Name the project (e.g., MathLibrary) and click "Create."

3. **Define a Class and Function:**

- Open the default class file (e.g., Class1.cs).
- Replace the content with a simple math function:

4. **Build the Class Library:**

- Build the solution to ensure there are no errors.

Step 2: Create a Console Application

1. **Add a Console Application to the Solution:**

- Right-click on the solution in Solution Explorer.

- Choose "Add" -> "New Project."
- Select "Console App" under the language of your choice (e.g., C#).
- Name the project (e.g., ConsoleApp) and click "Create."

2. Reference the Class Library:

- Right-click on the console application project.
- Choose "Add" -> "Reference."
- Select the class library project (MathLibrary) and click "OK."

3. Use the Class Library in the Console Application:

- Open the Program.cs file in the console application.

Build and Run:

- Build the solution.
- Run the console application.

Que3. Conditional statement in c#

In C#, conditional statements are used to control the flow of execution based on certain conditions. The primary conditional statements in C# are if, else if, else, and the switch statement. Here are examples of each:

1. if Statement:

The if statement allows you to execute a block of code if a specified condition is true.

```
Int  number = 10;


if (number > 0)

{

   Console.WriteLine("The number is positive.");

}
```

2. if-else Statement:

The if-else statement allows you to execute one block of code if the condition is true and another block if the condition is false.

```
int number = -5;


if (number > 0)

{

    Console.WriteLine("The number is positive.");

}

else

{

    Console.WriteLine("The number is non-positive.");

}
```

3. else if Statement:

The else if statement is used to test multiple conditions. It is executed if the preceding if or else if conditions are false and its own condition is true.

```
int number = 0;


if (number > 0)

{

    Console.WriteLine("The number is positive.");

}

else if (number < 0)

{

    Console.WriteLine("The number is negative.");

}

else

{

    Console.WriteLine("The number is zero.");
}


{
    Console.WriteLine("The number is positive.");
}
```

4. switch Statement:

The switch statement is used when you have multiple possible conditions to test. It provides a concise way to compare a variable against multiple values.

```
int dayOfWeek = 3;


switch (dayOfWeek)
{
    case 1:
        Console.WriteLine("Monday");
        break;
    case 2:
        Console.WriteLine("Tuesday");
        break;
    case 3:
        Console.WriteLine("Wednesday");
        break;
    // ... other cases ...
    default:
        Console.WriteLine("Invalid day");
        break;
}
```

Que4. Loops statements in C#

1. for Loop:

The for loop is used when you know in advance how many times you want to execute a block of code.

<u>Syntax :</u>

```
for (int i = 0; i < 5; i++)

{

   Console.WriteLine($"Iteration {i + 1}");

}
```

2. while Loop:

The while loop is used when you want to repeat a block of code as long as a specified condition is true.

int count = 0;

<u>Syntax :</u>

```
while (count < 5)

{

   Console.WriteLine($"Iteration {count + 1}");

   count++;

}
```

3. do-while Loop:

The do-while loop is similar to the while loop, but it ensures that the block of code is executed at least once before checking the condition

<u>Syntax :</u>

```
int count = 0;

do

{

   Console.WriteLine($"Iteration {count + 1}");

   count++;
} while (count < 5);
```

4. foreach Loop:

The foreach loop is used to iterate over elements in an array or a collection.

Syntax :

```
int[] numbers = { 1, 2, 3, 4, 5 };

foreach (int number in numbers)

{

    Console.WriteLine($"Number: {number}");

}
```

Que5. Classes in C#

1. Declaring a Class:

You declare a class using the class keyword, followed by the class name. The body of the class contains fields, properties, methods, and other members.

```
public class MyClass

{

    // Fields

    private int myField;


    // Properties

    public int MyProperty

    {

        get { return myField; }

        set { myField = value; }

    }


    // Methods

    public void MyMethod()
    {
        Console.WriteLine("Executing MyMethod");
    }
}
```

2. Creating Objects (Instances):

Once a class is defined, you can create objects (instances) of that class. Objects represent real entities based on the class blueprint.

```
MyClass myObject = new MyClass();
```

3. Accessing Members:

You can access the members (fields, properties, methods) of a class using the dot notation.

```
myObject.MyProperty = 47;
int value = myObject.MyProperty;

myObject.MyMethod();
```

4. Constructors:

A constructor is a special method used to initialize the object when it is created. It has the same name as the class.

```
public class MyClass
{
    private int myField;

    // Constructor
    public MyClass(int initialValue)
    {
        myField = initialValue;
    }
}
```

}

5. Inheritance:

C# supports inheritance, allowing one class to inherit the members of another. The : base keyword is used to specify the base class.

```
public class DerivedClass : MyBaseClass

{

   // Additional members for the derived class

}
```

6. Encapsulation:

encapsulation is achieved through the use of classes and access modifiers such as public , private , and protected .

Classes provide encapsulation, which means bundling the data (fields) and methods that operate on the data within a single unit. Access modifiers like public, private, and protected control the visibility of class members.

7. Example Usage:

```
class Program

{

   static void Main()

   {

      // Create an instance of MyClass

      MyClass myObject = new MyClass(10);



      // Access members

      myObject.MyMethod();

      Console.WriteLine($"Value: {myObject.MyProperty}");

   }

}
```

Que6. Delegates in C#

}

<u>Delegate :</u> A delegate in C# is a type that refers to methods with a parameter list and return type. Delegates are used to pass methods as arguments to other methods. Use the `interface` keyword to define contracts that any implementing type must support.

