Bộ môn An toàn Thông tin – Khoa MMT&TT

Trường Đại học Công nghệ Thông tin (UIT)

LAB REPORT

Subject: Basic network programming

(Session 02)

Topic name: Lab 2

Group: Ningguang

DETAILED REPORT

1. Scenario 01

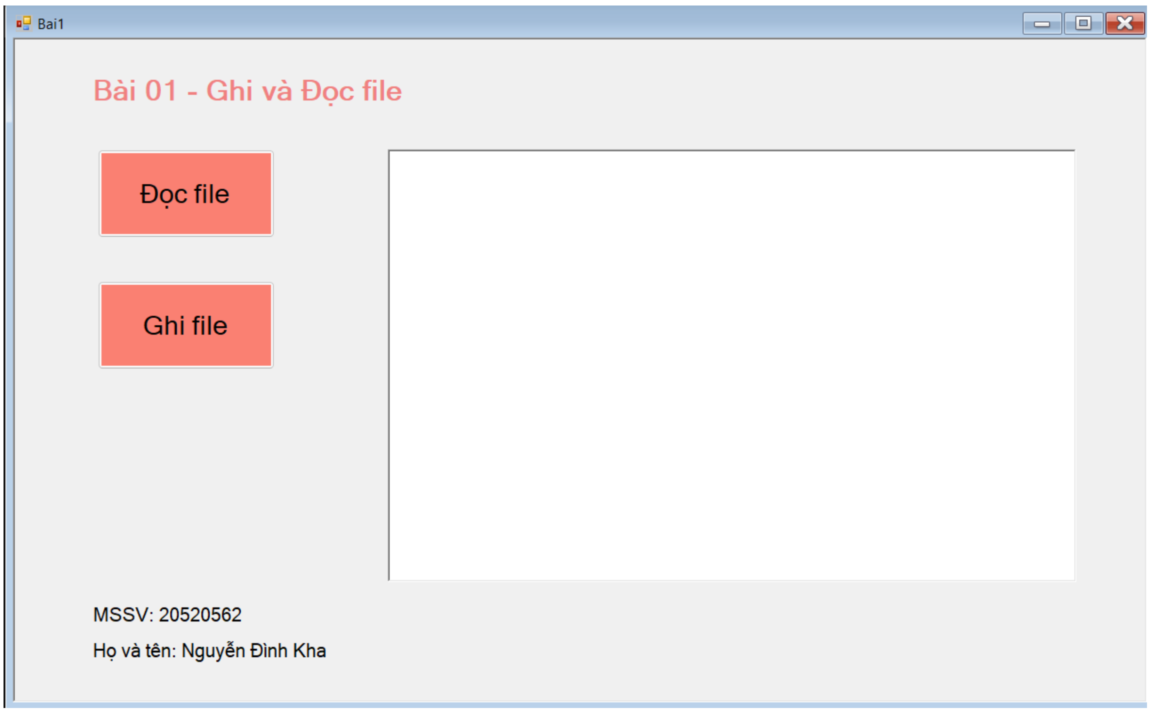
- **Objective:**

Write a program to read the content of an “input.txt” file and display it on the screen. Then write the content (convert all characters to uppercase) to an “output.txt” file

- **Steps:**

- **Step 1: Design the Program Interface**

The interface includes two buttons: one for opening and reading a file ("Read File" button) and another for saving content to a file ("Write File" button). Additionally, a RichTextBox (or similar text display component) is used to show the content of the file.

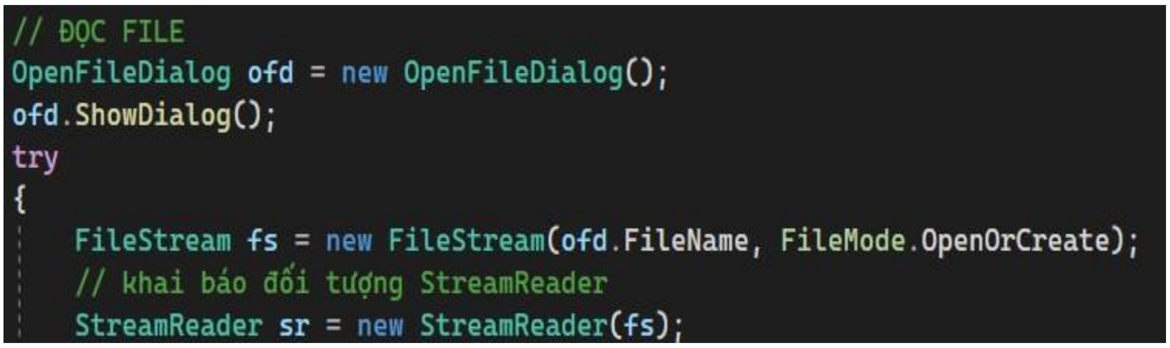


MSSV: 20520562
Họ và tên: Nguyễn Đình Kha

- **Step 2: Implementing the Code for the Buttons**

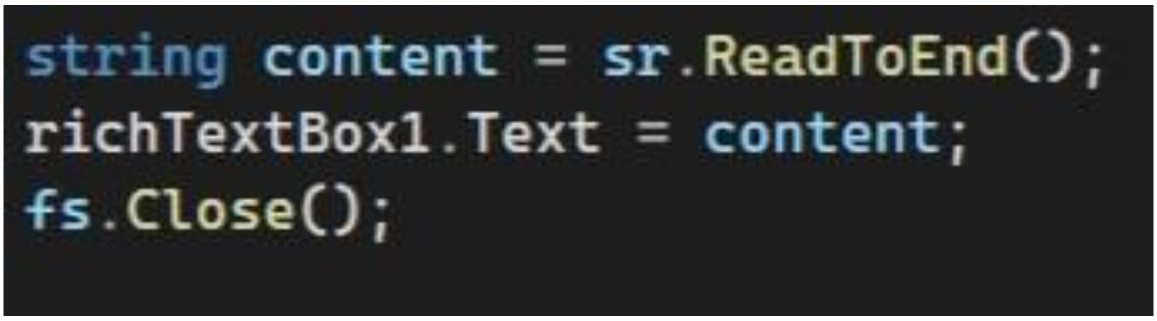
+ Read File Button:

• Utilise an OpenFileDialog to browse and select the file you want to read. Use the FileStream class to handle the file data and the StreamReader class to read the file.



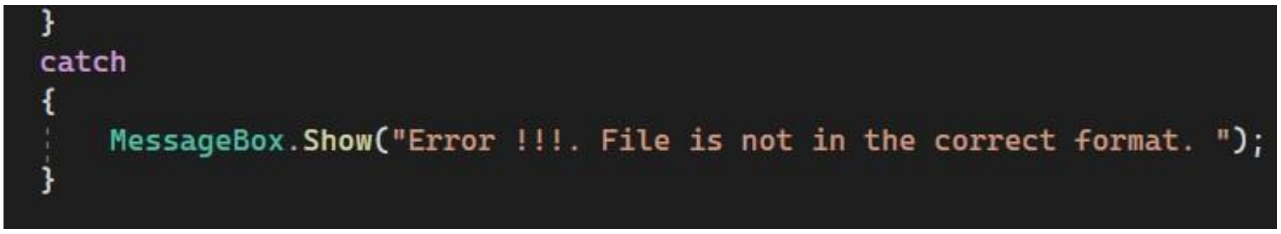
```
// ĐỌC FILE
OpenFileDialog ofd = new OpenFileDialog();
ofd.ShowDialog();
try
{
    FileStream fs = new FileStream(ofd.FileName, FileMode.OpenOrCreate);
    // khai báo đối tượng StreamReader
    StreamReader sr = new StreamReader(fs);
```

- Employ the ReadToEnd method of StreamReader to read all the data from the file and display it in the RichTextBox.



```
string content = sr.ReadToEnd();
richTextBox1.Text = content;
fs.Close();
```

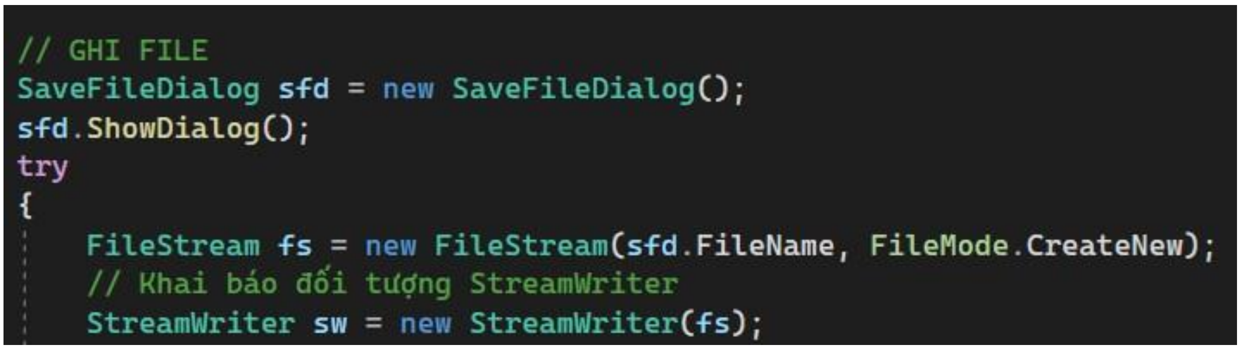
- Implement try-catch blocks to handle potential errors, such as the file not being in the correct format or not found.



```
}
catch
{
    MessageBox.Show("Error !!!. File is not in the correct format. ");
}
```

