

BÁO CÁO THỰC HÀNH

Môn học: Nhập môn mạng máy tính

Buổi báo cáo: Lab 02

Tên chủ đề: File và I/O Stream trong C#

GVHD: Nguyễn Xuân Hà

Ngày thực hiện: 28/03/2024

THÔNG TIN CHUNG:

Lớp: NT106.O23.2

(Liệt kê tất cả các thành viên trong nhóm)

STT	Họ và tên	MSSV	Email
1	Phạm Huỳnh Tấn Khang	22520624	22520624@gm.uit.edu.vn

1. ĐÁNH GIÁ KHÁC:

Nội dung	Kết quả
Tổng thời gian thực hiện bài thực hành trung bình	8 tiếng
Link GitHub (nếu có)	https://github.com/VitalsZen
Ý kiến (nếu có) + Khó khăn + Đề xuất ...	

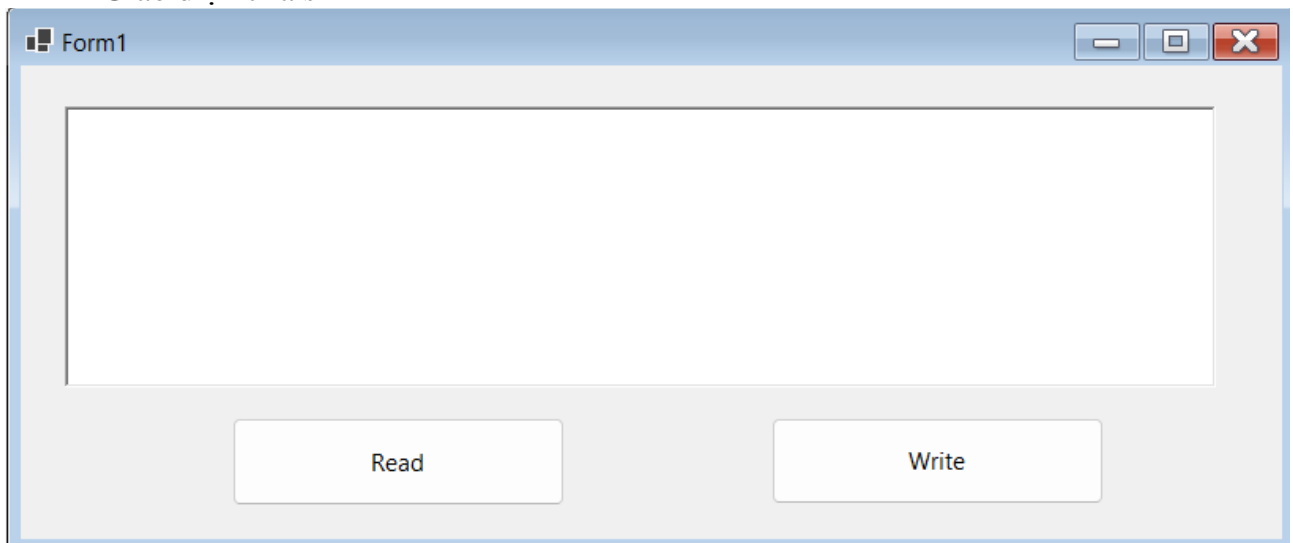
Phần bên dưới của báo cáo này là báo cáo chi tiết của nhóm thực hiện.

BÁO CÁO CHI TIẾT

Bài 01 – Ghi và Đọc file

Viết chương trình đọc nội dung một file “input1.txt” và xuất ra màn hình. Sau đó ghi nội dung (chuyển toàn bộ ký tự sang kiểu in hoa) xuống file “output1.txt”.

