Họ tên: Lưu Gia Huy

MSSV: 21520916

Lớp: NT106.N21.ATTN

FILE VÀ I/O STREAM TRONG C#

Báo cáo

Trước tiên ta sẽ tạo 1 form gồm các button chuyển hướng đến 5 bài tập riêng biệt như sau:

Giao diện:

Code:

```
1 reference
private void btnBai1_Click(object sender, EventArgs e)
{
    new Bai1().ShowDialog();
}
1 reference
private void btnBai2_Click(object sender, EventArgs e)
{
    new Bai2().ShowDialog();
}
1 reference
private void btnbai3_Click(object sender, EventArgs e)
{
    new Bai3().ShowDialog();
}
1 reference
private void btnbai4_Click(object sender, EventArgs e)
{
    new Bai4().ShowDialog();
}
1 reference
private void btnBai5_Click(object sender, EventArgs e)
{
    new Bai5().ShowDialog();
}
```

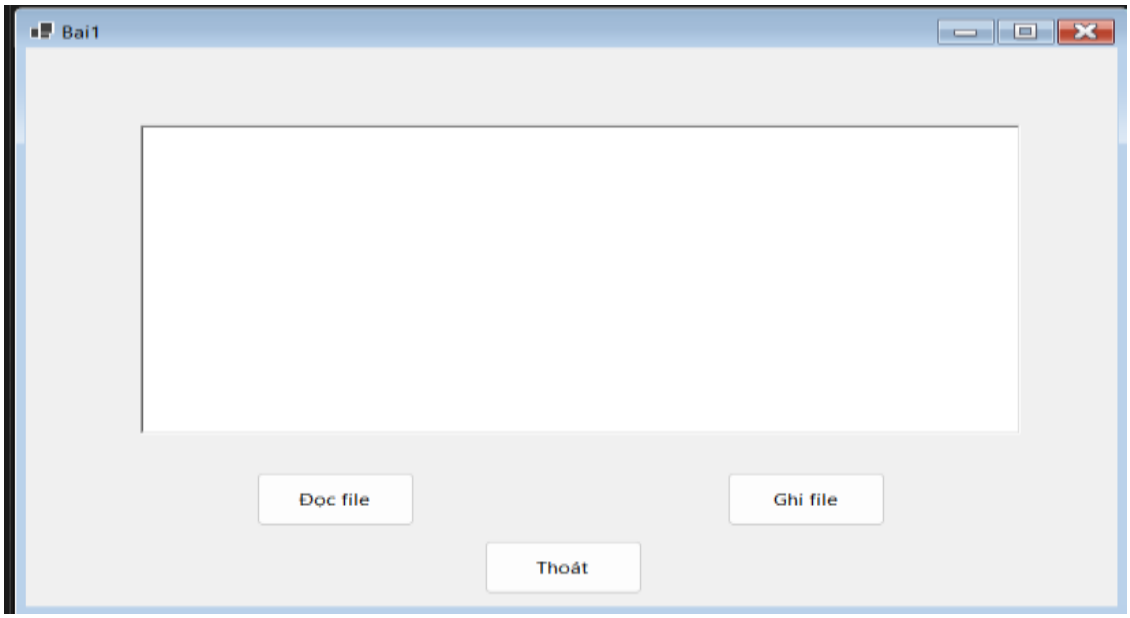
Với mỗi click vào từng button ta sẽ chuyển hướng đến form của bài tương ứng:

Bằng lệnh: new Baix().ShowDialog();

Baix với x là (1,2,3,4,5) tương ứng là tên của form từng bài.

Bài 1: Đọc và ghi file cơ bản

Giao diện:



Code:

- Đọc file:

```
1 reference
private void btnRead_Click(object sender, EventArgs e)
{
    OpenFileDialog ofd = new OpenFileDialog();
    string content;

    ofd.ShowDialog();
    if (ofd.FileName != "")
    {
        StreamReader str = new StreamReader(ofd.FileName);
        content = str.ReadToEnd();
        rtb.Text = content;
        str.Close();
    }
}
```

- Sử dụng StreamReader để đọc file.
- Mở hộp thoại chọn file với OpenFileDialog
- Nếu đã chọn được file (tức FileName khác rỗng) thì sẽ thực hiện quá trình ghi file ra richtextbox bằng cách :
- Đọc từ đầu đến cuối file rồi gán cho biến content (kiểu string) sau lại gán cho rtb.Text tức là richtextbox, Sau cùng sẽ dừng đọc file với str.Close()