1.<u>Delegate Declaration:</u>

To declare a delegate, you specify the method signature it can reference. The delegate keyword is used for this purpose.

```
public delegate void MyDelegate(string message);
```

In the above example, MyDelegate is a delegate that can reference methods taking a string parameter and returning void.

2. <u>Using Delegates:</u>

Once a delegate is declared, you can create an instance of it and associate it with methods that match its signature.

```
public class MyClass

{

   public void Method1(string message)

   {

      Console.WriteLine($"Method1: {message}");

   }


   public void Method2(string message)

   {

      Console.WriteLine($"Method2: {message}");

   }

}


class Program
```

```
{
    static void Main()
    {
        MyClass myObject = new MyClass();

        // Create delegate instances and associate with methods
        MyDelegate delegate1 = myObject.Method1;
        MyDelegate delegate2 = myObject.Method2;

        // Invoke delegates
        delegate1("Hello");
        delegate2("World");
    }
}
```

3. Multicast Delegates:

Delegates can be combined to create a multicast delegate, which can invoke multiple methods.

```
MyDelegate multiDelegate = delegate1 + delegate2;
multiDelegate("Combined invocation");
```

4. Built-in Delegates:

C# provides several built-in generic delegate types in the System namespace, such as Action and Func. These are widely used and eliminate the need to define custom delegates for many scenarios.

```
Action<string> actionDelegate = myObject.Method1;
actionDelegate("Using Action");

Func<int, int, int> addDelegate = (a, b) => a + b;
int result = addDelegate(3, 5);
Console.WriteLine($"Result: {result}");
```

5. **Events:**

Delegates are commonly used to implement events, which provide a mechanism for one object to notify other objects when a specific event occurs.

```
public class Publisher

{

   public event MyDelegate MyEvent;


   public void RaiseEvent(string message)

   {

      MyEvent?.Invoke(message);

   }

}


class Program

{

   static void Main()

   {

      Publisher publisher = new Publisher();

      MyClass subscriber = new MyClass();


      // Subscribe to the event

      publisher.MyEvent += subscriber.Method1;


      // Raise the event

      publisher.RaiseEvent("Event triggered");

   }

}
```

Example :

```
using System;
public delegate double MathOperation(double x, double y);
public class Calculator
{
    public static double Add(double x, double y) => x + y;
    public static double Subtract(double x, double y) => x - y;
    public static double Multiply(double x, double y) => x * y;
    public static double Divide(double x, double y)
    {
        if (y != 0)
            return x / y;
        else
        {
            Console.WriteLine("Cannot divide by zero!");
            return double.NaN; // Not a Number
        }
    }
    public static void Main()
    {
        MathOperation operationDelegate;
        operationDelegate = Add;
        Console.WriteLine($"Addition result: {operationDelegate(5, 3)}");
        operationDelegate = Subtract;
        Console.WriteLine($"Subtraction result: {operationDelegate(5, 3)}");
        operationDelegate = Multiply;
        Console.WriteLine($"Multiplication result: {operationDelegate(5, 3)}");
        operationDelegate = Divide;
        Console.WriteLine($"Division result: {operationDelegate(5, 3)}");
        operationDelegate = Add;
        Console.WriteLine($"Runtime switch - Addition result: {operationDelegate(7, 
2)}");
        operationDelegate = Multiply;
        Console.WriteLine($"Runtime switch - Multiplication result: 
{operationDelegate(7, 2)}");
    }
}
```

Que7. File Handling in C#

Generally, we use the file to store data. The term File Handling in C# refers to the various operations that we can perform on a file such as creating a file, reading data from the file, writing data into the file, appending the file, etc.
Generally, we mostly perform three basic operations on a file: reading data from a file, writing data to a file and appending data to a file.

1. **Reading:** This operation is the basic read operation where the data is going to be read from a file.

2. **Writing:** This operation is the basic write operation where the data is going to be written to a file. By default, all existing contents are removed from the file, and new content is written in the file.

3. **Appending:** This operation is the same as the write operation where the data is going to be written to a file. The only difference between Write and Append is that the existing data in the file will not be overwritten. With Append, the new data is going to be added at the end of the file.

1.Reading from a File:

```
string filePath = "example.txt";

if (File.Exists(filePath))

{

    string[] lines = File.ReadAllLines(filePath);

    // Process lines

}
```

2.Writing to a File:

```
string filePath = "example.txt";

string[] lines = { "Line 1", "Line 2", "Line 3" };

File.WriteAllLines(filePath, lines);
```

3.Appending to a File:

```
string filePath = "example.txt";

string[] newLines = { "New Line 1", "New Line 2" };

File.AppendAllLines(filePath, newLines);
```

4.Reading and Writing Binary Files:

```
string filePath = "binaryfile.bin";



// Writing binary data

byte[] data = { 0x48, 0x65, 0x6C, 0x6C, 0x6F };

File.WriteAllBytes(filePath, data);



// Reading binary data
```

byte[] buffer = File.ReadAllBytes(filePath);