- Giao diện cửa sổ



- Hàm kích hoạt khi được Click vào của button “Read”
 - Đọc file input1.txt
 - Kiểm tra lỗi

```

1 reference
private void bt1_Read_Click(object sender, EventArgs e)
{
    try
    {
        FileStream fs = new FileStream("input1.txt", FileMode.Open);
        StreamReader sr = new StreamReader(fs);
        rtb1.Text = sr.ReadToEnd();
        fs.Close();
        if (rtb1.Text == "")
            MessageBox.Show("File input1.txt không có dữ liệu nào");
        else
            MessageBox.Show("Hiển thị nội dung file");
    }
    catch (FileNotFoundException)
    {
        MessageBox.Show("Không tìm thấy file input1.txt, vui lòng chọn file khác");
        OpenFileDialog ofd = new OpenFileDialog();
        ofd.ShowDialog();
        FileStream fs = new FileStream(ofd.FileName, FileMode.Open);
        StreamReader sr = new StreamReader(fs);
        rtb1.Text = sr.ReadToEnd();
        fs.Close();
        if (rtb1.Text == "")
            MessageBox.Show("File không có dữ liệu nào");
        else
            MessageBox.Show("Hiển thị nội dung file trên richtextbox");
    }
    catch (IOException)
    {
        MessageBox.Show("Đã xảy ra lỗi đọc file");
    }
    catch (Exception)
    {
        MessageBox.Show("Đã xảy ra lỗi");
    }
}

```

- Hàm kích hoạt khi được click của button “Write”
 - Ghi nội dung từ richtextbox rtb1 vào file output1.txt
 - Check và thông báo lỗi

```

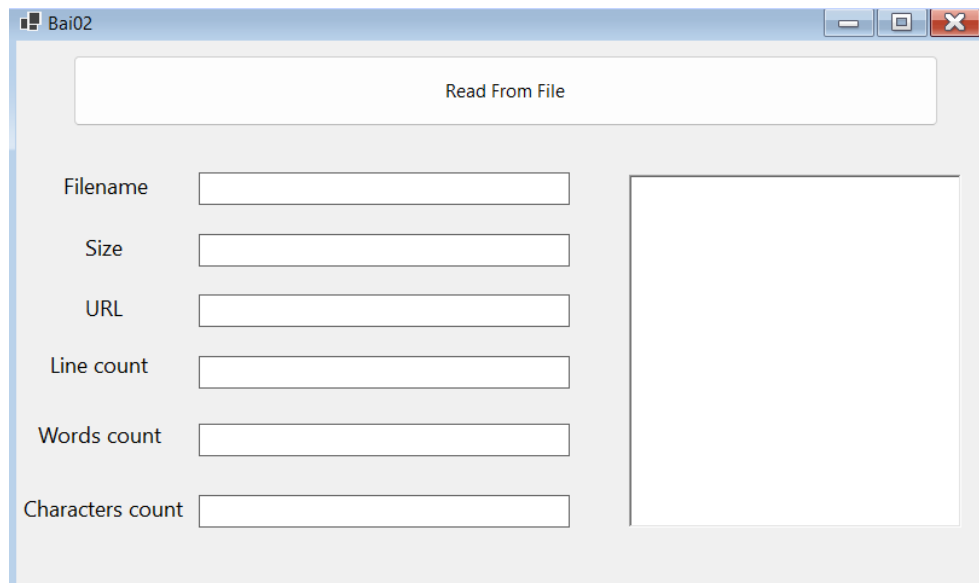
1 reference
private void bt2_Write_Click(object sender, EventArgs e)
{
    try
    {
        FileStream fs = new FileStream("output1.txt", FileMode.Create);
        using (StreamWriter sr = new StreamWriter(fs))
        {
            sr.WriteLine(rtb1.Text);
            MessageBox.Show("Đã ghi file thành công");
            sr.Close();
        }
    }
    catch (IOException)
    {
        MessageBox.Show("Đã xảy ra lỗi ghi file");
    }
    catch (Exception)
    {
        MessageBox.Show("Đã xảy ra lỗi");
    }
}

```

Bài 02 – Đọc thông tin một file .txt

Viết chương trình đọc file và hiển thị các thông tin sau:

- Tên file
- Kích thước file
- Đường dẫn Url
- Số dòng, số từ, số ký tự
- Hiển thị nội dung của file