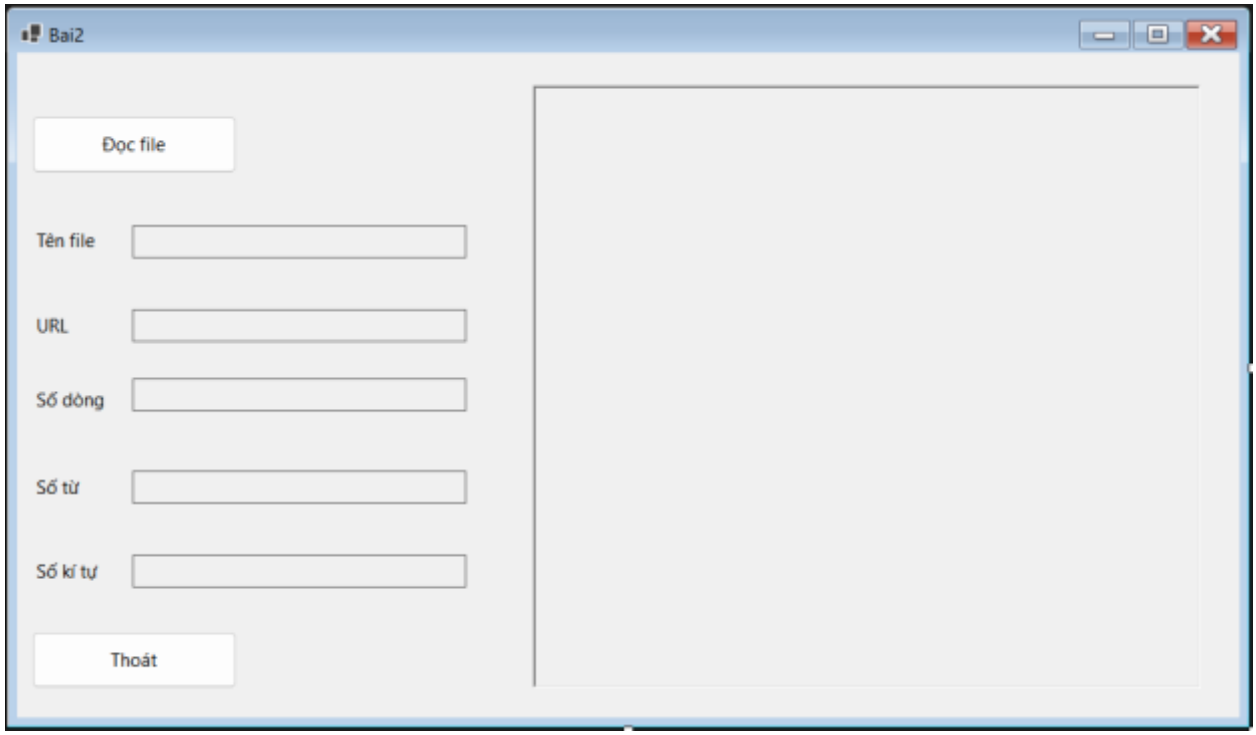
- Ghi file:

```
1 reference
private async void btnWrite_Click(object sender, EventArgs e)
{
    SaveFileDialog sfd = new SaveFileDialog();
    sfd.Filter = "Text (*.txt)|*.txt";
    sfd.ShowDialog();
    if (sfd.FileName != "")
    {
        using (StreamWriter write = new StreamWriter(sfd.FileName, false, Encoding.UTF8))
        {
            write.WriteLine(rtb.Text);
        }
    }
}
```

- Đoạn code trên sẽ tạo ra một SaveFileDialog (hộp thoại Lưu tệp tin) để cho phép người dùng chọn vị trí và tên tệp tin để lưu dữ liệu.
- Filter "Text (*.txt)|*.txt" chỉ cho phép người dùng lưu các tệp tin có phần mở rộng là .txt.
- Nếu người dùng đã chọn file để ghi (tức FileName phải khác rỗng) thi thực hiện việt ghi file
- Sử dụng StreamWriter để ghi dữ liệu từ một RichTextBox (rtb) vào tệp tin được chọn. Các tùy chọn của StreamWriter cụ thể là viết đè nội dung tệp tin (false) và sử dụng mã hóa UTF8 để đọc và ghi tệp tin.

Bài 2: Đọc thông tin tập tin (File .txt)

Giao diện:



Code:

```
private void btnReadfile_Click(object sender, EventArgs e)
{
    OpenFileDialog ofd = new OpenFileDialog();
    string content;
    ofd.ShowDialog();
    if (ofd.FileName != "")
    {
        StreamReader str = new StreamReader(ofd.FileName);
        content = str.ReadToEnd();
        rtbbai2.Text = content;
        str.Close();

        tbname.Text = ofd.SafeFileName;
        tburl.Text = ofd.FileName;
        tbsodong.Text = File.ReadLines(ofd.FileName).Count().ToString();
        tbsokitu.Text = content.Length.ToString();
        int l = 0;
        string[] myfile = rtbbai2.Text.Split("\n");
        foreach (string s in myfile)
        {
            string[] mstr = s.Split(" ");
            foreach (string s2 in mstr)
            {
                if (s2 != "")
                    l ++;
            }
        }
        tbsotu.Text = l.ToString();
    }
}
```

- Phần đọc file thì tương tự như đã giải thích ở bài 1:

```
OpenFileDialog ofd = new OpenFileDialog();
string content;
ofd.ShowDialog();
if (ofd.FileName != "")
{
    StreamReader str = new StreamReader(ofd.FileName);
    content = str.ReadToEnd();
    rtbbai2.Text = content;
    str.Close();
```