// Process binary data

Example :

```
using System;
using System.IO;

namespace FileHandling1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            string rootPath = @"D:\java";
            var dir = Directory.EnumerateDirectories(rootPath); // to get directories 
from given path
            foreach (var file in dir)
            {
                Console.WriteLine(file);
            }
            var files = Directory.GetFiles(rootPath); // to get file from given path
            foreach (var file in files)
            {
                Console.WriteLine(file);
            }
            //to get all files from directories and given path
            var allFiles = 
Directory.GetFiles(rootPath,"*.*",SearchOption.AllDirectories);
            foreach (var file in allFiles)
            {
                Console.WriteLine(Path.GetFileName(file)); // to get file names
                Console.WriteLine(Path.GetFileNameWithoutExtension(file));
                Console.WriteLine(Path.GetDirectoryName(file));
                var info = new FileInfo(file); // to get complete information related to 
file
                Console.WriteLine(info.Name+" "+info.Length+" Bytes");
            }
            // to check directory is exist or not
            if (Directory.Exists(rootPath))
            {
                Console.WriteLine("Directory is exist");
            }
            else
            {
                Console.WriteLine("Directory is not exist");
            }
            // to copy files from one directory to another Directory
            string SourcePath = @"D:\Source";
            string DestinationPath = @"D:\Destination\";
            foreach (var file in Directory.GetFiles(SourcePath))
            {
                File.Copy(file, DestinationPath + Path.GetFileName(file));
            }
            // to move files from one directory to another Directory
            foreach (var file in Directory.GetFiles(SourcePath))
            {
                File.Move(file, DestinationPath + Path.GetFileName(file));
            }
        }
    }
}
```

Que8. Create a windows form app individual studio with C#

Steps to Create a Windows Forms Application:

1.Open Visual Studio:

- Launch Visual Studio.

2.Create a New Project:

- Go to "File" -> "New" -> "Project..."
- In the "Create a new project" dialog, select "Windows Forms App (.NET Core)" or "Windows Forms App (.NET Framework)" based on your preference and system setup.
- Click "Next."

3.Configure Project:

- Enter a name for your project.
- Choose the location where you want to save the project.
- Set the solution name (optional).
- Choose the framework version (e.g., .NET Core 3.1 or .NET Framework 4.8).
- Click "Create."

4.Design the Form:

- Once the project is created, you'll see the default form (Form1.cs) in the designer.
- You can design your form by dragging and dropping controls from the Toolbox (View -> Toolbox) onto the form.
- Customize the properties of the controls using the Properties window.

5.Add Code to the Form:

- Double-click on a control to create an event handler.
- Add your C# code to handle events and perform actions.
- For example, you can add code to the button click event:

```
private void button1_Click(object sender, EventArgs e)

{

   MessageBox.Show("Hello, Windows Forms!");

}
```

6.Build and Run:

- Build your project by clicking on "Build" -> "Build Solution."

- Run your application by pressing F5 or clicking on the "Start Debugging" button.

That's it! You've created a simple Windows Forms application. You can further enhance your application by adding more controls, implementing additional functionality, and exploring features provided by the Windows Forms framework.

Que9. Assembly in .net

Assembly :

Assembly is an important concept in C#. It is a collection of code files that are compiled into an executable or a Dynamic Link Library (DLL). Depending on their location and intended use, assemblies can be divided into many categories.

Types of Assemblies :

1. **Single-File Assemblies (EXE):** Contains all the necessary information in a single executable file with the .exe extension.

2. **Multi-File Assemblies:** Comprises multiple files, including one main .exe file and accompanying .dll files containing additional code.

Components of an Assembly :

Manifest: Contains metadata about the assembly, such as version information, culture, public key token for strong naming, and a list of files that make up the assembly.

MSIL (Microsoft Intermediate Language) Code: The compiled code that is platform-independent and needs further compilation by the Just-In-Time (JIT) compiler at runtime.

Strong Naming :

Assemblies can be strongly named, which involves signing the assembly with a cryptographic key pair. Strong naming ensures uniqueness and integrity of the assembly.

Private and Shared Assemblies :

Private Assemblies: Used by a single application and stored in the application's directory.

Shared Assemblies (Global Assembly Cache - GAC): Accessible by multiple applications, allowing for code reuse. Shared assemblies are stored in the GAC.

Versioning :

Assemblies support versioning, allowing multiple versions of the same assembly to coexist. This helps in managing updates and ensuring backward compatibility.

Deployment :

Assemblies can be deployed along with the application or shared among multiple applications. Deployment options include XCOPY deployment, ClickOnce deployment, and deployment through installers.

References :

Assemblies can reference other assemblies, and this reference information is stored in the manifest. References help in resolving dependencies between different components.

Reflection :

.NET provides reflection, which allows runtime inspection of the metadata and types within an assembly. This enables dynamic loading and invocation of types.

Global Assembly Cache (GAC) :

The GAC is a machine-wide repository for shared assemblies. Shared assemblies in the GAC are accessible to multiple applications.

Que10. Window based Tic-Tac-Toe Game in C#

1.Create a new Windows Forms Application:

- Open Visual Studio.
- Create a new project: "File" -> "New" -> "Project..."
- Choose "Windows Forms App (.NET Core)" or "Windows Forms App (.NET Framework)" based on your preference.
- Name your project and click "Create."