Giao diện

- Hàm kích hoạt khi được click của button “Read From File”
 - Mở OpenFileDialog để chọn file

```

private void bt1_Read_Click(object sender, EventArgs e)
{
    try
    {
        OpenFileDialog ofd = new OpenFileDialog();
        ofd.ShowDialog();
        FileStream fs = new FileStream(ofd.FileName, FileMode.Open);
        StreamReader sr = new StreamReader(fs);
        string s = fs.Name.ToString();
        int dem = sr.Peek();
        string k = sr.ReadToEnd();
        tb1_Name.Text = ofd.SafeFileName.ToString();
        long ByteFile = new FileInfo(s).Length;
        tb2_Size.Text = ByteFile.ToString() + " bytes";
        tb3_URL.Text = s.ToString();
        tb4_LineC.Text = k.Split('\n').Length.ToString() ;
        tb5_WordsC.Text = k.Split(' ', '\n').Length.ToString();
        tb6_CharC.Text = k.Length.ToString();
        rtb1.Text = k;
        fs.Close();
        if (k == "")
            MessageBox.Show("File không có dữ liệu nào");
        else
            MessageBox.Show("Hiển thị nội dung file trên richtextbox");
    }

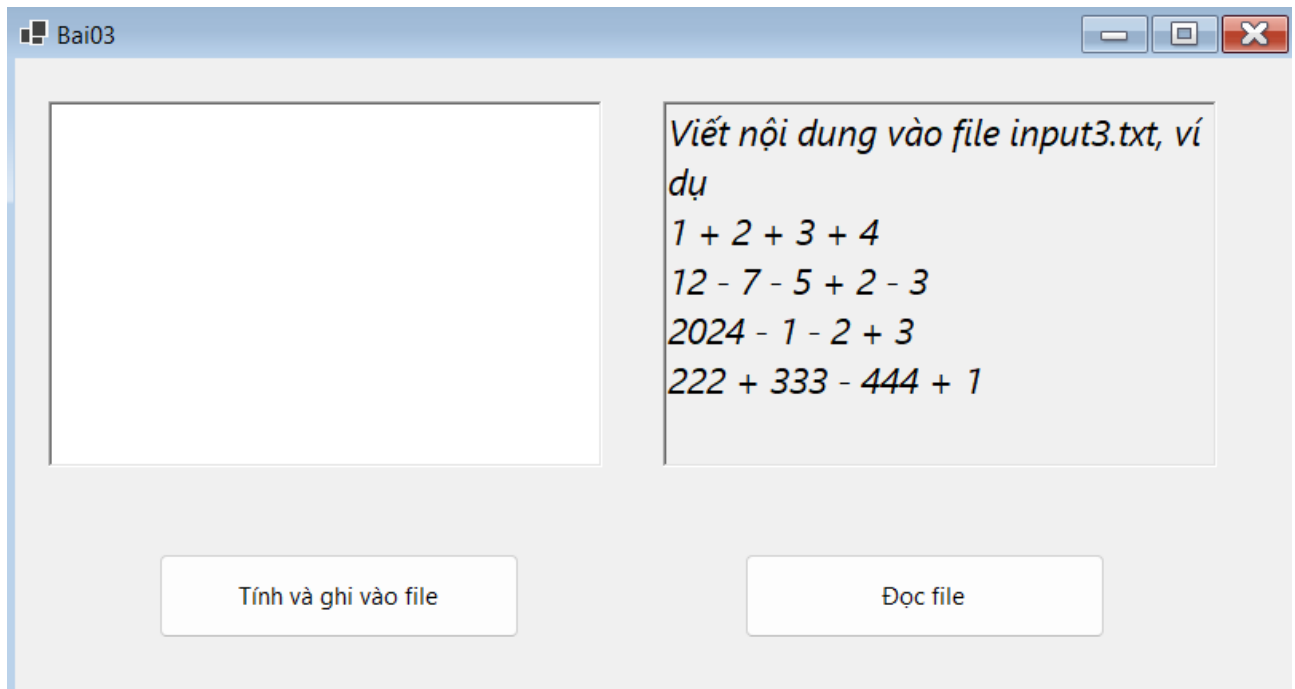
    catch (IOException)
    {
        MessageBox.Show("Đã xảy ra lỗi đọc file");
    }
    catch (Exception)
    {
        MessageBox.Show("Đã xảy ra lỗi");
    }
}

```

Bài 03 - Đọc và Ghi file và tính toán

Đọc nội dung từ file “input3.txt” với nội dung theo định dạng, sau đó thực hiện các phép tính và ghi kết quả xuống file “output3.txt”. Các phép tính bao gồm: cộng trừ, nhân, chia, ngoặc đơn.

Lưu ý: Không sử dụng phương thức `DataTable.Compute`.