- Tuy nhiên sẽ không dừng lại mỗi việc đọc file sau khi nhấn vào button “Đọc file” Mã sẽ tính toán các vấn đề liên quan đến file vừa được đọc lên richtextbox, cụ thể như sau:

- Nếu người dùng đã chọn file để đọc tức là (FileName khác rỗng) thì file được load nội dung lên richtextbox
  Kế tiếp:

```
tbname.Text = ofd.SafeFileName;
tburl.Text = ofd.FileName;
tbsodong.Text = File.ReadLines(ofd.FileName).Count().ToString();
tbsokitu.Text = content.Length.ToString();
```

- Thực hiện lấy tên File gán vào textbox với nhãn là tên file
- Tương tự thế với việc lấy url
- Đếm số dòng bằng với File.ReadLines().Count(), rồi chuyển sang dạng chuỗi với ToString() rồi gán vào textbox với nhãn là số dòng
- Tiếp đến sẽ đếm số lượng kí tự và gán tương tự vào textbox với nhãn số kí tự

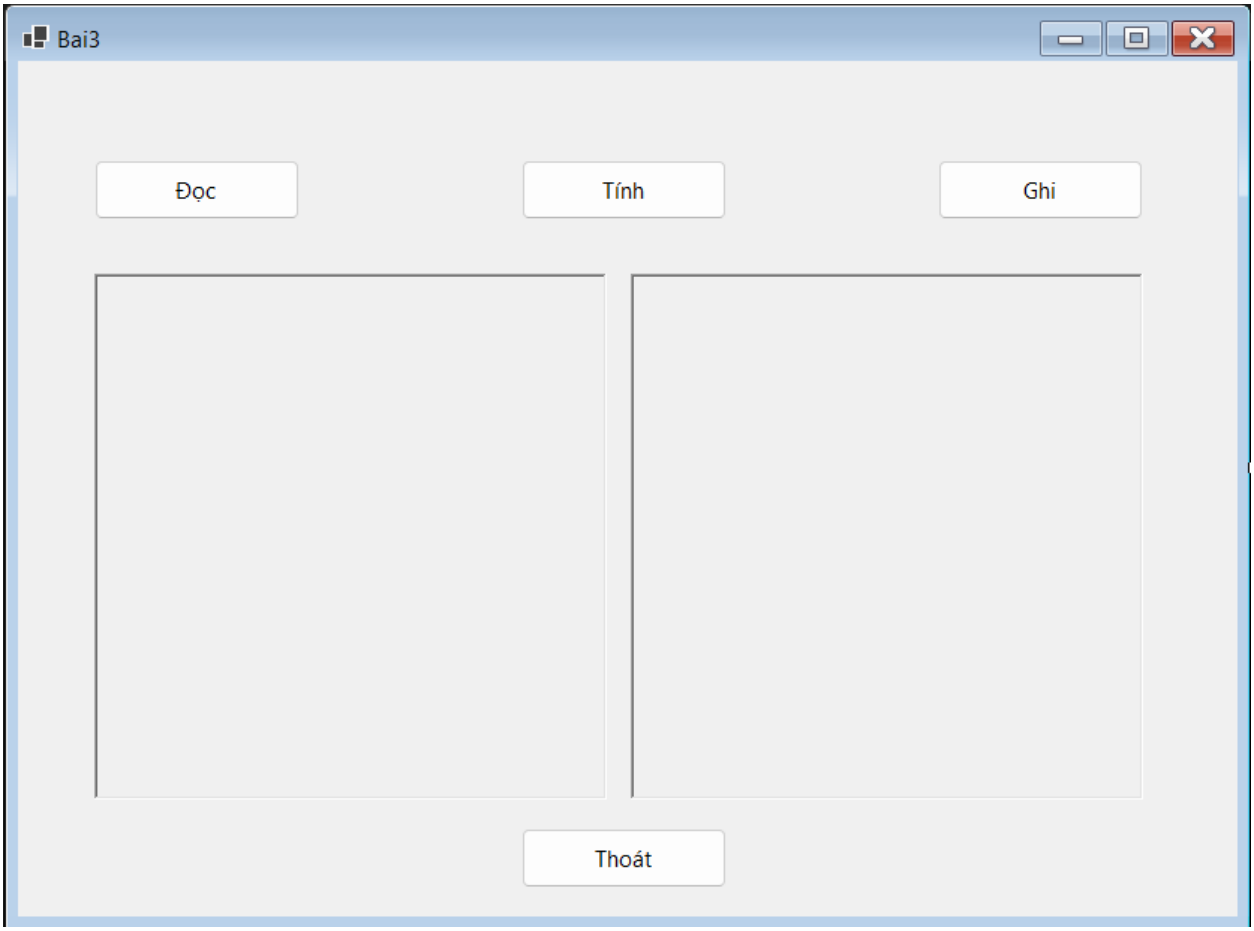
- Cuối cùng là đếm số từ bằng cách:

```
int l = 0;
string[] myfile = rtbbai2.Text.Split("\n");
foreach (string s in myfile)
{
    string[] mstr = s.Split(" ");
    foreach (string s2 in mstr)
    {
        if (s2 != "")
            l ++;
    }
}
tbsotu.Text = l.ToString();
```