Source Code :-

```
using System;

using System.Drawing;

using System.Windows.Forms;

namespace TicTacToe

{

  public partial class Form1 : Form

  {

    // Variables

    private bool player1_turn = true;
    private int[,] boardValue = new int[3, 3];

    const int X_VALUE = 1;

    const int O_VALUE = 2;


    public Form1()

    {

       InitializeComponent();

       restart_matrix();

       player1_turn = true;

    }


    // TIC TAC TOE BUTTONS ------------------------------

    private void button1_Click(object sender, EventArgs e)

    {

       if (player1_turn == true)

       {

          player1_turn = false;

          ((Button)sender).Text = "X";

          boardValue[0, 0] = X_VALUE;

       }

       else

       {

          player1_turn = true;

          ((Button)sender).Text = "O";

          boardValue[0, 0] = O_VALUE;
```

```
    }

    check_win();
}

private void button2_Click(object sender, EventArgs e)
{
    if (player1_turn == true)
    {
        player1_turn = false;
        ((Button)sender).Text = "X";
        boardValue[0, 1] = X_VALUE;
    }
    else
    {
        player1_turn = true;
        ((Button)sender).Text = "O";
        boardValue[0, 1] = O_VALUE;
    }
    check_win();
}

private void button3_Click(object sender, EventArgs e)
{
    if (player1_turn == true)
    {
        player1_turn = false;
        ((Button)sender).Text = "X";
        boardValue[0, 2] = X_VALUE;
    }
    else
    {
        player1_turn = true;
        ((Button)sender).Text = "O";
        boardValue[0, 2] = O_VALUE;
    }
    check_win();
}

private void button6_Click(object sender, EventArgs e)
{
    if (player1_turn == true)
    {
        player1_turn = false;
        ((Button)sender).Text = "X";
        boardValue[1, 0] = X_VALUE;
    }
    else
    {
        player1_turn = true;
        ((Button)sender).Text = "O";
        boardValue[1, 0] = O_VALUE;
    }
    check_win();
}

private void button5_Click(object sender, EventArgs e)
{
    if (player1_turn == true)
    {
        player1_turn = false;
        ((Button)sender).Text = "X";
        boardValue[1, 1] = X_VALUE;
    }
    else
    {
        player1_turn = true;
        ((Button)sender).Text = "O";
        boardValue[1, 1] = O_VALUE;
    }
    check_win();
}

private void button4_Click(object sender, EventArgs e)
{
    if (player1_turn == true)

    {

        player1_turn = false;

        ((Button)sender).Text = "X";

        boardValue[1, 2] = X_VALUE;

    }

    else

    {

        player1_turn = true;

        ((Button)sender).Text = "O";

        boardValue[1, 2] = O_VALUE;

    }

    check_win();

}


private void button9_Click(object sender, EventArgs e)

{

    if (player1_turn == true)

    {

        player1_turn = false;

        ((Button)sender).Text = "X";

        boardValue[2, 0] = X_VALUE;

    }

    else

    {
        player1_turn = true;
        ((Button)sender).Text = "O";
        boardValue[2, 0] = O_VALUE;
    }
    check_win();
}

private void button8_Click(object sender, EventArgs e)
{
    if (player1_turn == true)
    {
        player1_turn = false;
        ((Button)sender).Text = "X";
        boardValue[2, 1] = X_VALUE;
    }
    else
    {
        player1_turn = true;
        ((Button)sender).Text = "O";
        boardValue[2, 1] = O_VALUE;
    }
    check_win();
}

private void button7_Click(object sender, EventArgs e)
```

```
{
    if (player1_turn == true)
    {
        player1_turn = false;
        ((Button)sender).Text = "X";
        boardValue[2, 2] = X_VALUE;
    }
    else
    {
        player1_turn = true;
        ((Button)sender).Text = "O";
        boardValue[2, 2] = O_VALUE;
    }
    check_win();
}
// TIC TAC TOE BUTTONS -------------------------------


// TIC TAC TOE MATRIX
private void check_win()
{
    // Check rows
    for (int i = 0; i < 3; ++i)
    {
        if (Convert.ToInt32(boardValue[i, 0]) == Convert.ToInt32(boardValue[i, 1]) &&
            Convert.ToInt32(boardValue[i, 1]) == Convert.ToInt32(boardValue[i, 2]) &&
            Convert.ToInt32(boardValue[i, 1]) != 0)
        {
            color_winning_buttons(find_button_by_value(i, 0),
                find_button_by_value(i, 1),
                find_button_by_value(i, 2));
            display_win_screen(boardValue[i, 0]);

            return;
        }
    }

    // Check columns
    for (int i = 0; i < 3; ++i)
    {
        if (Convert.ToInt32(boardValue[0, i]) == Convert.ToInt32(boardValue[1, i]) &&
            Convert.ToInt32(boardValue[1, i]) == Convert.ToInt32(boardValue[2, i]) &&
            Convert.ToInt32(boardValue[2, i]) != 0)
        {
            color_winning_buttons(find_button_by_value(0, i),
                find_button_by_value(1, i),
                find_button_by_value(2, i));
            display_win_screen(boardValue[1, i]);
            return;
        }
    }
```

```
// Check diagonals
if (Convert.ToInt32(boardValue[0, 0]) == Convert.ToInt32(boardValue[1, 1]) &&
  Convert.ToInt32(boardValue[1, 1]) == Convert.ToInt32(boardValue[2, 2]) &&
  Convert.ToInt32(boardValue[2, 2]) != 0)
{
  color_winning_buttons(find_button_by_value(0, 0),
    find_button_by_value(1, 1),
    find_button_by_value(2, 2));
  display_win_screen(boardValue[1, 1]);
  return;
}
else if (Convert.ToInt32(boardValue[0, 2]) == Convert.ToInt32(boardValue[1, 1]) &&
 Convert.ToInt32(boardValue[1, 1]) == Convert.ToInt32(boardValue[2, 0]) &&
 Convert.ToInt32(boardValue[2, 0]) != 0)
{
  color_winning_buttons(find_button_by_value(0, 2),
    find_button_by_value(1, 1),
    find_button_by_value(2, 0));
  display_win_screen(boardValue[1, 1]);
  return;