Giao diện

Hàm `Uutien` sử dụng để xác định việc ưu tiên phép nhân chia hơn là phép cộng trừ
Hàm `TinhCapSo` sử dụng để tính và trả về giá trị thực của các cặp tham số được truyền vào

2 references

```
private int Uutien(char op)
{
    if (op == '+' || op == '-')
        return 1;
    if (op == '*' || op == '/')
        return 2;
    return 0;
}
```

3 references

```
private double TinhCapSo(double a, double b, char op)
{
    switch (op)
    {
        case '+': return a + b;
        case '-': return a - b;
        case '*': return a * b;
        case '/': return a / b;
        default: return 0;
    }
}
```

- Hàm Tính là hàm chính trong bài
 - Tạo 2 stack values và ops (values: lưu số để thực hiện phép tính; ops: lưu các toán tử đã quét chạy cùng values)
 - Quét từ đầu cho đến cuối của tham số truyền vào
 - chuoiso[i] = ' ' → chuyển sang vòng lặp tiếp theo
 - chuoiso[i] = { một số } → thực hiện vào vòng lặp while từ vị trí hiện tại để gán chuoiso[i] hiện tại + những số theo sau (nếu đó là số có nhiều chữ số) vào chuỗi operand → sau đó push vào stack values
 - chuoiso[i] = '(' → push vào stack ops
 - chuoiso[i] = ')' → check nếu phần tử cuối cùng của ops có ký tự '('
 - Không (vị trí quét đang ở trong cặp ngoặc đơn): thực hiện TínhCapSo trong vòng lặp while → Tính tất cả số được lưu trong values từ lúc ops được push '('
 - Có (đã tính cặp ngoặc đơn trước đó) → Pop dấu '(' ra ngoài
 - chuoiso[i] thuộc {+,-,*,/} → Tính tất cả các số trong values dựa vào các toán tử trong ops (nếu ops.count > 0) + thực hiện hàm ƯuTien (nếu toán tử cuối cùng trong ops có độ ưu tiên cao hơn chuoiso[i]) để xác định chuoiso[i] với ops cuối cùng thì toán tử nào có độ ưu tiên cao hơn (*,/ > +,-) sau đó push chuoiso[i] vào ops
 - Chạy vòng lặp while khi chạy quét xong (tính tất cả các số còn lại trong values)

```

1reference
private double Tính(string chuoiso)
{
    Stack<double> values = new Stack<double>();
    Stack<char> ops = new Stack<char>();

    for (int i = 0; i < chuoiso.Length; i++)
    {
        if (chuoiso[i] == ' ')
            continue;

        if (char.IsDigit(chuoiso[i]) || chuoiso[i] == '.') // chuoiso[i] = so
        {
            string operand = "";
            while (i < chuoiso.Length && (char.IsDigit(chuoiso[i]) || chuoiso[i] == '.')) // trường hợp số có trên 1 chu số
            {
                operand += chuoiso[i];
                i++;
            }
            i--;
            values.Push(double.Parse(operand)); // push vào lưu trong stack
        }
        else if (chuoiso[i] == '(')
        {
            ops.Push(chuoiso[i]);
        }
        else if (chuoiso[i] == ')')
        {

```

```

while (ops.Count > 0 && ops.Peek() != '(') // check đã o trong ngoac don hay chua ?
{
    double val2 = values.Pop();
    double val1 = values.Pop();
    char op = ops.Pop();
    values.Push(TinhCapSo(val1, val2, op));
}
if (ops.Count > 0 && ops.Peek() == '(')
    ops.Pop(); // Remove '('
}
else
{
    while (ops.Count > 0 && Utien(ops.Peek()) >= Utien(chuoiso[i])) // check dau de uu tien nhan chia truoac
    {
        double val2 = values.Pop();
        double val1 = values.Pop();
        char op = ops.Pop();
        values.Push(TinhCapSo(val1, val2, op));
    }
    ops.Push(chuoiso[i]);
}

while (ops.Count > 0) // tinh nhung gi con lai trong stack
{
    double val2 = values.Pop();
    double val1 = values.Pop();
    char op = ops.Pop();
    values.Push(TinhCapSo(val1, val2, op));
}

return values.Peek();
}

```

- 2 hàm kích hoạt khi click của 2 button “Tính và ghi vào file” và “Đọc file”
 - Đọc file và ghi file

```

1 reference
private void bt1_Read_Click_1(object sender, EventArgs e)
{
    rtb1.Clear();
    try
    {
        FileStream fs = new FileStream("output3.txt", FileMode.Open);
        using (StreamReader sr = new StreamReader(fs))
        {
            rtb1.Text += sr.ReadToEnd();
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

1 reference
private void bt2_Write_Click(object sender, EventArgs e)
{
    try
    {
        TinhvaGhi("input3.txt", "output3.txt");
        MessageBox.Show("Tinh thanh cong ");
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