  - ✓ Tách từng dòng nội dung đã được load lên richtextbox với Split("\n") gán vào mảng string tên là myfile
  - ✓ Với mỗi phần tử trong mảng myfile (tức mỗi dòng trong nội dung được load lên richtextbox) sẽ tiến hành tách từng từ với Split("\n") gán vào mảng string với tên là mstr
  - ✓ Tiếp đến với mỗi phần tử trong mảng mstr (tức là mỗi từ trong mỗi dòng). Nếu từ đó khác rỗng (để xử lý các dòng không tồn tại bất kì kí tự nào) thì tiến hành đếm (cộng 1 vào biến đếm l) , ban đầu biến l được khai báo bằng 0.
  - ✓ Sau khi đếm xong sẽ chuyển l sang dạng chuỗi với ToString() và gán vào textbox với nhãn là số từ.

Bài 3: Đọc và ghi file (mở rộng)

Giao diện:



Code:

- Nút Đọc :

```
1 reference
private void btnDoc_Click(object sender, EventArgs e)
{
    ofd.ShowDialog();
    if (ofd.FileName != "")
    {
        StreamReader str = new StreamReader(ofd.FileName);
        content = str.ReadToEnd();
        rtbinput.Text = content;
        str.Close();
    }
}
```

Tương tự như đã giải thích ở bài 1

- Nút tính:

```
string[] lines = rtbinput.Text.Trim().Split(new[] { "\r\n", "\n" }, StringSplitOptions.RemoveEmptyEntries);
```

Câu lệnh này nhằm tạo mảng string với mỗi phần tử là một dòng, với hàm Trim() để xóa khoảng trắng đầu và cuối chuỗi, Split() để xóa các kí tự xuống hàng “\n” hoặc “\r\n” khác nhau ở mỗi hệ thống, StringSplitOptions.RemoveEmptyEntries để xóa các phần tử rỗng trong mảng nếu có , lưu các phần tử tách được vào mảng lines

```
foreach (string line in lines)
{
    string[] parts = line.Trim().Split(new[] { '+', '-', '*', '/' }, StringSplitOptions.RemoveEmptyEntries);
    if (parts.Length != 2) continue;
```

Với mỗi phần tử trong mảng vừa tách ở trên (tức là mỗi dòng vừa tách được) sẽ tiếp tục tách các phép tính +, -, *, / và lưu các phần tử và mảng parts
Nếu chiều dài parts khác 2 thì sẽ bỏ qua dòng hiện tại, có nghĩa là chỉ thực hiện tính toán với 2 số

```
if (!double.TryParse(parts[0], out double num1)) continue;
if (!double.TryParse(parts[1], out double num2)) continue;
```

Dòng này sử dụng phương thức TryParse() để chuyển đổi chuỗi số đầu vào thành giá trị số thực (double). Nếu không thể chuyển đổi, dòng lệnh tiếp theo trong vòng lặp sẽ được bỏ qua.

```
double result = 0;
char op = line.FirstOrDefault(c => "+-*/".Contains(c));
```

Khai báo biến result để lưu kết quả, khởi tạo bằng 0
Khai báo biến op (char) để lưu giá trị của biểu thức sẽ được thực hiện, sử dụng phương thức FirstOrDefault() để lấy giá trị đầu tiên hoặc giá trị đầu tiên thỏa mãn là một trong các kí tự “+-*/” và gán lại giá trị ấy cho biến op

```
if (op == '+')
{
    result = num1 + num2;
    output += $"{line} = {result}\r\n";
}
else if (op == '-')
{
    result = num1 - num2;
    output += $"{line} = {result}\r\n";
}
else if (op == '*')
{
    result = num1 * num2;
    output += $"{line} = {result}\r\n";
}
else if (op == '/')
{
    if (num2 != 0)
    {
        result = num1 / num2;
        result = Math.Round(result, 3);
        output += $"{line} = {result}\r\n";
    }
    else output += $"{line} = Null \r\n";
}
```

- ✓ Với mỗi giá trị mà op lưu trữ (tương ứng là phép tính) thì ta sẽ thực hiện các phép tính tương ứng, riêng phép chia thì ta sẽ làm tròn 3 chữ số phần thập phân với Math.Roung(result,3)
- ✓ output+= …. ở đây nghĩa là cộng kết quả của các line thỏa mãn vào cách nhau "\r\n" tức là mỗi line là mỗi dòng trong output
- ✓ Dấu $ cho phép lồng ghép các buổi thức vào chuỗi string

```
rtbouput.Text = output;
```