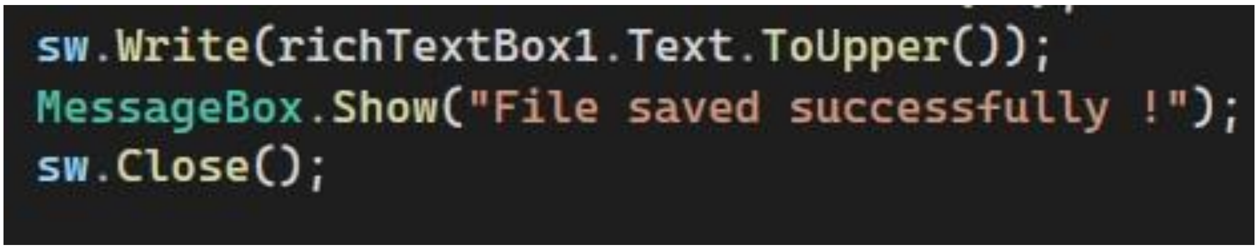
+ Write File Button:

- Use a SaveFileDialog to choose the location and name of the file to save. Like with reading, employ FileStream for handling file data but use StreamWriter for writing content to the file.



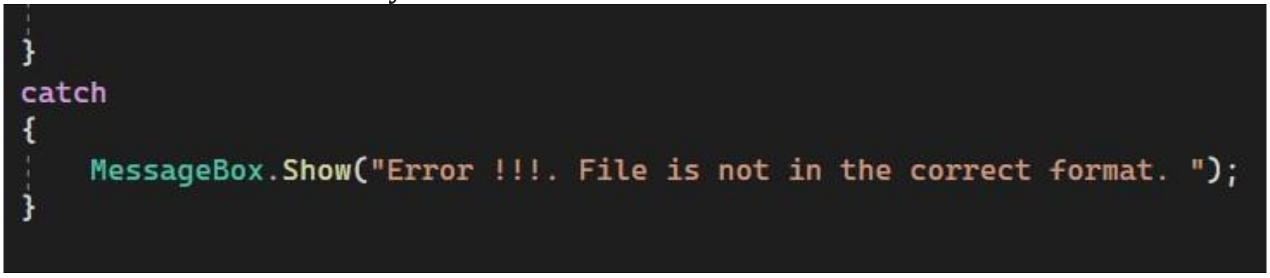
```
// GHI FILE
SaveFileDialog sfd = new SaveFileDialog();
sfd.ShowDialog();
try
{
    FileStream fs = new FileStream(sfd.FileName, FileMode.CreateNew);
    // Khai báo đối tượng StreamWriter
    StreamWriter sw = new StreamWriter(fs);
```

• The content displayed in the RichTextBox is to be written into the file. Use the Write method of StreamWriter, and apply the ToUpper method to convert all characters to uppercase before writing, as per the requirement. Utilize MessageBox to notify the user once the file has been successfully saved.



```
sw.Write(richTextBox1.Text.ToUpper());
MessageBox.Show("File saved successfully !");
sw.Close();
```

• Implement try-catch blocks for error handling during the file writing process, similar to the read functionality.



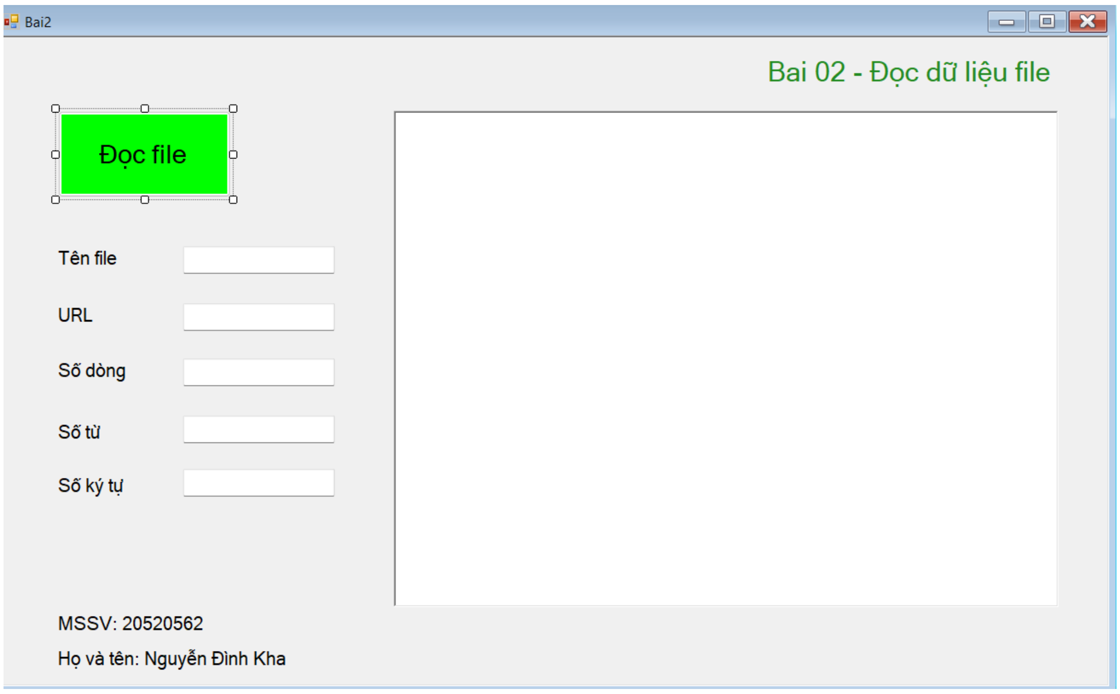
```
}
catch
{
    MessageBox.Show("Error !!!. File is not in the correct format. ");
}
```

2. **Scenario 02**
   - **Resource:** Attached compressed file
   - **Objective:**
     Writing a program that reads a file and displays the following information:
   - File name
   - URL path
   - Number of lines, words, and characters
   - Content of the file

   - **Steps to implement:**

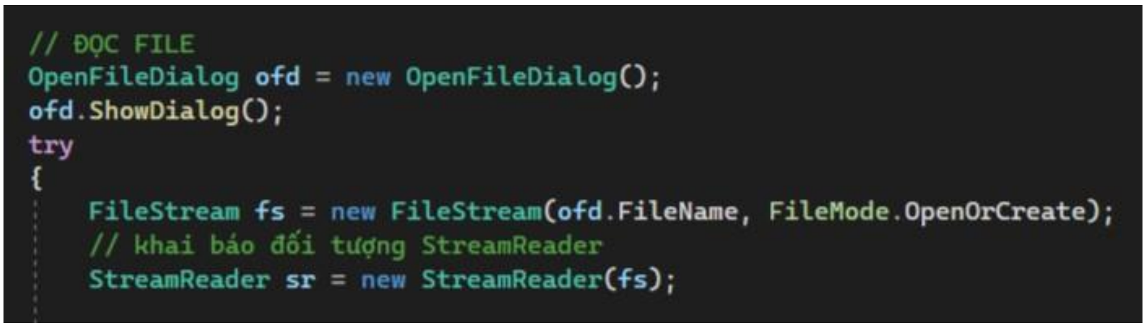
- Step 1: Design the Interface

The interface should include a "Read File" button that triggers the file reading function, five textboxes to display the file name, URL path, number of lines, number of words, number of characters, respectively, and a RichTextBox to show the content of the file.



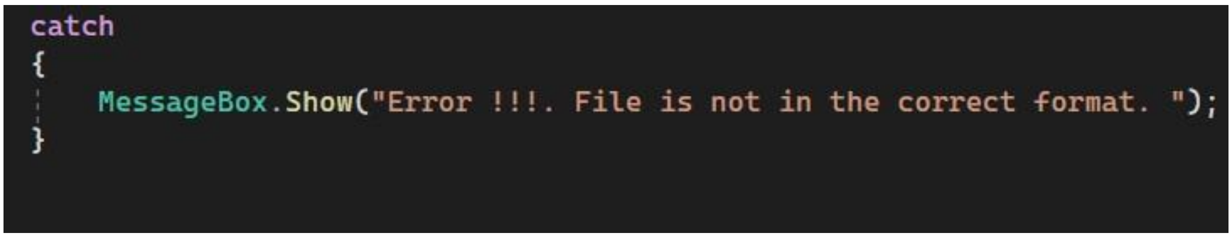
- Step 2: Implement the "Read File" Button Functionality

+ Use an OpenFileDialog to allow the user to navigate to and select the file to be read. Employ the FileStream class for reading data from the file and the StreamReader class for processing the file's content.