}

// Check draw
bool is_completed = true;
```

```
    for (int i = 0; i < 3; ++i)
        for (int j = 0; j < 3; j++)
            if (boardValue[i, j] == 0)
                is_completed = false;
    if (is_completed)
    {
        MessageBox.Show("Draw");
        restart_matrix();
    }
}

// Display win message
private void display_win_screen(int val)
{
    if (val == X_VALUE)
    {
        MessageBox.Show("Player1 wins!");
        int player1_score = Convert.ToInt32(label6.Text);
        ++player1_score;
        label6.Text = player1_score.ToString();
    }
    else
    {
        MessageBox.Show("Player2 wins!");
        int player2_score = Convert.ToInt32(label7.Text);
        ++player2_score;

        label7.Text = player2_score.ToString();

    }


    restart_matrix();

}


// Restart tic tac toe board

private void restart_matrix()

{

    for (int i = 0; i < 3; ++i)

        for (int j = 0; j < 3; ++j)

            boardValue[i, j] = 0;


    clear_buttons();

    player1_turn = true;

}


// Color winning buttons

private void color_winning_buttons(Button button1, Button button2, Button button3)

{

    button1.BackColor = Color.Red;

    button2.BackColor = Color.Red;

    button3.BackColor = Color.Red;

}
```

```
// Clear buttons
private void clear_buttons()
{
    // Content
    button1.Text = "";
    button2.Text = "";
    button3.Text = "";
    button4.Text = "";
    button5.Text = "";
    button6.Text = "";
    button7.Text = "";
    button8.Text = "";
    button9.Text = "";

    // Back color
    button1.BackColor = Color.Green;
    button2.BackColor = Color.Green;
    button3.BackColor = Color.Green;
    button4.BackColor = Color.Green;
    button5.BackColor = Color.Green;
    button6.BackColor = Color.Green;
    button7.BackColor = Color.Green;
    button8.BackColor = Color.Green;
    button9.BackColor = Color.Green;
    // Fore color

    button1.ForeColor = Color.White;

    button2.ForeColor = Color.White;

    button3.ForeColor = Color.White;

    button4.ForeColor = Color.White;

    button5.ForeColor = Color.White;

    button6.ForeColor = Color.White;

    button7.ForeColor = Color.White;

    button8.ForeColor = Color.White;

    button9.ForeColor = Color.White;

}


// Find a button by it's value

private Button find_button_by_value(int value_x, int value_y)

{

    switch (value_x)

    {

        case 0:

            if (value_y == 0)

                return button1;

            if (value_y == 1)

                return button2;

            if (value_y == 2)

                return button3;
            return button1;
            break;
        case 1:
            if (value_y == 0)
                return button6;
            if (value_y == 1)
                return button5;
            if (value_y == 2)
                return button4;
            return button1;
            break;
        case 2:
            if (value_y == 0)
                return button9;
            if (value_y == 1)
                return button8;
            if (value_y == 2)
                return button7;
            return button1;
            break;

        default:
            restart_matrix();
            MessageBox.Show("Something failed!");
            return button1;
```

```
            }
        }

        // Select active player
        private void timer1_Tick(object sender, EventArgs e)
        {
            if (player1_turn == true)
            {
                label1.ForeColor = Color.Green;
                label2.ForeColor = Color.Black;
            }
            else
            {
                label2.ForeColor = Color.Green;
                label1.ForeColor = Color.Black;
            }
        }

        // Reset game
        private void button10_Click(object sender, EventArgs e)
        {
            Application.Restart();
        }
    }
}
```

```
namespace TicTacToe

{

    partial class Form1

    {

        /// <summary>

        /// Required designer variable.

        /// </summary>

        private System.ComponentModel.IContainer components = null;


        /// <summary>

        /// Clean up any resources being used.

        /// </summary>

        /// <param name="disposing">true if managed resources should be disposed; otherwise,
false.</param>

        protected override void Dispose(bool disposing)

        {

            if (disposing && (components != null))

            {

                components.Dispose();

            }

            base.Dispose(disposing);

        }


        #region Windows Form Designer generated code
```

```
/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

    this.components = new System.ComponentModel.Container();

    this.button3 = new System.Windows.Forms.Button();

    this.button1 = new System.Windows.Forms.Button();

    this.button2 = new System.Windows.Forms.Button();

    this.button4 = new System.Windows.Forms.Button();

    this.button5 = new System.Windows.Forms.Button();

    this.button6 = new System.Windows.Forms.Button();

    this.button7 = new System.Windows.Forms.Button();

    this.button8 = new System.Windows.Forms.Button();

    this.button9 = new System.Windows.Forms.Button();

    this.TITLE = new System.Windows.Forms.Label();

    this.label1 = new System.Windows.Forms.Label();

    this.label2 = new System.Windows.Forms.Label();

    this.label3 = new System.Windows.Forms.Label();

    this.label4 = new System.Windows.Forms.Label();

    this.label5 = new System.Windows.Forms.Label();

    this.label6 = new System.Windows.Forms.Label();

    this.label7 = new System.Windows.Forms.Label();
```