- ✓ Cuối cùng sẽ gán output cho richtextbox ouput tương ứng

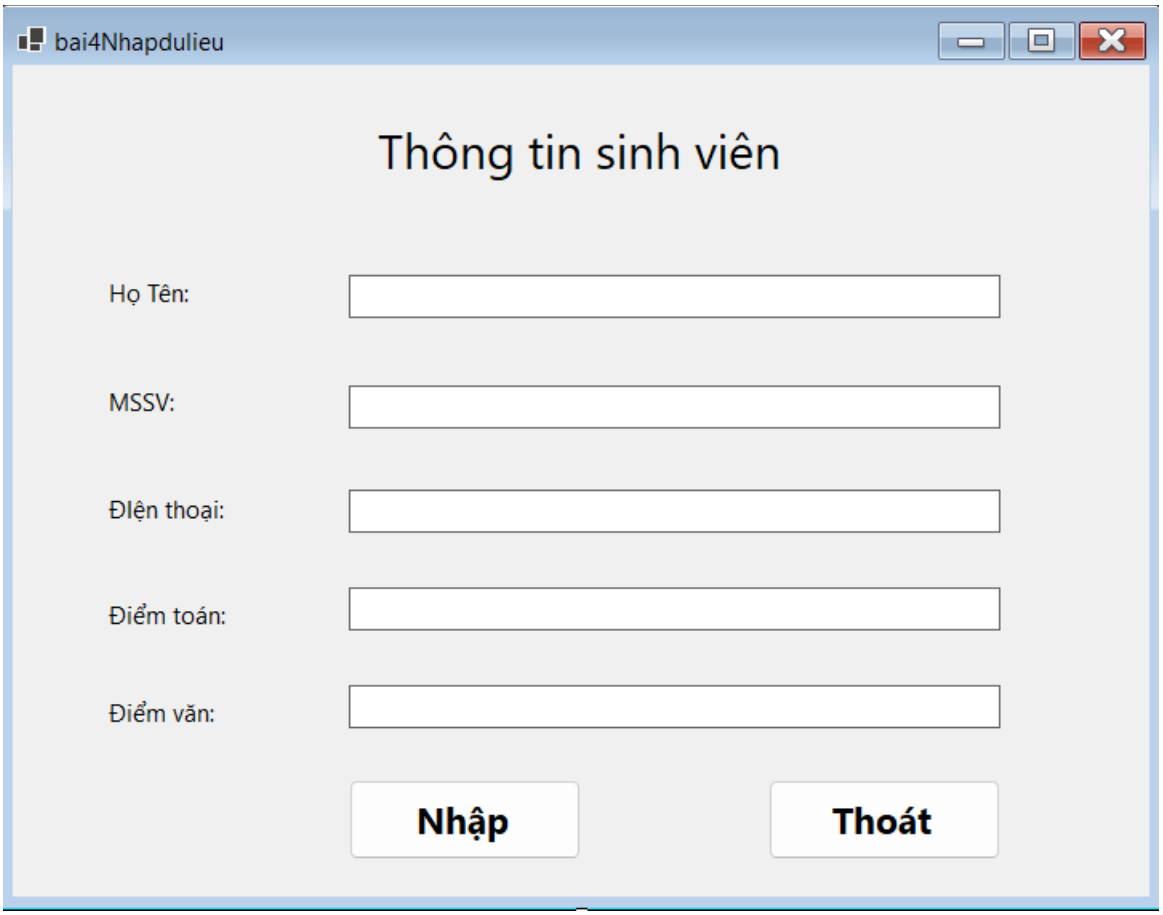
- Nút ghi:

```
1 reference
private void btnGhi_Click(object sender, EventArgs e)
{
    SaveFileDialog sfd = new SaveFileDialog();
    sfd.Filter = "Text (*.txt)|*.txt";
    sfd.ShowDialog();
    if (sfd.FileName != "")
    {
        using (StreamWriter write = new StreamWriter(sfd.FileName, false, Encoding.UTF8))
        {
            write.WriteLine(rtbouput.Text);
        }
    }
}
```

Tương tự như đã giải thích ở bài 1

Bài 4: Làm việc với file, CSDL

Giao diện:



Code:

- Phần nhập dữ liệu:

```
double number;
bool success1 = double.TryParse(tbdiemtoan.Text, out number);
bool success2 = double.TryParse(tbdiemvan.Text, out number);
bool hoTenHopLe = Regex.IsMatch(tbhoten.Text, "^[a-zA-Z ]+$");
bool soDienThoaiHopLe = Regex.IsMatch(tbdt.Text, "^[0-9]+$");
```

  - ✓ 2 biến success1,2 để kiểm tra xem 2 con điểm toán và văn được nhập vào có phải là số hay không, nếu là số thì trả về true, ngược lại là false
  - ✓ 2 biến hoTenHopLe,soDienThoaiHopLe sẽ kiểm tra xem tên có phải chỉ chứa các kí tự chữ cái và số điện thoại có phải chỉ chứa số hay không, nếu đúng sẽ trả về true, nếu sai sẽ trả về false
  - ✓ Regex.IsMatch để kiểm tra xem giá trị hiện tại có chứa các kí tự không đúng định dạng hay không.

```
if (tbhoten.Text == "" || tbmssv.Text == "" || tbdt.Text == "" || tbdiemtoan.Text == "" || tbdiemvan.Text == "" || !success1 || !success2 || !hoTenHopLe || !soDienThoaiHopLe)
        {MessageBox.Show("Hãy nhập đúng và đủ các trường thông tin");}
```