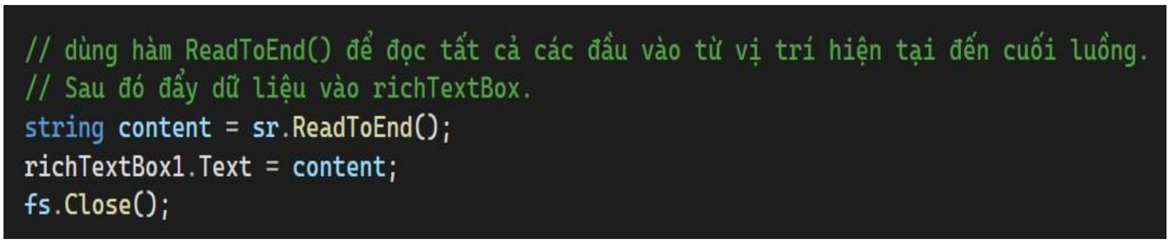
```
// ĐỌC FILE
OpenFileDialog ofd = new OpenFileDialog();
ofd.ShowDialog();
try
{
    FileStream fs = new FileStream(ofd.FileName, FileMode.OpenOrCreate);
    // khai báo đối tượng StreamReader
    StreamReader sr = new StreamReader(fs);
```

+ Implement try-catch blocks to handle any errors that might occur if the file is not in the expected format or cannot be found.



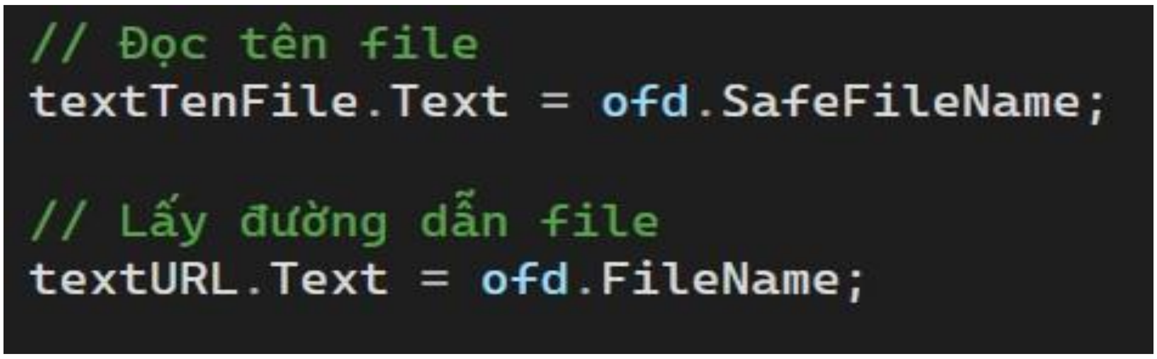
```
catch
{
    MessageBox.Show("Error !!!. File is not in the correct format. ");
}
```

+ Utilize the ReadToEnd method of StreamReader to read all the file's data and display it in the RichTextBox.



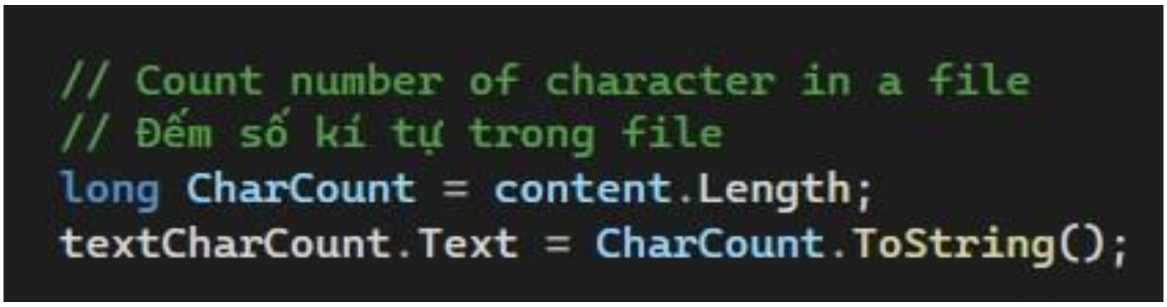
```
// dùng hàm ReadToEnd() để đọc tất cả các đầu vào từ vị trí hiện tại đến cuối luồng.
// Sau đó đẩy dữ liệu vào richTextBox.
string content = sr.ReadToEnd();
richTextBox1.Text = content;
fs.Close();
```

+ Retrieve the file name using the .SafeFileName property of OpenFileDialog and the file's full path using the .FileName property of FileStream.



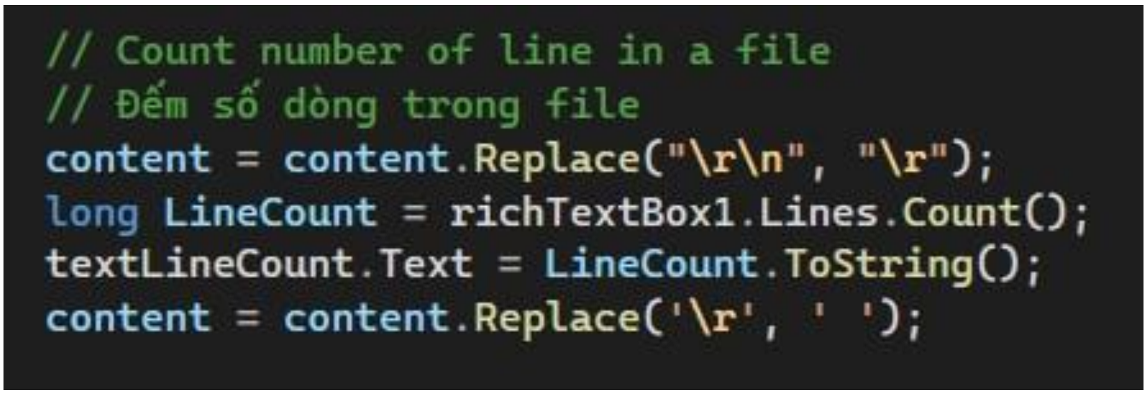
```
// Đọc tên file
textTenFile.Text = ofd.SafeFileName;

// Lấy đường dẫn file
textURL.Text = ofd.FileName;
```

+ Count the number of characters, lines, and words in the file. Then, assign these values to the respective textboxes to display the results.Đếm số kí tự trong file:



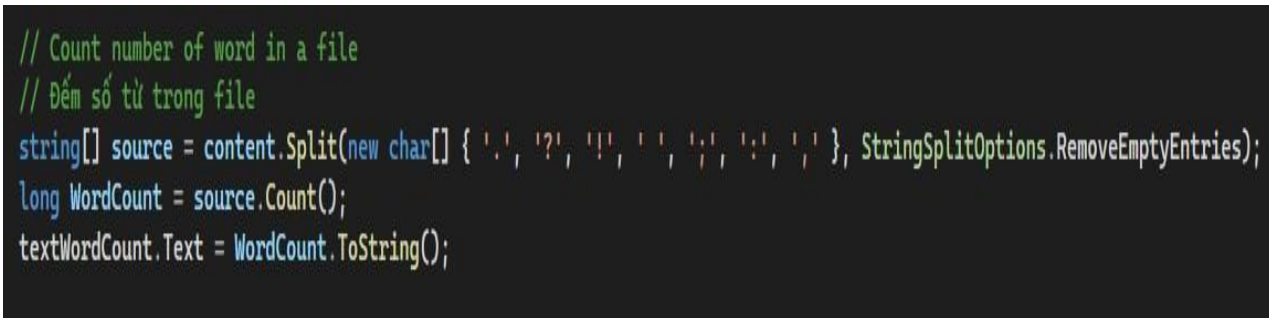
```
// Count number of character in a file
// Đếm số kí tự trong file
long CharCount = content.Length;
textCharCount.Text = CharCount.ToString();
```

- Count number of lines in a file:



```
// Count number of line in a file
// Đếm số dòng trong file
content = content.Replace("\r\n", "\r");
long LineCount = richTextBox1.Lines.Count();
textLineCount.Text = LineCount.ToString();
content = content.Replace('\r', ' ');
```