```


Bài 4 - Đọc và Ghi file sử dụng BinaryFormatter (JsonSerializer)

Viết chương trình sử dụng BinaryFormatter cho phép : Nhập 1 mảng các sinh viên (không nhập điểm trung bình) và ghi xuống file “input4.txt”. Cấu trúc của Sinh viên như sau :

Họ và tên : String

MSSV : Int

Điện thoại : String

Điểm môn 1 : Float

Điểm môn 2 : Float

Điểm môn 3 : Float

Điểm trung bình : Float

Đọc thông tin mảng Học Viên từ file “input4.txt” và tính điểm trung bình cho từng sinh viên sau đó ghi xuống file “output4.txt” và xuất ra màn hình.

Giao diện

- Class Bai04_Student

```
internal class Bai04_Student
{
    public string FullName { get; set; }
    public int ID { get; set; }
    public string PhoneNum { get; set; }
    public float C1 { get; set; }
    public float C2 { get; set; }
    public float C3 { get; set; }
    public float Avg { get; set; }
    public override string ToString()
    {
        return $"FullName: {FullName}\n" +
            $"ID: {ID}\n" +
            $"PhoneNum: {PhoneNum}\n" +
            $"C1: {C1}\n" +
            $"C2: {C2}\n" +
            $"C3: {C3}\n" +
            $"Avg: {Avg}\n";
    }
}
```

- Hàm kích hoạt khi được click của button “Write to File”
 - Bắt đầu quá trình viết dữ liệu vào file

```
private void bt1_WtF_Click(object sender, EventArgs e)
{
    if (tb7_Total.Text == "")
    {
        MessageBox.Show("Nhập số học sinh vào Total");
        tb7_Total.Enabled = true;
        rtb1_Show.Text = "Bấm vào đúng số lượng học sinh cần nhập vào ô Total";
        return;
    }
}
```

- Hàm SerializeJson(object obj, string Filepath): Chuyển đổi tượng thành chuỗi JSON và lưu vào tệp tin tại đường dẫn được chỉ định. Sử dụng thư viện Newtonsoft hoặc System.Text.Json để thực hiện việc chuyển đổi

```
< references
private void SerializeJson(object obj, string Filepath)
{
    string json = JsonSerializer.Serialize(students);
    File.WriteAllText(Filepath, json);
}
```

- Hàm ShowStudent(int pagenum): Hiển thị thông tin của một học sinh ở trang được chỉ định. Dùng để hiển thị thông tin chi tiết của học sinh trong danh sách

```
private void ShowStudent(int pagenum)
{
    pagenum--;
    tb1_0Name.Text = students[pagenum].FullName;
    tb2_0ID.Text = students[pagenum].ID.ToString();
    tb3_0Phone.Text = students[pagenum].PhoneNum;
    tb4_0C1.Text = students[pagenum].C1.ToString();
    tb5_0C2.Text = students[pagenum].C2.ToString();
    tb6_0C3.Text = students[pagenum].C3.ToString();
    tb7_0Avg.Text = students[pagenum].Avg.ToString();
}
```

- Hàm EnableInput(): Kích hoạt các ô nhập liệu và nút thêm học sinh. Đảm bảo rằng các trường nhập liệu và nút thêm học sinh được kích hoạt để người dùng có thể thêm thông tin của học sinh mới

```
private void EnableInput()
{
    tb1_Name.Enabled = true;
    tb2_ID.Enabled = true;
    tb3_Phone.Enabled = true;
    tb4_C1.Enabled = true;
    tb5_C2.Enabled = true;
    tb6_C3.Enabled = true;
    bt2_Add.Enabled = true;
}
```

- Hàm DisableInput(): Vô hiệu hóa các ô nhập liệu và nút thêm học sinh. Sử dụng để ngăn người dùng nhập liệu sau khi đã hoàn thành quá trình nhập liệu hoặc khi không cần thiết

```

1 reference
private void DisableInput()
{
    tb1_Name.ReadOnly = true;
    tb2_ID.ReadOnly = true;
    tb3_Phone.ReadOnly = true;
    tb4_C1.ReadOnly = true;
    tb5_C2.ReadOnly = true;
    tb6_C3.ReadOnly = true;
    bt2_Add.Enabled = false;
    tb1_Name.Enabled = false;
    tb2_ID.Enabled = false;
    tb3_Phone.Enabled = false;
    tb4_C1.Enabled = false;
    tb5_C2.Enabled = false;
    tb6_C3.Enabled = false;
}