- ✓ Dòng if này kiểm tra xem có trường thông tin nào người dùng còn để trống, hoặc trường điểm toán và văn, tên, số điện thoại nhập sai định dạng.
- ✓ Nếu nhập sai sẽ hiện thông báo yêu cầu người dùng nhập lại mà không thực hiện bất kì tính toán nào khác trên dữ liệu bị sai.

```
else
{
    string filePath = "input.txt";
    string data = "";
    double diemtoan = double.Parse(tbdiemtoan.Text);
    double diemvan = double.Parse(tbdiemvan.Text);

    data = tbhoten.Text + ';' + tbmssv.Text + ';' + tbdt.Text + ';' + tbdiemtoan.Text + ';

    using (StreamWriter writer = new StreamWriter(filePath, true, Encoding.UTF8))
    {
        writer.Write(data);
    }
    tbhoten.Text = string.Empty;
    tbmssv.Text = string.Empty;
    tbdt.Text = string.Empty;
    tbdiemvan.Text = string.Empty;
    tbdiemtoan.Text = string.Empty;
    MessageBox.Show("Đã cập nhật dữ liệu");

}
```

- ✓ Nếu input đã không có bất cứ vấn đề nào, ta sẽ bắt đầu tính toán.
- ✓ Convert điểm toán và văn ở dạng string sang dạng double với Double.Parse()
- ✓ Kế đến sẽ gán cho biến data(có kiểu dữ liệu là string) chứa các trường vừa nhập từ user cách nhau bởi dấu “;” để lưu với định dạng này vào file input.txt
- ✓ Dùng using để đảm bảo tài nguyên được giải phóng sau khi sử dụng
- ✓ Ta dùng StreamWriter để ghi data vào filePath chính là input.txt
- ✓ Sau đó sẽ thực hiện xóa các trường vừa nhập và Hiện thông báo rằng là đã cập nhật dữ liệu.

- Phần Lưu:

```
1 reference
private void btnsave_Click(object sender, EventArgs e)
{
    string content = File.ReadAllText("input.txt");
    const string SheetName = "Sheet1";
    var workbook = new XLWorkbook();
    var worksheet = workbook.Worksheets.Add(SheetName);

    worksheet.Cell(1, 1).Value = "Tên";
    worksheet.Cell(1, 2).Value = "MSSV";
    worksheet.Cell(1, 3).Value = "SĐT";
    worksheet.Cell(1, 4).Value = "Toán";
    worksheet.Cell(1, 5).Value = "Văn";
    worksheet.Cell(1, 6).Value = "TB";

    string[] myfile = content.Split("\n");
    for (int i = 0; i < myfile.Length; i++)
    {
        string[] mstr = myfile[i].Split(";");
        for (int j = 0; j < mstr.Length; j++)
        {
            worksheet.Cell(i + 2, j+1 ).Value = mstr[j];
        }
    }
    workbook.SaveAs("output.xlsx");
}
```

using ClosedXML.Excel;

- ✓ Ta dùng thư viện CloseXML.Excel để thực hiện ghi file excel

- ✓ Tạo mỗi biến kiểu string tên là content lưu giá trị của input.txt với method File.ReadAllText
- ✓ Khai báo một hằng số SheetName có giá trị là “Sheet1”
- ✓ Tạo một workbook mới sử dụng thư viện ClosedXML
- ✓ Thêm một worksheet mới vào workbook với tên là sheetname (Sheet1)
- ✓ Thêm các tên cột tại các vị trí cụ thể với worksheet.Cell().Value
- ✓ Tạo 1 mạng chuỗi tương ứng với mỗi phần tử là mỗi dòng trong file input.txt tách nhau bởi dấu “\n”
- ✓ Ứng với mỗi dòng ta tách được, sẽ thực hiện tách tiếp mỗi phần tử trong dòng đó cách nhau bởi “;” với Split() gán vào mảng chuỗi mstr
- ✓ Và thực hiện gán mỗi phần tử đấy vào ô tương ứng, với i biểu hiện cho cột, j biểu hiện cho dòng
- ✓ Cuối cùng lưu workbook vào file Sample.xlsx

- Phần đọc nội dung File excel ra DataGridView:

```
1 reference
private void btnoutput_Click(object sender, EventArgs e)
{
    using (FileStream stream = new FileStream("output.xlsx", FileMode.Open, FileAccess.Read))
    {
        IWorkbook workbook = new XSSFWorkbook(stream);
        ISheet sheet = workbook.GetSheetAt(0);
        DataTable table = new DataTable();
        table.Columns.Add("Tên");
        table.Columns.Add("MSSV");
        table.Columns.Add("SĐT");
        table.Columns.Add("Toán");
        table.Columns.Add("Văn");
        table.Columns.Add("TB");

        foreach (IRow row in sheet)
        {
            DataRow dataRow = table.NewRow();
            foreach (ICell cell in row)
                if (cell != null)
                    if (row.RowNum != 0)
                        dataRow[cell.ColumnIndex] = cell.ToString();

            if (row.RowNum != 0)
                table.Rows.Add(dataRow);
        }
        dgv.DataSource = table;
        workbook.Dispose();
    }
}
```