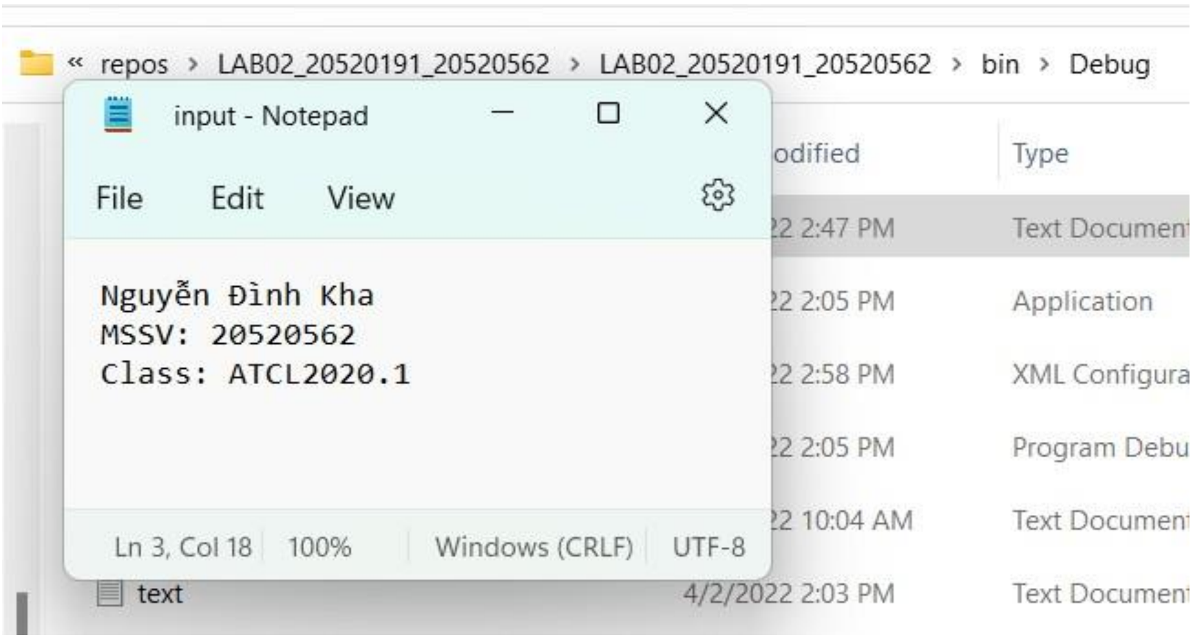
- Count number of word in a file:



```
// Count number of word in a file
// Đếm số từ trong file
string[] source = content.Split(new char[] { '.', '?', '!', ' ', ';', ':', ',' }, StringSplitOptions.RemoveEmptyEntries);
long WordCount = source.Count();
textWordCount.Text = WordCount.ToString();
```

- Step 03: Demonstration

+ Use an example input.txt file for the demo:



« repos › LAB02_20520191_20520562 › LAB02_20520191_20520562 › bin › Debug

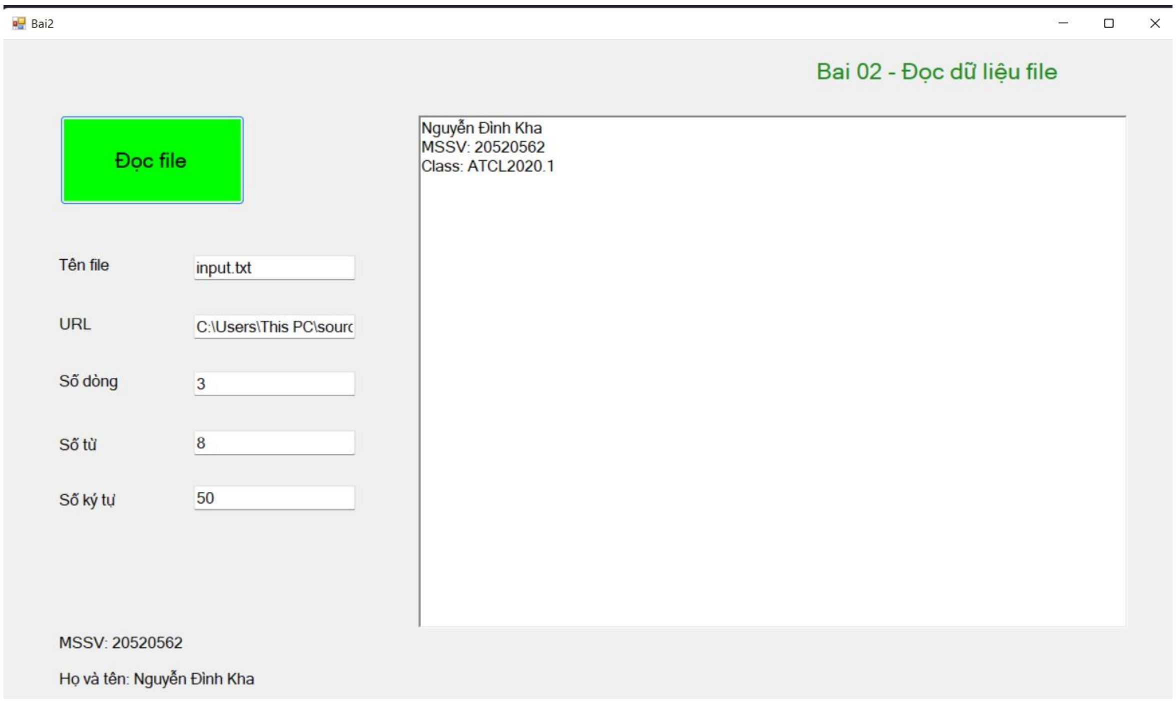
input - Notepad

```
Nguyễn Đình Kha
MSSV: 20520562
Class: ATCL2020.1
```

Ln 3, Col 18 | 100% | Windows (CRLF) | UTF-8

	odified	Type
	22 2:47 PM	Text Document
	22 2:05 PM	Application
	22 2:58 PM	XML Configura
	22 2:05 PM	Program Debu
	22 10:04 AM	Text Document
text	4/2/2022 2:03 PM	Text Document

+ The content of the file will be displayed in the RichTextBox, and the file name, URL path, number of words, characters, and lines will be shown in the respective textboxes following the execution of the program:



Nguyễn Đình Kha
MSSV: 20520562
Class: ATCL2020.1

Tên file: input.txt
URL: C:\Users\This PC\sourc
Số dòng: 3
Số từ: 8
Số ký tự: 50

MSSV: 20520562
Họ và tên: Nguyễn Đình Kha

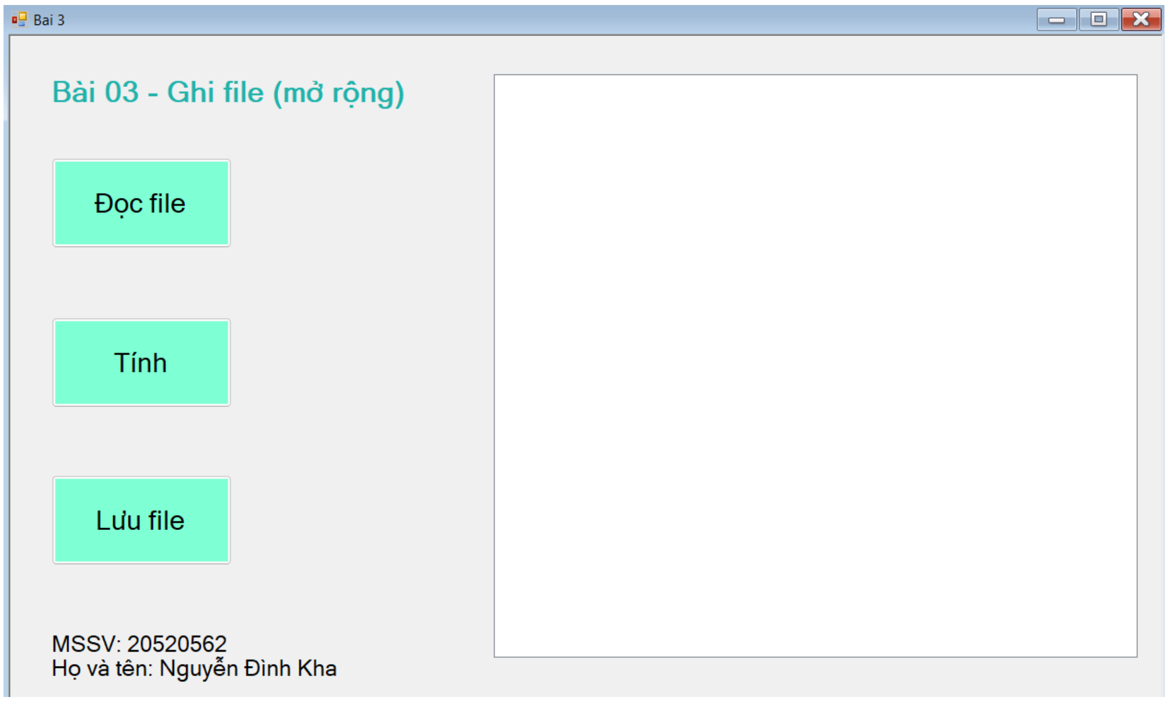
3. Scenario 03

- **Resource:**
- **Objective:** Read content from a file named "input.txt" which is formatted in a specific way, perform calculations as specified, and write the results to a file named "output.txt".