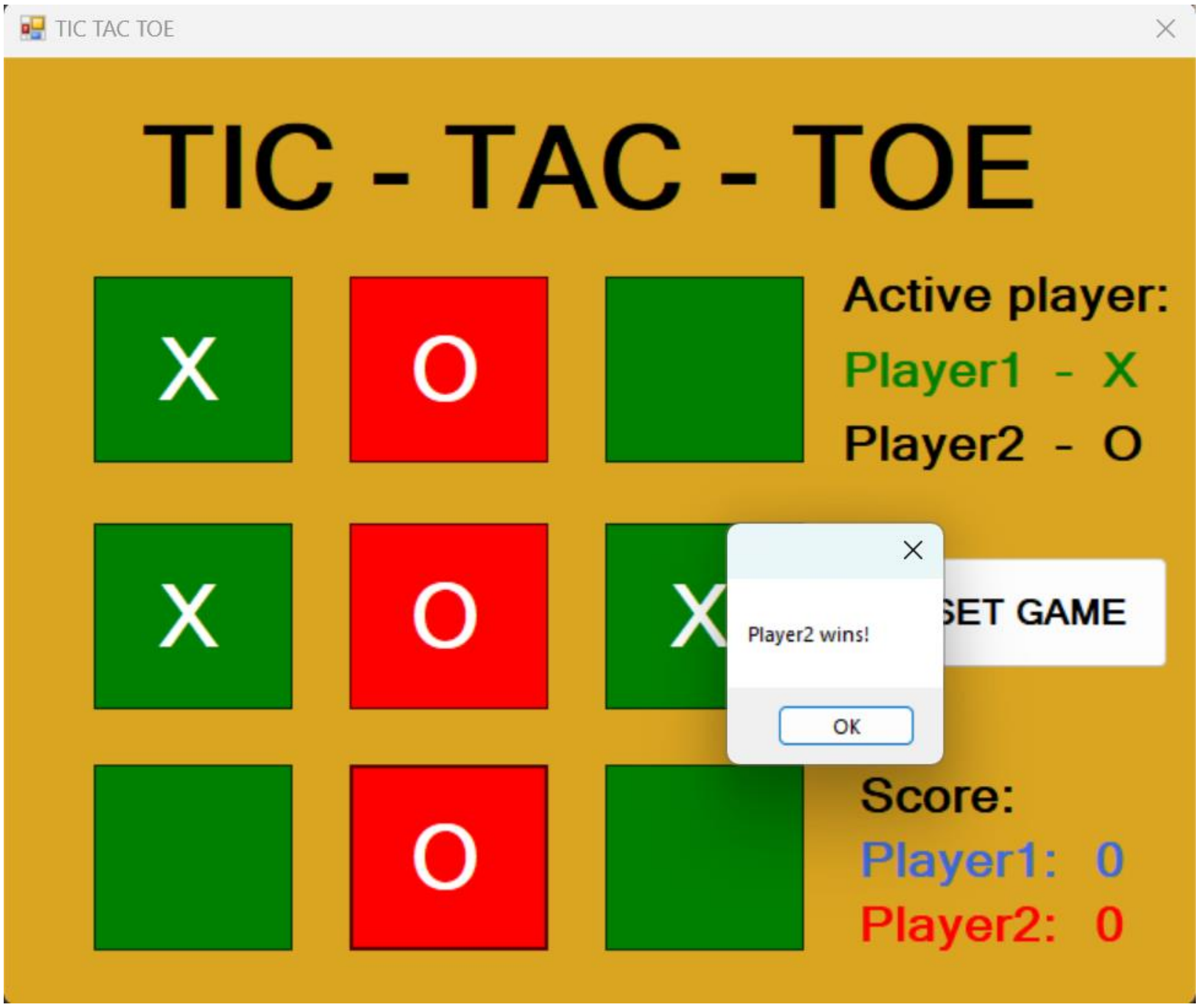
```
            this.timer1 = new System.Windows.Forms.Timer(this.components);

            this.label8 = new System.Windows.Forms.Label();

            this.button10 = new System.Windows.Forms.Button();

            this.SuspendLayout();

            //

            // button3

            //

            this.button3.BackColor = System.Drawing.Color.Silver;

            this.button3.FlatStyle = System.Windows.Forms.FlatStyle.Popup;

            this.button3.Font = new System.Drawing.Font("Microsoft Sans Serif", 36F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.button3.Location = new System.Drawing.Point(327, 119);

            this.button3.Name = "button3";

            this.button3.Size = new System.Drawing.Size(108, 101);

            this.button3.TabIndex = 2;

            this.button3.UseVisualStyleBackColor = false;

            this.button3.Click += new System.EventHandler(this.button3_Click);

            //

            // button1

            //

            this.button1.BackColor = System.Drawing.Color.Silver;

            this.button1.FlatStyle = System.Windows.Forms.FlatStyle.Popup;

            this.button1.Font = new System.Drawing.Font("Microsoft Sans Serif", 36F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.button1.Location = new System.Drawing.Point(49, 119);

            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(108, 101);

            this.button1.TabIndex = 0;

            this.button1.UseVisualStyleBackColor = false;

            this.button1.Click += new System.EventHandler(this.button1_Click);

            //

            // button2

            //

            this.button2.BackColor = System.Drawing.Color.Silver;

            this.button2.FlatStyle = System.Windows.Forms.FlatStyle.Popup;

            this.button2.Font = new System.Drawing.Font("Microsoft Sans Serif", 36F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.button2.Location = new System.Drawing.Point(188, 119);

            this.button2.Name = "button2";

            this.button2.Size = new System.Drawing.Size(108, 101);

            this.button2.TabIndex = 1;

            this.button2.UseVisualStyleBackColor = false;

            this.button2.Click += new System.EventHandler(this.button2_Click);

            //

            // button4

            //

            this.button4.BackColor = System.Drawing.Color.Silver;

            this.button4.FlatStyle = System.Windows.Forms.FlatStyle.Popup;

            this.button4.Font = new System.Drawing.Font("Microsoft Sans Serif", 36F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.button4.Location = new System.Drawing.Point(327, 253);

            this.button4.Name = "button4";
            this.button4.Size = new System.Drawing.Size(108, 101);

            this.button4.TabIndex = 5;

            this.button4.UseVisualStyleBackColor = false;

            this.button4.Click += new System.EventHandler(this.button4_Click);

            //

            // button5

            //

            this.button5.BackColor = System.Drawing.Color.Silver;

            this.button5.FlatStyle = System.Windows.Forms.FlatStyle.Popup;

            this.button5.Font = new System.Drawing.Font("Microsoft Sans Serif", 36F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.button5.Location = new System.Drawing.Point(188, 253);

            this.button5.Name = "button5";

            this.button5.Size = new System.Drawing.Size(108, 101);

            this.button5.TabIndex = 4;

            this.button5.UseVisualStyleBackColor = false;

            this.button5.Click += new System.EventHandler(this.button5_Click);

            //

            // button6

            //

            this.button6.BackColor = System.Drawing.Color.Silver;

            this.button6.FlatStyle = System.Windows.Forms.FlatStyle.Popup;

            this.button6.Font = new System.Drawing.Font("Microsoft Sans Serif", 36F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.button6.Location = new System.Drawing.Point(49, 253);

            this.button6.Name = "button6";
            this.button6.Size = new System.Drawing.Size(108, 101);

            this.button6.TabIndex = 3;

            this.button6.UseVisualStyleBackColor = false;

            this.button6.Click += new System.EventHandler(this.button6_Click);

            //

            // button7

            //

            this.button7.BackColor = System.Drawing.Color.Silver;

            this.button7.FlatStyle = System.Windows.Forms.FlatStyle.Popup;

            this.button7.Font = new System.Drawing.Font("Microsoft Sans Serif", 36F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.button7.Location = new System.Drawing.Point(327, 384);