- ✓ Ở phần đọc file excel ta sẽ dùng 2 thư viện:

```
using NPOI.SS.UserModel;
using NPOI.XSSF.UserModel;
```

- ✓ Dùng using để đảm bảo tài nguyên được giải phóng sau khi sử dụng
- ✓ Ta sẽ đọc file với FileStream
- ✓ Sau khi mở tập tin, chúng ta tạo một đối tượng IWorkbook để đại diện cho file Excel.
- ✓ XSSFWorkbook là một lớp cung cấp chức năng đọc và ghi workbook định dạng Office Open XML (xlsx) trong Apache POI.
- ✓ workbook.GetSheetAt(0) lấy bảng đầu tiên trong workbook để đọc dữ liệu.
- ✓ DataTable là một lớp trong ADO.NET cho phép tạo và quản lý các bảng trong bộ nhớ.
- ✓ table.Columns.Add() thêm các cột vào bảng, ở đây là các cột "Tên", "MSSV", "SĐT", "Toán", "Văn", "TB".

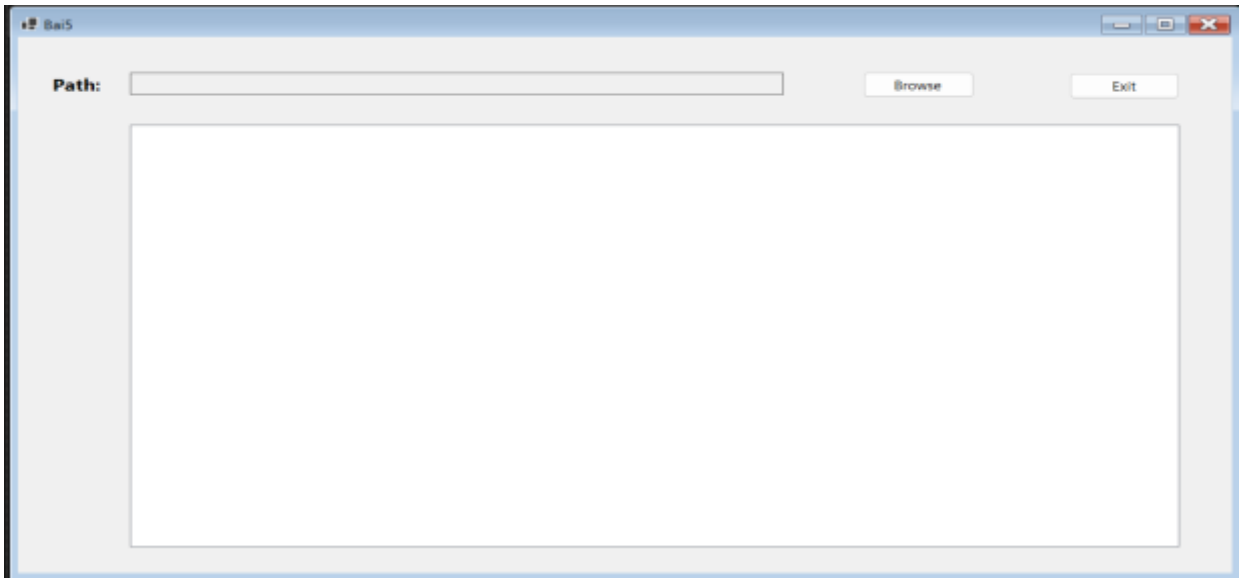
```
foreach (IRow row in sheet)
{
    DataRow dataRow = table.NewRow();
    foreach (ICell cell in row)
        if (cell != null)
            if (row.RowNum != 0)
                dataRow[cell.ColumnIndex] = cell.ToString();

    if (row.RowNum != 0)
        table.Rows.Add(dataRow);
}
dgv.DataSource = table;
workbook.Dispose();
```

- ✓ Với mỗi hàng (IRow) trong bảng, chúng ta tạo một đối tượng DataRow mới để lưu trữ dữ liệu hàng đó.

- ✓ Với mỗi ô (ICell) trong hàng đó, chúng ta kiểm tra xem nó có khác null hay không, nếu có thì xem tiếp nếu nó không phải hàng đầu tiên (do hàng đầu tiên chứa tên cột) thì gán giá trị ô đó vào cột tương ứng trong DataRow.
- ✓ row.RowNum trả về số thứ tự của hàng trong bảng, nếu hàng đó không phải là hàng đầu tiên thì chúng ta thêm nó vào bảng.
- ✓ Cuối cùng, dgv.DataSource được gán bằng table để hiển thị dữ liệu lên DataGridView
- ✓ workbook.Dispose() được sử dụng để giải phóng tài nguyên của workbook.