- **Steps to implement:**

- **Step 1: Design the Program Interface**

The interface should include three buttons for reading the file, performing calculations, and saving the file, respectively, as required by the task. Additionally, a RichTextBox should be used to display the content of the file before and after processing the calculations.

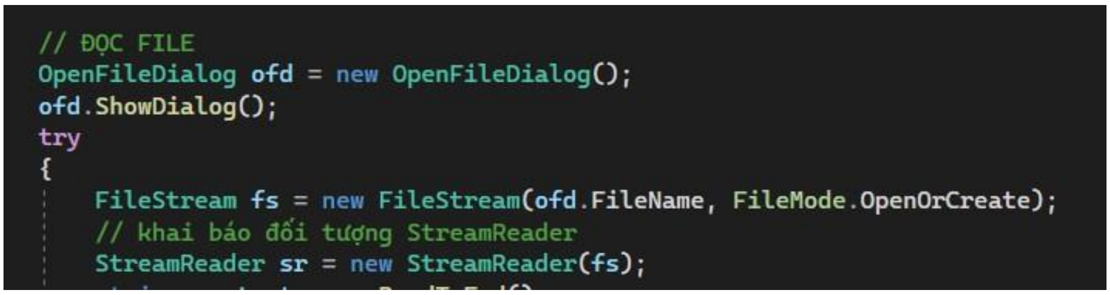


MSSV: 20520562
Họ và tên: Nguyễn Đình Kha

- **Step 2: Implement the Functionality for Each Button**

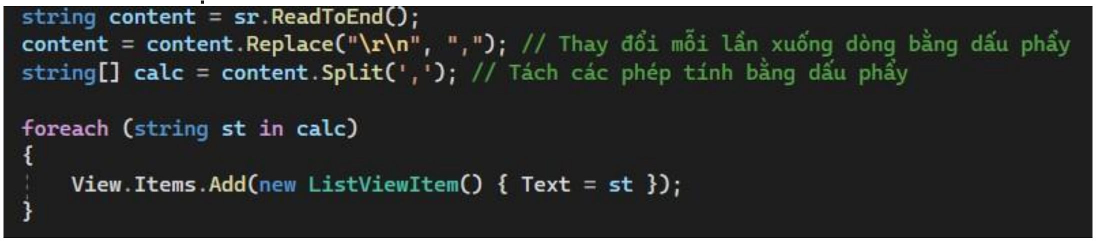
+ Read File Button:

Use an OpenFileDialog to navigate to and select the file for reading. Employ FileStream for reading data from the file and StreamReader for interpreting the file's data.



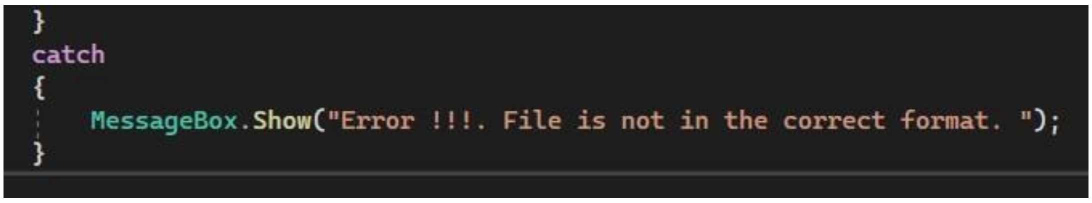
```
// ĐỌC FILE
OpenFileDialog ofd = new OpenFileDialog();
ofd.ShowDialog();
try
{
    FileStream fs = new FileStream(ofd.FileName, FileMode.OpenOrCreate);
    // khai báo đối tượng StreamReader
    StreamReader sr = new StreamReader(fs);
```

Read all data using the ReadToEnd method and display it in a ListView, initializing ListViewItem for each piece of data. Dùng hàm ReadToEnd để đọc toàn bộ dữ liệu và đẩy dữ liệu vào listView. Tiến hành code giống như yêu cầu đề bài. Sau đó sử dụng foreach và khởi tạo ListViewItem



```
string content = sr.ReadToEnd();
content = content.Replace("\r\n", ","); // Thay đổi mỗi lần xuống dòng bằng dấu phẩy
string[] calc = content.Split(','); // Tách các phép tính bằng dấu phẩy

foreach (string st in calc)
{
    View.Items.Add(new ListViewItem() { Text = st });
}
```

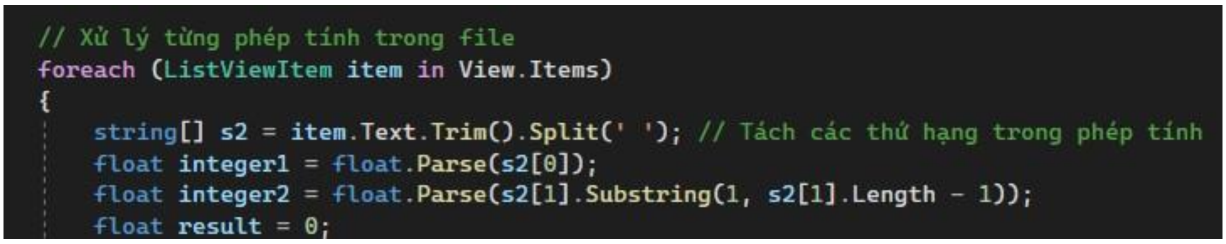
- Implement try-catch blocks to handle potential errors in file format or reading process.



```
}
catch
{
    MessageBox.Show("Error !!!. File is not in the correct format. ");
}
```

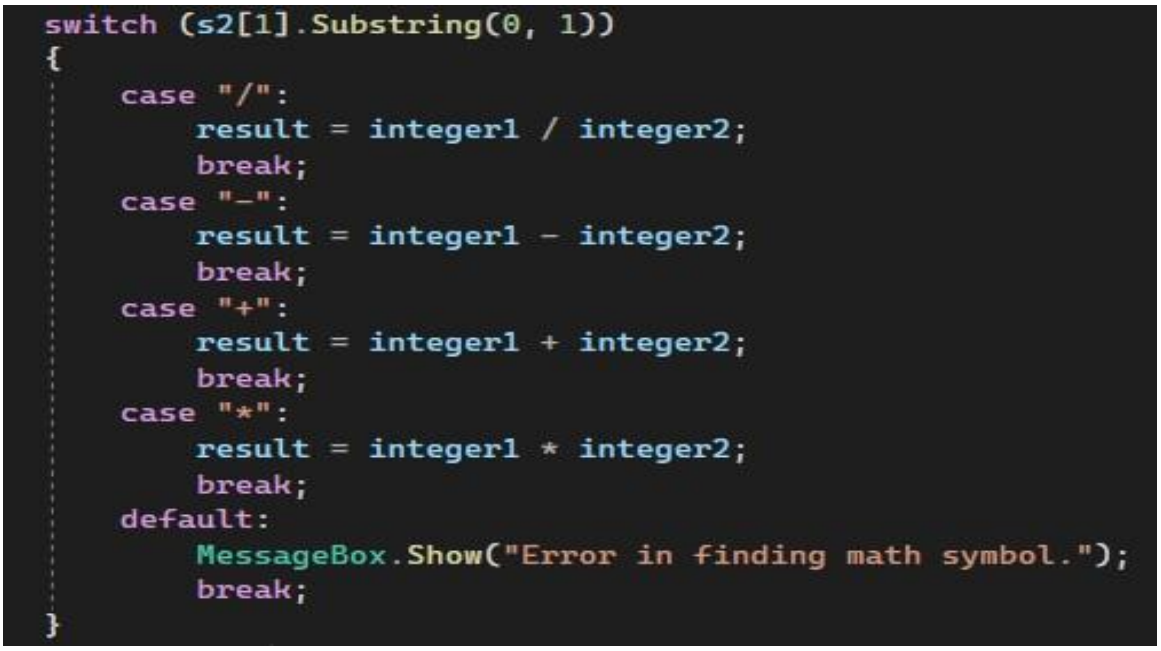
+ Calculate Button:

• Process each calculation in the file using the .Split() method to separate the operands in the calculations, declaring two integer variables for the operands found in the "input.txt" file to perform the calculations, and initializing a result variable to store the outcome of each calculation.