```

- Hàm EnableOBt(): Kích hoạt các nút điều hướng trang. Sử dụng để kích hoạt nút "Next" và "Previous" để người dùng có thể chuyển đổi giữa các trang của danh sách học sinh

```

private void EnableOBt()
{
    bt4_Next.Enabled = true;
    bt5_Prev.Enabled = true;
}

```

- Hàm DeserializeJson(string Filepath): Đọc và chuyển đổi chuỗi JSON từ tệp tin thành danh sách các đối tượng sinh viên. Thực hiện việc đọc dữ liệu từ tệp tin JSON và chuyển đổi thành danh sách các đối tượng sinh viên để xử lý

```

1 reference
private List<Bai04_Student> DeserializeJson(string Filepath)
{
    string json = File.ReadAllText(Filepath);
    List<Bai04_Student> students = JsonSerializer.Deserialize<List<Bai04_Student>>(json);
    return students;
}

```

- Hàm CheckLoiNhap(): Kiểm tra và cảnh báo lỗi nếu thông tin nhập vào không hợp lệ. Đảm bảo rằng thông tin được nhập vào các ô nhập liệu là hợp lệ trước khi thêm vào danh sách học sinh

```

I reference
private bool CheckLoiNhap()
{
    if (tb2_ID.Text.Length != 8)
    {
        MessageBox.Show("ID cần phải có 8 chữ số!!");
        return true;
    }
    if (tb3_Phone.Text.Length != 10)
    {
        MessageBox.Show("Số điện thoại cần phải có 10 chữ số!!");
        return true;
    }
    if (tb3_Phone.Text[0] != '0')
    {
        MessageBox.Show("Số điện thoại cần có số 0 đầu tiên");
        return true;
    }
    double num = Double.Parse(tb4_C1.Text);
    if (num < 0 || num > 10)
    {
        MessageBox.Show("Số điểm nhập ở môn 1 không hợp lý");
        return true;
    }
}

```

```

double num = Double.Parse(tb4_C1.Text);
if (num < 0 || num > 10)
{
    MessageBox.Show("Số điểm nhập ở môn 1 không hợp lý");
    return true;
}
num = Double.Parse(tb5_C2.Text);
if (num < 0 || num > 10)
{
    MessageBox.Show("Số điểm nhập ở môn 2 không hợp lý");
    return true;
}
num = Double.Parse(tb6_C3.Text);
if (num < 0 || num > 10)
{
    MessageBox.Show("Số điểm nhập ở môn 3 không hợp lý");
    return true;
}
if (tb1_Name.Text == "" ||
    tb2_ID.Text == "" ||
    tb3_Phone.Text == "" ||
    tb4_C1.Text == "" ||
    tb5_C2.Text == "" ||
    tb6_C3.Text == "")
{
    MessageBox.Show("Xin mời nhập đầy đủ thông tin");
    return true;
}
return false;
}

```

- Hàm kích hoạt khi được click của button “Add”
 - Kiểm tra thông tin nhập
 - Gán tất cả dữ liệu nhập vào đối tượng student sau đó Add đối tượng trên vào List students
 - Kích hoạt và hiển thị bộ đếm lần nhập còn lại cho người dùng để nắm bắt số lượng cần nhập hiện thời

```

private void bt2_Add_Click(object sender, EventArgs e)
{
    if (CheckLoiNhap())
    {
        return;
    }

    Bai04_Student student = new Bai04_Student();
    int num = Int32.Parse(tb8_Count.Text);
    student.FullName = tb1_Name.Text;
    student.ID = Int32.Parse(tb2_ID.Text);
    student.PhoneNum = tb3_Phone.Text;
    student.C1 = float.Parse(tb4_C1.Text);
    student.C2 = float.Parse(tb5_C2.Text);
    student.C3 = float.Parse(tb6_C3.Text);
    students.Add(student);
    tb8_Count.Text = (num - 1).ToString();
    {
        MessageBox.Show("Nhập thành công!!, con lại " + tb8_Count.Text + " lần nhập");
        tb1_Name.Text = "";
        tb2_ID.Text = "";
        tb3_Phone.Text = "";
        tb4_C1.Text = "";
        tb5_C2.Text = "";
        tb6_C3.Text = "";
        tb7_Total.ReadOnly = true;
        tb7_Total.Enabled = false;
    }
    if (tb8_Count.Text == "0")
    {
        MessageBox.Show("Thực hiện đẩy thông tin vào file ...", "Cảnh báo", MessageBoxButtons.OK);
        SerializeJson(students, "input4.txt");
        rtb1_Show.Text = @"Ấn ""Read a file"" để đọc thông tin";
        DisableInput();
        bt3_Read.Enabled = true;
    }
}