Bài 5: Duyệt file trong thư mục

Giao diện:



Code:

- Đầu tiên sẽ sử dụng 1 cái dictionary để khi nhận được type .txt có thể convert sang Text document

```
private Dictionary<string, string> extensions = new Dictionary<string, string>()
{
    { ".txt", "Text document" },
    { ".pdf", "PDF document" },
    { ".docx", "Word document" },
    { ".xlsx", "Excel document" },
    { ".png", "PNG image" },
    { ".jpg", "JPEG image" },
    { ".gif", "GIF image" },
    { ".bmp", "Bitmap image" },
    { ".zip", "Zip archive" },
    { ".rar", "RAR archive" },
    { ".7z", "7-Zip archive" },
    { ".mp3", "MP3 audio" },
    { ".wav", "WAV audio" },
    { ".avi", "AVI video" },
    { ".mp4", "MPEG-4 video" },
    { ".mkv", "Matroska video" },
    { ".exe", "Executable file" },
    { ".dll", "Dynamic link library" },
    { ".pptx", "PowerPoint document" },
    { ".pub", "Publisher document" },
    { ".psd", "Photoshop document" },
```

……..

- Tiếp đến là :

```
1 reference
private void Bai5_Load(object sender, EventArgs e)
{
    listView.View = View.Details;
    listView.FullRowSelect = true;
    listView.Columns.Clear();
    listView.Columns.Add("Name", 450);
    listView.Columns.Add("Date Modified", 270);
    listView.Columns.Add("Type", 220);
    listView.Columns.Add("Size", 90);
    listView.Columns[0].ListView.Font = new Font(listView.Font, FontStyle.Bold);
    listView.Columns[1].ListView.Font = new Font(listView.Font, FontStyle.Bold);
    listView.Columns[2].ListView.Font = new Font(listView.Font, FontStyle.Bold);
    listView.Columns[3].ListView.Font = new Font(listView.Font, FontStyle.Bold);
}
```

✓ Tức là khi form được load lên ta sẽ thêm setup như là thêm các cột vào, định dạng nó

- Browse

```
private void btnBrowse_Click(object sender, EventArgs e)
{
    FolderBrowserDialog FBD = new FolderBrowserDialog();
    if (FBD.ShowDialog() == DialogResult.OK)
    {
        tburl.Text = FBD.SelectedPath;
        listView.Items.Clear();
        DirectoryInfo di = new DirectoryInfo(FBD.SelectedPath);
        FileInfo[] fiArr = di.GetFiles();
        foreach (FileInfo fi in fiArr)
        {
            ListViewItem lvi = new ListViewItem(fi.Name);
            lvi.SubItems.Add(fi.LastWriteTime.ToString());
            if (extensions.ContainsKey(fi.Extension))
                lvi.SubItems.Add(extensions[fi.Extension]);
            else
                lvi.SubItems.Add(fi.Extension);
            long sizeInBytes = fi.Length;
            string[] sizes = { "B", "KB", "MB", "GB" };
            int order = 0;
            while (sizeInBytes >= 1024 && order < sizes.Length - 1)
            {
                order++;
                sizeInBytes /= 1024;
            }
            string size = String.Format("{0:0.##} {1}", sizeInBytes, sizes[order]);
            lvi.SubItems.Add(size);
            listView.Items.Add(lvi);
        }
    }
```

  - ✓ Đầu tiên sẽ khởi tạo một đối tượng mới của lớp FolderBrowserDialog để hiển thị hộp thoại để người dùng có thể chọn một thư mục.
  - ✓ Sau đó kiểm tra xem người dùng đã chọn thư mục và nhấn OK chưa. Nếu đã ok rồi sẽ tiếng hành xử lý
  - ✓ Gán URL của thư mục đã chọn vào textbox với nhãn là url
  - ✓ Clear listview hiện tại, để chuẩn bị cho sự xuất hiện của danh sách mới

```
DirectoryInfo di = new DirectoryInfo(FBD.SelectedPath);
FileInfo[] fiArr = di.GetFiles();
```

  - ✓ Đoạn này tạo một đối tượng DirectoryInfo và lấy tất cả các tệp trong thư mục đã chọn bằng phương thức GetFiles().

```
foreach (FileInfo fi in fiArr)
{
    ListViewItem lvi = new ListViewItem(fi.Name);
    lvi.SubItems.Add(fi.LastWriteTime.ToString());
    if (extensions.ContainsKey(fi.Extension))
        lvi.SubItems.Add(extensions[fi.Extension]);
    else
        lvi.SubItems.Add(fi.Extension);
    long sizeInBytes = fi.Length;
    string[] sizes = { "B", "KB", "MB", "GB" };
    int order = 0;
    while (sizeInBytes >= 1024 && order < sizes.Length - 1)
    {
        order++;
        sizeInBytes /= 1024;
    }
    string size = String.Format("{0:0.##} {1}", sizeInBytes, sizes[order]);
    lvi.SubItems.Add(size);
    listView.Items.Add(lvi);
}
```

- ✓ Đoạn code này lặp lại các FileInfo đã lấy được và tạo các ListViewItem tương ứng để hiển thị tên tệp, ngày ghi cuối cùng, loại tệp và kích thước tệp.
- ✓ Kích thước tệp được chuyển đổi sang đơn vị bytes, kilobytes, megabytes hoặc gigabytes và được hiển thị với độ chính xác hai chữ số thập phân.
- ✓ Cuối cùng, ListViewItem được thêm vào ListView để hiển thị danh sách các tệp.