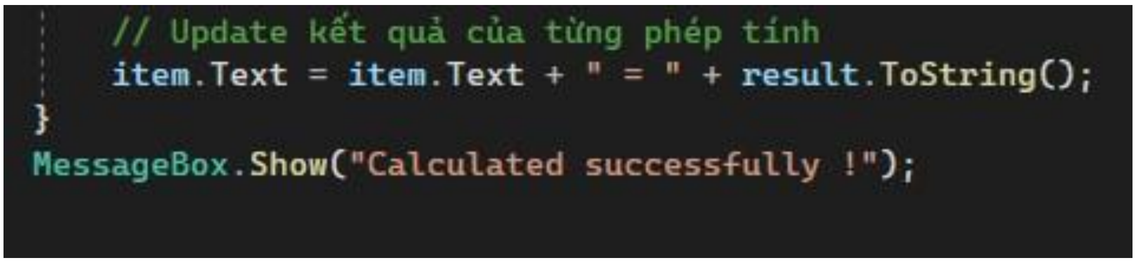
```
// Xử lý từng phép tính trong file
foreach (ListViewItem item in View.Items)
{
    string[] s2 = item.Text.Trim().Split(' '); // Tách các thứ hạng trong phép tính
    float integer1 = float.Parse(s2[0]);
    float integer2 = float.Parse(s2[1].Substring(1, s2[1].Length - 1));
    float result = 0;
```

• Employ a switch-case structure to handle different mathematical operations as specified by each line in "input.txt". Use a MessageBox to alert if an appropriate mathematical operator is not found.



```
switch (s2[1].Substring(0, 1))
{
    case "/":
        result = integer1 / integer2;
        break;
    case "-":
        result = integer1 - integer2;
        break;
    case "+":
        result = integer1 + integer2;
        break;
    case "*":
        result = integer1 * integer2;
        break;
    default:
        MessageBox.Show("Error in finding math symbol.");
        break;
}
```

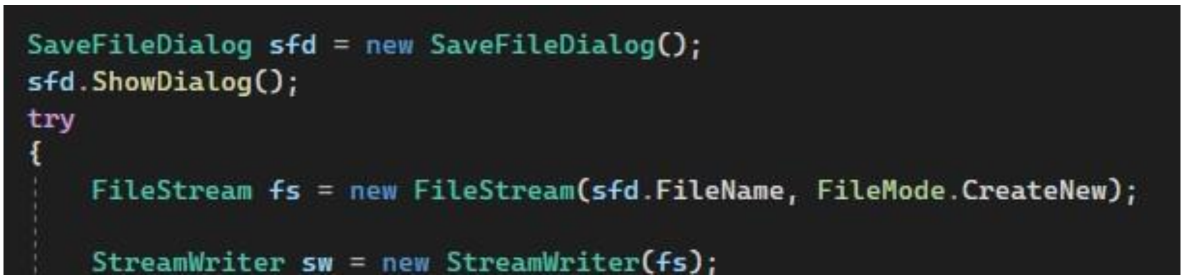
• Update the result of each calculation and save it to the "output.txt" file, using a MessageBox to notify when the calculation process is complete.



```
        // Update kết quả của từng phép tính
        item.Text = item.Text + " = " + result.ToString();
    }
    MessageBox.Show("Calculated successfully !");
```

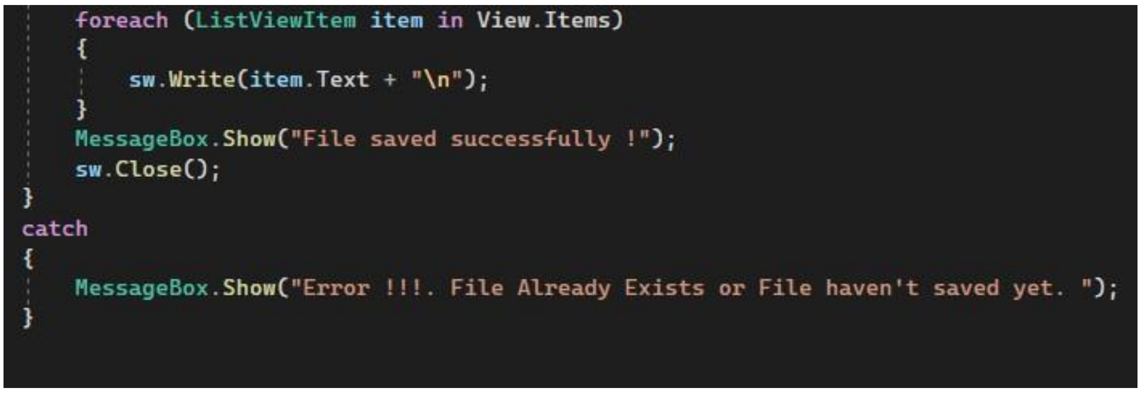
+ Save File Button:

Use a SaveFileDialog to select the location and name of the file when saving. Use FileStream for accessing the data and StreamWriter for writing the content to the file.



```
SaveFileDialog sfd = new SaveFileDialog();
sfd.ShowDialog();
try
{
    FileStream fs = new FileStream(sfd.FileName, FileMode.CreateNew);

    StreamWriter sw = new StreamWriter(fs);
```

• Iterate through each item in the ListView using foreach, and display a MessageBox to confirm successful file saving.

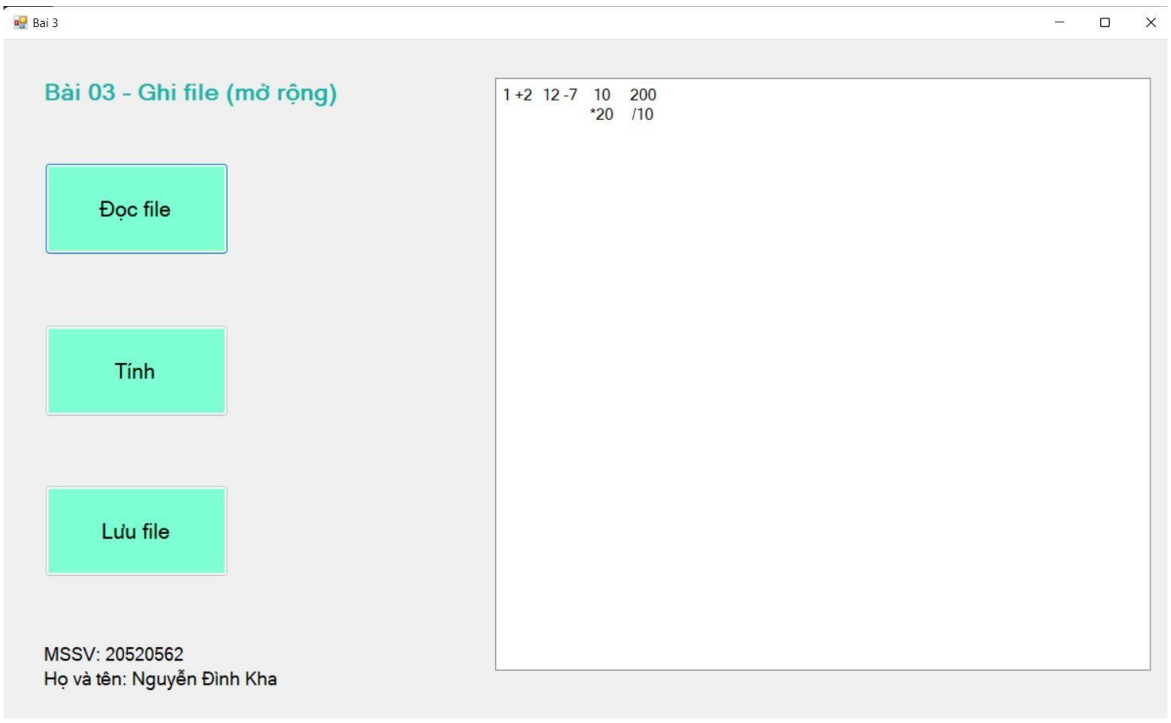


```
    foreach (ListViewItem item in View.Items)
    {
        sw.Write(item.Text + "\n");
    }
    MessageBox.Show("File saved successfully !");
    sw.Close();
}
catch
{
    MessageBox.Show("Error !!!. File Already Exists or File haven't saved yet. ");
}
```

• Implement try-catch blocks to handle potential errors in file saving or format issues. Alert with a MessageBox if the file already exists or has not been saved correctly.