```

- Hàm kích hoạt khi có sự kiện thay đổi chuỗi hiển thị trên textbox “Total”
 - Đảm bảo việc nhập số nguyên (số lượng của học sinh cần nhập)
 - Cảnh báo nếu không phải

```

1 reference
private void tb7_Total_TextChanged(object sender, EventArgs e)
{
    if (!Int32.TryParse(tb7_Total.Text, out int temp))
    {
        MessageBox.Show(@"Nhập số nguyên vào thành ""Tong hoc sinh"": ");
        return;
    }
    n = temp;
    rtb1_Show.Text = "";
    tb8_Count.Text = n.ToString();
    MessageBox.Show("Nhập đầy đủ thông tin học sinh vào các ô phía dưới!! ");
    EnableInput();
}

```

- Hàm kích hoạt khi được click của button “Read a File”
 - Đọc thông tin từ tệp
 - Tính điểm trung bình của học sinh
 - In ra bộ thông tin của các học sinh vào richtextbox rtb1

```

1 reference
private void button1_Click(object sender, EventArgs e)
{
    try
    {
        rtb1_Show.Text = "";
        students = DeserializeJson("input4.txt");
        foreach (var student in students)
        {
            student.Avg = (student.C1 + student.C2 + student.C3) / 3;
            rtb1_Show.Text += student.FullName + '\n' + student.ID.ToString() + '\n' + student.PhoneNum + '\n' +
                student.C1.ToString() + '\n' + student.C2.ToString() + '\n' + student.C3.ToString() + '\n' + student.Avg.ToString() + "\n\n";
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    SerializeJson(students, "output4.txt");
    bt3_Read.Enabled = false;
    ShowStudent(page);
    tb8_0Page.Text = page.ToString();
    EnableOBt();
}

```

Hai hàm kích hoạt khi được click của button “Prev” và “Next”: dùng để chuyển trang hiển thị của từng học sinh có trong bộ dữ liệu

```

private void bt4_Next_Click(object sender, EventArgs e)
{
    if (page + 1 <= n)
    {
        page++;
        ShowStudent(page);
        tb8_0Page.Text = (page).ToString();
    }
}

```

```

1 reference
private void bt5_Prev_Click(object sender, EventArgs e)
{
    if (page - 1 > 0)
    {
        page--;
        ShowStudent(page);
        tb8_0Page.Text = (page).ToString();
    }
}

```

Bài 05 – Quản lý phòng vé (phiên bản số 2)

Lấy ý tưởng từ bài 4 - bài thực hành số 1, tuy nhiên các thông tin của phòng vé sẽ được nhập bằng file “input5.txt” với cấu trúc:

<tên phim>

<giá vé chuẩn>

<phòng chiếu>

...

Viết chương trình cho phép nhập và ghi thành file, đọc file để lấy các thông tin cần thiết. Các yêu cầu tương tự, bổ sung tính năng xuất nội dung thống kê theo phim “output5.txt” bao gồm: tên phim, số lượng vé bán ra, số lượng vé tồn, tỉ lệ vé bán ra, doanh thu, xếp hạng doanh thu phòng vé. Trong đó, doanh thu là tổng số tiền thu được khi bán vé của phim đó, xếp hạng doanh thu phòng vé là xếp hạng phim có doanh thu theo thứ tự từ cao đến thấp.