            this.button7.Name = "button7";

            this.button7.Size = new System.Drawing.Size(108, 101);

            this.button7.TabIndex = 8;

            this.button7.UseVisualStyleBackColor = false;

            this.button7.Click += new System.EventHandler(this.button7_Click);

            //

            // button8

            //

            this.button8.BackColor = System.Drawing.Color.Silver;

            this.button8.FlatStyle = System.Windows.Forms.FlatStyle.Popup;

            this.button8.Font = new System.Drawing.Font("Microsoft Sans Serif", 36F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.button8.Location = new System.Drawing.Point(188, 384);

            this.button8.Name = "button8";
            this.button8.Size = new System.Drawing.Size(108, 101);
            this.button8.TabIndex = 7;
            this.button8.UseVisualStyleBackColor = false;
            this.button8.Click += new System.EventHandler(this.button8_Click);
            //
            // button9
            //
            this.button9.BackColor = System.Drawing.Color.Silver;
            this.button9.FlatStyle = System.Windows.Forms.FlatStyle.Popup;
            this.button9.Font = new System.Drawing.Font("Microsoft Sans Serif", 36F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.button9.Location = new System.Drawing.Point(49, 384);
            this.button9.Name = "button9";
            this.button9.Size = new System.Drawing.Size(108, 101);
            this.button9.TabIndex = 6;
            this.button9.UseVisualStyleBackColor = false;
            this.button9.Click += new System.EventHandler(this.button9_Click);
            //
            // TITLE
            //
            this.TITLE.AutoSize = true;
            this.TITLE.Font = new System.Drawing.Font("Microsoft Sans Serif", 48F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.TITLE.Location = new System.Drawing.Point(62, 23);
            this.TITLE.Name = "TITLE";
            this.TITLE.Size = new System.Drawing.Size(520, 73);
            this.TITLE.TabIndex = 9;

            this.TITLE.Text = "TIC - TAC - TOE";

            //

            // label1

            //

            this.label1.AutoSize = true;

            this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.label1.Location = new System.Drawing.Point(450, 153);

            this.label1.Name = "label1";

            this.label1.Size = new System.Drawing.Size(174, 31);

            this.label1.TabIndex = 10;

            this.label1.Text = "Player1  -  X";

            //

            // label2

            //

            this.label2.AutoSize = true;

            this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.label2.Location = new System.Drawing.Point(450, 193);

            this.label2.Name = "label2";

            this.label2.Size = new System.Drawing.Size(177, 31);

            this.label2.TabIndex = 11;

            this.label2.Text = "Player2  -  O";

            //

            // label3
            //
            this.label3.AutoSize = true;
            this.label3.Font = new System.Drawing.Font("Microsoft Sans Serif", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label3.Location = new System.Drawing.Point(459, 384);
            this.label3.Name = "label3";
            this.label3.Size = new System.Drawing.Size(99, 31);
            this.label3.TabIndex = 12;
            this.label3.Text = "Score:";
            //
            // label4
            //
            this.label4.AutoSize = true;
            this.label4.Font = new System.Drawing.Font("Microsoft Sans Serif", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label4.ForeColor = System.Drawing.Color.Red;
            this.label4.Location = new System.Drawing.Point(459, 454);
            this.label4.Name = "label4";
            this.label4.Size = new System.Drawing.Size(122, 31);
            this.label4.TabIndex = 14;
            this.label4.Text = "Player2:";
            //
            // label5
            //
            this.label5.AutoSize = true;
            this.label5.Font = new System.Drawing.Font("Microsoft Sans Serif", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
```

```
            this.label5.ForeColor = System.Drawing.Color.RoyalBlue;

            this.label5.Location = new System.Drawing.Point(459, 419);

            this.label5.Name = "label5";

            this.label5.Size = new System.Drawing.Size(122, 31);

            this.label5.TabIndex = 13;

            this.label5.Text = "Player1:";

            //

            // label6

            //

            this.label6.AutoSize = true;