- **Step 3: Demonstration**

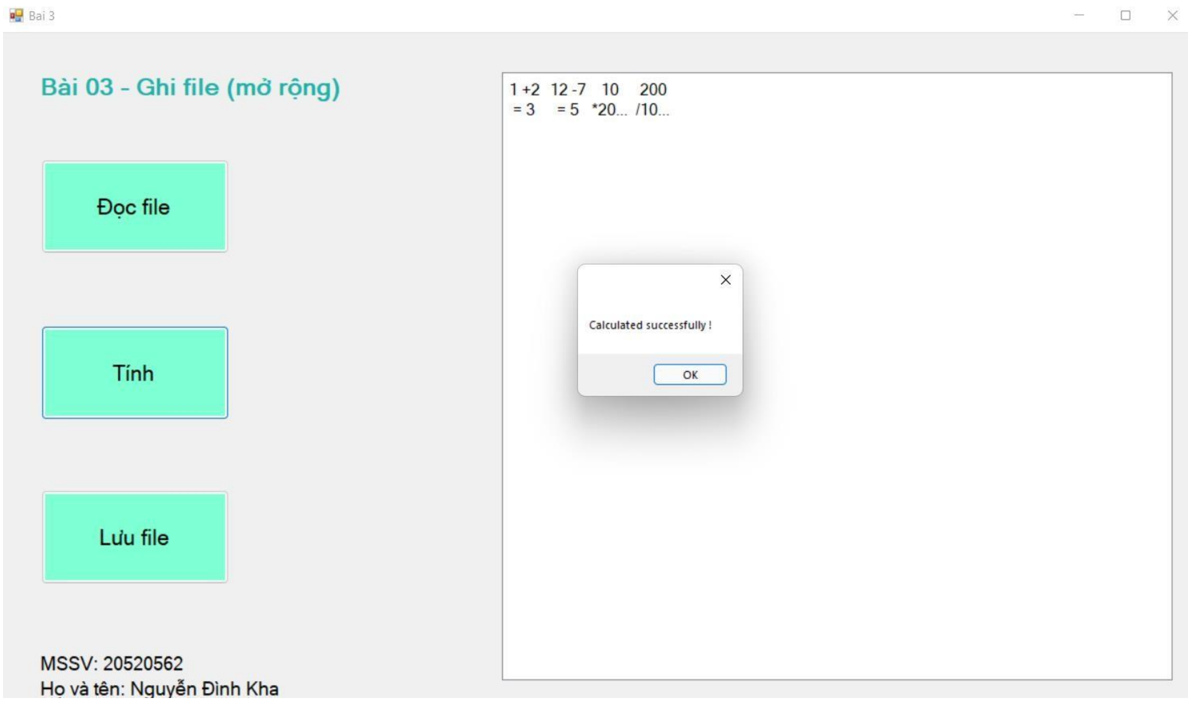
+ The file contents post-reading will be displayed on the right side.



MSSV: 20520562
Họ và tên: Nguyễn Đình Kha

```
1 +2  12 -7   10    200
               *20   /10
```

+ After performing the calculations, the results will be displayed along with a notification message indicating "Successful calculation!".

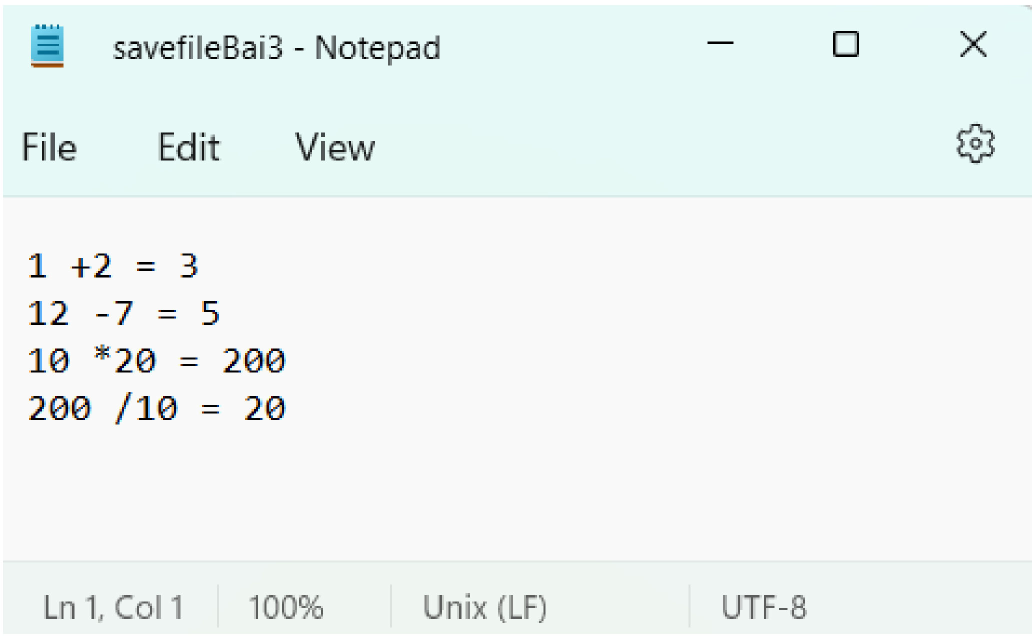


MSSV: 20520562
Họ và tên: Nguyễn Đình Kha

```
1 +2  12 -7   10    200
 = 3   = 5  *20...  /10...
```

Calculated successfully !

+ The file named "savefileBai3.txt" will be saved upon program completion, showcasing the results of the operations performed during the demonstration.



```
1 +2 = 3
12 -7 = 5
10 *20 = 200
200 /10 = 20
```

4. **Scenario 04**
   - **Resource:** Attached compressed file
   - **Objective:**

     Write a program using BinaryFormatter which allows for the input of various student details and saves this data to a file named input.txt. The structure of the student details includes:

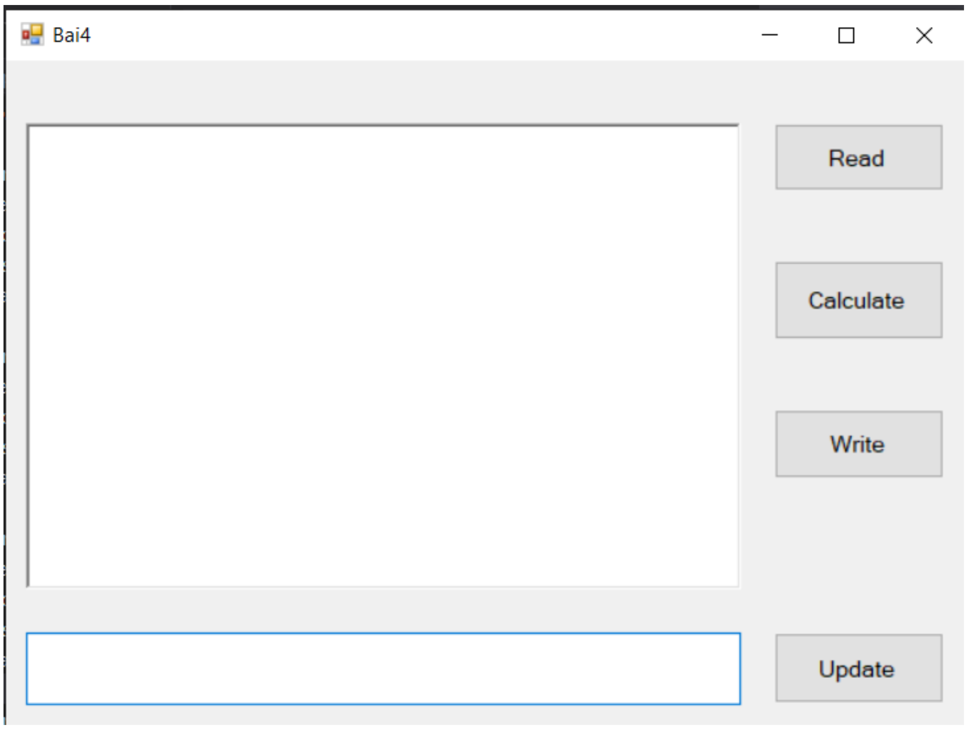
- MSSV (Student ID): String
- HoTen (Full Name): String
- DienThoai (Phone Number): String
- DiemToan (Math Grade): float
- DiemVan (Literature Grade): float
- DTB (GPA): float

The task is to read the student information from the input.txt file, calculate the average grade (presumably as GPA), save the data to an output.txt file, and display it on the screen.

Steps to implement:

- **Step 1: Design the Program Interface**

Interface Components: The design includes a text box for user input, where each piece of student information is separated by commas, and each student record is delimited by semicolons. There are also four buttons to perform various functions and a rich text box to display information.



- **Step 2: Code Functionality for Buttons**

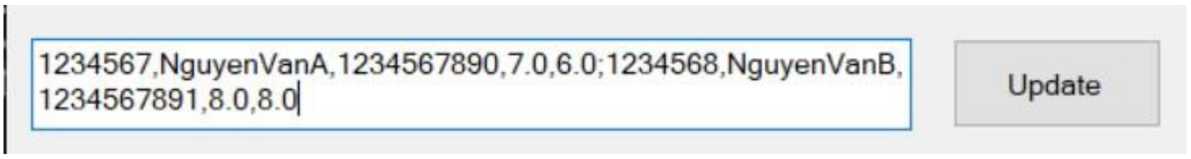
Update Button: This button is used to save the user-entered data into a new file using serialization, after processing the input with split and replace functions.

Read Button: Reads information from a selected file using deserialization and displays it on the screen.

Calculate Button: Calculates and displays the average score for each student.

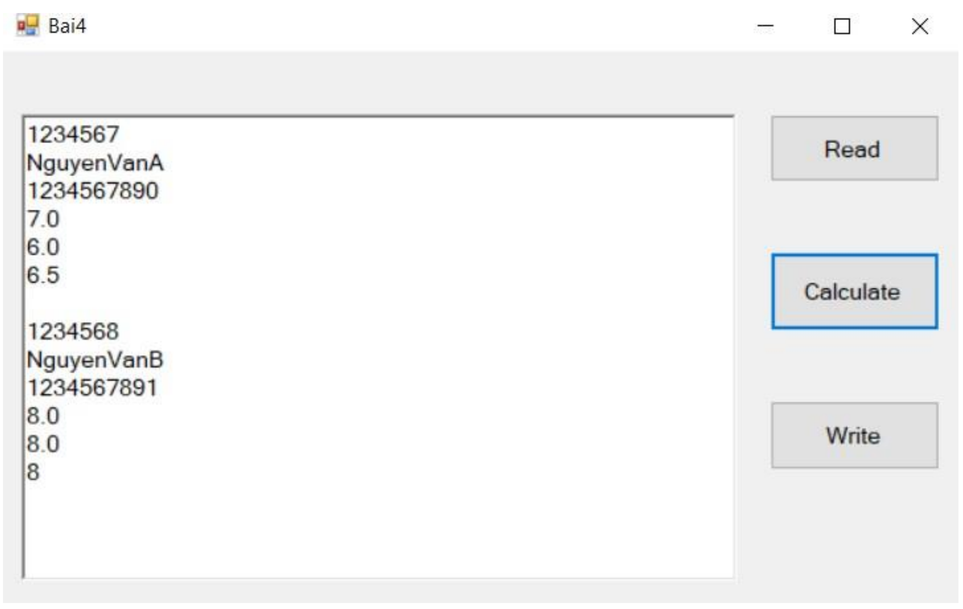
Write Button: Saves the information displayed in the rich text box into a new file using serialization.

Step 3: Demonstration
The process involves entering input data, using the Update button to save it into an "input.txt" file



1234567,NguyenVanA,1234567890,7.0,6.0;1234568,NguyenVanB,1234567891,8.0,8.0

Reading the content of "input.txt" with the Read button, calculating the average scores of students by pressing the Calculate button, and finally, saving the displayed information into an "output.txt" file using the Write button



```
1234567
NguyenVanA
1234567890
7.0
6.0
6.5

1234568
NguyenVanB
1234567891
8.0
8.0
8
```

5. Scenario 05

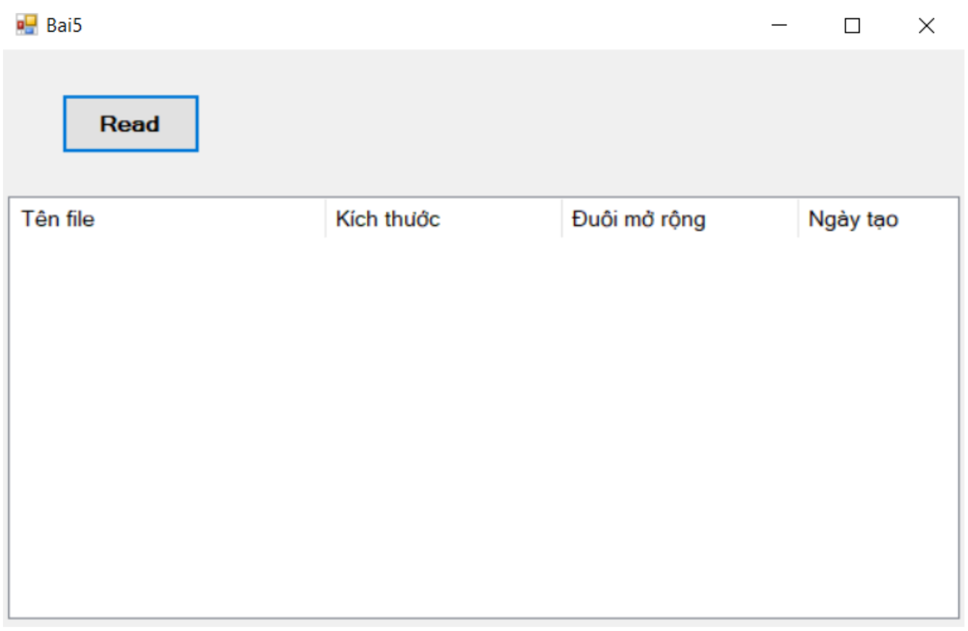
- **Resource:**

- **Description / Objective:**
  Write an application that allows browsing all the files in a directory, displaying a list with details for each file, which includes:

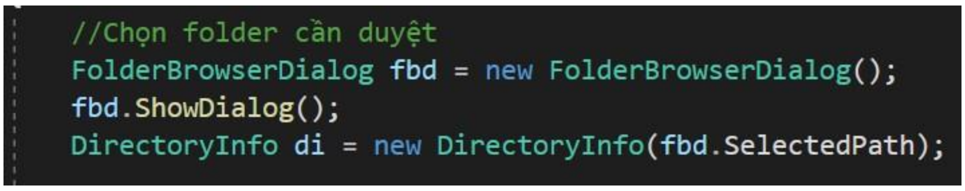
- File name
- Size
- Date created
- File extension
-

- **Steps to implement:**

- **Step 1:** Design a WinForms interface that includes a button to browse folders and a ListView control with pre-defined columns to display the required information.



Step 2: Code the functionality for the 'Read' button (to read information about the files in the selected folder). Initially, display a dialog that allows the user to select a folder to browse.

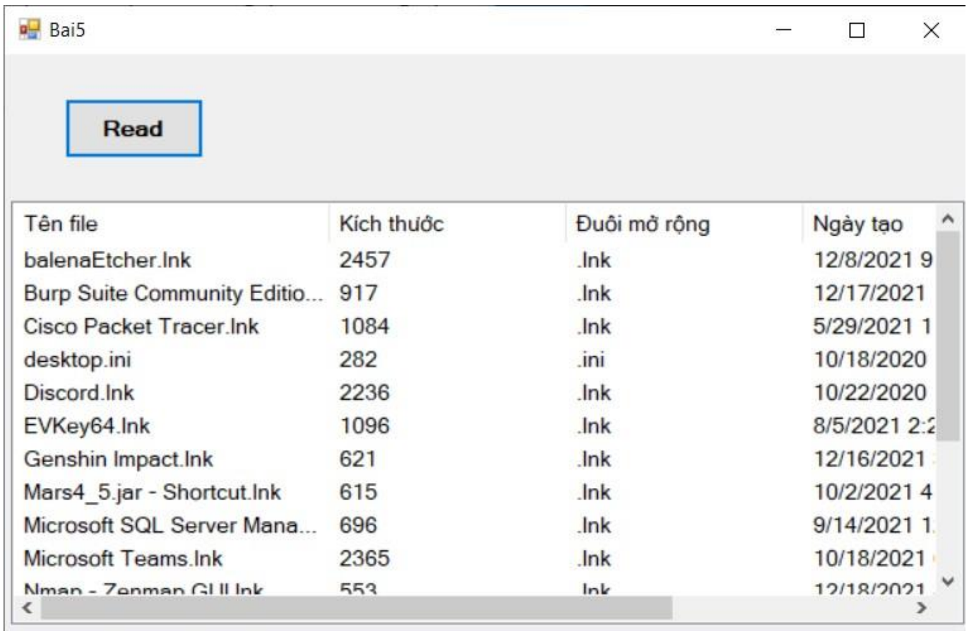


```
//Chọn folder cần duyệt
FolderBrowserDialog fbd = new FolderBrowserDialog();
fbd.ShowDialog();
DirectoryInfo di = new DirectoryInfo(fbd.SelectedPath);
```

After selecting a folder, retrieve the information of each file within it using the GetFiles() method and display the information (as requested in the specifications) in the ListView.

- **Step 3: Demonstration**

Press the 'Read' button to choose the directory you want to browse.
The information displayed will include the file name, size, extension, and creation date of the files in the selected directory.



Tên file	Kích thước	Đuôi mở rộng	Ngày tạo
balenaEtcher.lnk	2457	.lnk	12/8/2021 9
Burp Suite Community Editio...	917	.lnk	12/17/2021
Cisco Packet Tracer.lnk	1084	.lnk	5/29/2021 1
desktop.ini	282	.ini	10/18/2020
Discord.lnk	2236	.lnk	10/22/2020
EVKey64.lnk	1096	.lnk	8/5/2021 2:2
Genshin Impact.lnk	621	.lnk	12/16/2021
Mars4_5.jar - Shortcut.lnk	615	.lnk	10/2/2021 4
Microsoft SQL Server Mana...	696	.lnk	9/14/2021 1
Microsoft Teams.lnk	2365	.lnk	10/18/2021
Nmap - Zenmap GUI.lnk	553	.lnk	12/18/2021

END