            this.label6.Font = new System.Drawing.Font("Microsoft Sans Serif", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.label6.ForeColor = System.Drawing.Color.RoyalBlue;

            this.label6.Location = new System.Drawing.Point(587, 419);

            this.label6.Name = "label6";

            this.label6.Size = new System.Drawing.Size(30, 31);

            this.label6.TabIndex = 15;

            this.label6.Text = "0";

            //

            // label7

            //

            this.label7.AutoSize = true;

            this.label7.Font = new System.Drawing.Font("Microsoft Sans Serif", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.label7.ForeColor = System.Drawing.Color.Red;

            this.label7.Location = new System.Drawing.Point(587, 454);
            this.label7.Name = "label7";

            this.label7.Size = new System.Drawing.Size(30, 31);

            this.label7.TabIndex = 16;

            this.label7.Text = "0";

            //

            // timer1

            //

            this.timer1.Enabled = true;

            this.timer1.Tick += new System.EventHandler(this.timer1_Tick);

            //

            // label8

            //

            this.label8.AutoSize = true;

            this.label8.Font = new System.Drawing.Font("Microsoft Sans Serif", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.label8.Location = new System.Drawing.Point(450, 112);

            this.label8.Name = "label8";

            this.label8.Size = new System.Drawing.Size(192, 31);

            this.label8.TabIndex = 17;

            this.label8.Text = "Active player:";

            //

            // button10

            //

            this.button10.Font = new System.Drawing.Font("Microsoft Sans Serif", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

            this.button10.Location = new System.Drawing.Point(456, 271);
            this.button10.Name = "button10";

            this.button10.Size = new System.Drawing.Size(177, 61);

            this.button10.TabIndex = 18;

            this.button10.Text = "RESET GAME";

            this.button10.UseVisualStyleBackColor = true;

            this.button10.Click += new System.EventHandler(this.button10_Click);

            //

            // Form1

            //

            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

            this.BackColor = System.Drawing.Color.Goldenrod;

            this.ClientSize = new System.Drawing.Size(648, 515);

            this.Controls.Add(this.button10);

            this.Controls.Add(this.label8);

            this.Controls.Add(this.label7);

            this.Controls.Add(this.label6);

            this.Controls.Add(this.label4);

            this.Controls.Add(this.label5);

            this.Controls.Add(this.label3);

            this.Controls.Add(this.label2);

            this.Controls.Add(this.label1);

            this.Controls.Add(this.TITLE);

            this.Controls.Add(this.button7);

            this.Controls.Add(this.button8);
```

```
    this.Controls.Add(this.button9);

    this.Controls.Add(this.button4);

    this.Controls.Add(this.button5);

    this.Controls.Add(this.button6);

    this.Controls.Add(this.button3);

    this.Controls.Add(this.button2);

    this.Controls.Add(this.button1);

    this.MaximizeBox = false;

    this.MinimizeBox = false;

    this.Name = "Form1";

    this.Text = "TIC TAC TOE";

    this.ResumeLayout(false);

    this.PerformLayout();


}


#endregion


private System.Windows.Forms.Button button3;

private System.Windows.Forms.Button button1;

private System.Windows.Forms.Button button2;

private System.Windows.Forms.Button button4;

private System.Windows.Forms.Button button5;

private System.Windows.Forms.Button button6;

private System.Windows.Forms.Button button7;
```

```
        private System.Windows.Forms.Button button8;

        private System.Windows.Forms.Button button9;

        private System.Windows.Forms.Label TITLE;

        private System.Windows.Forms.Label label1;

        private System.Windows.Forms.Label label2;

        private System.Windows.Forms.Label label3;

        private System.Windows.Forms.Label label4;

        private System.Windows.Forms.Label label5;

        private System.Windows.Forms.Label label6;

        private System.Windows.Forms.Label label7;

        private System.Windows.Forms.Timer timer1;

        private System.Windows.Forms.Label label8;

        private System.Windows.Forms.Button button10;

    }

}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;
using TicTacToe;

namespace TicTacToe
{
    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());
        }
    }
}
```



Player2 wins!