Lưu ý: Tìm hiểu **ProgressBar** và thêm vào khi xuất file “output5.txt” để hỗ trợ người dùng dễ theo dõi tình trạng ứng dụng

- Giao diện của phần nhập thông tin vé

The screenshot shows a Windows application window titled "Bai05". The interface is divided into two main sections. The left section contains a form for entering movie information. It starts with a text box labeled "Write to a File". Below this is a label "Số lượng phim" followed by a text box. Then, there's a label "Tên phim" followed by a text box. Below that, there's a label "Phòng chiếu" followed by three radio buttons labeled "P1", "P2", and "P3". Then, there's a label "Giá vé chuẩn" followed by a text box with the placeholder text "Nhập giá tiền (đơn vị: VND)". Below this, there's a label "Số lần nhập còn lại" followed by a text box and an "Add" button. At the bottom of the left section, there are two buttons: "Thống kê" and "Đặt vé". The right section of the window is a large yellow rectangular area, likely a placeholder for a progress bar or a list of movies.

- Giao diện Đặt Vé

- Class CPhim (tổng hợp thông tin phòng vé)

```
namespace LAB2
{
    internal class CPhim
    {
        public struct CPhong
        {
            public string TenPhong { get; set; }
            public string[] suat { get; set; }
            public CPhong(string tenPhong, string[] suat)
            {
                TenPhong = tenPhong;
                this.suat = suat;
            }
            public CPhong()
            {
                TenPhong = "";
                suat = new[] { "A1", "A2", "A3", "A4", "A5",
                    "B1", "B2", "B3", "B4", "B5", "C1", "C2", "C3", "C4",
                    "C5" };
            }
        }
    }
}
```

```
public string TenPhim { get; set; }
public List<CPhong> Phong { get; set; }
public long GiaVe { get; set; }
public long TongTien { get; set; }
public int Rank { get; set; }

public CPhim()
{
    TenPhim = string.Empty;
    Phong = new List<CPhong>();
    GiaVe = 0;
    TongTien = 0;
    Rank = -1;
}

~CPhim()
{
}

}
```


PHẦN THỐNG KẾ

- Hàm kích hoạt khi được click của button "Write to a file"
 - Kiểm tra xem trường "Total" có được nhập hay không. Nếu không, hiển thị thông báo yêu cầu nhập số lượng phim

```

I reference
private void bt1_WtF_Click(object sender, EventArgs e)
{
    if (tbTotal.Text == "")
    {
        MessageBox.Show("Nhập số lượng phim vào Total");
        tbTotal.Enabled = true;
        return;
    }
}

```

- Hàm 'EnableInput()': Kích hoạt các ô nhập liệu để người dùng có thể nhập thông tin về phim

```

I reference
private void EnableInput()
{
    tbName.Enabled = true;
    cbRoom1.Enabled = true;
    cbRoom2.Enabled = true;
    cbRoom3.Enabled = true;
    tbCost.Enabled = true;
    btAdd.Enabled = true;
}

```

- Hàm kích hoạt khi có chuỗi dữ liệu hiển thị bị thay đổi ở textbox trường "Total"
 - Đảm bảo chỉ nhập số nguyên
 - Kiểm tra xem giá trị nhập vào có phải là số nguyên hay không và hiển thị thông báo nếu không phải

```

private void tbTotal_TextChanged(object sender, EventArgs e)
{
    if (!Int32.TryParse(tbTotal.Text, out int temp))
    {
        MessageBox.Show(@"Nhập số nguyên vào thành "So lượng phim": ");
        return;
    }
    n = temp;
    tbCount.Text = n.ToString();
    MessageBox.Show("Nhập đầy đủ thông tin phim vào các ô phía dưới!! ");

    EnableInput();
}

```

- Hàm kích hoạt khi được click của button "Add": Thêm thông tin của phim vào danh sách sau khi kiểm tra thông tin nhập vào và đảm bảo là thông tin hợp lệ

```
private void btAdd_Click(object sender, EventArgs e)
{
    try {
        CPhim Phim = new CPhim();
        int num = Int32.Parse(tbCount.Text);
        Phim.TenPhim = tbName.Text;
        if (cbRoom1.CheckState == CheckState.Checked)
        {
            Phim.Phong.Add(new CPhim.CPhong { TenPhong = cbRoom1.Text.Trim() });
        }
        if (cbRoom2.CheckState == CheckState.Checked)
        {
            Phim.Phong.Add(new CPhim.CPhong { TenPhong = cbRoom2.Text.Trim() });
        }
        if (cbRoom3.CheckState == CheckState.Checked)
        {
            Phim.Phong.Add(new CPhim.CPhong { TenPhong = cbRoom3.Text.Trim() });
        }
        Phim.GiaVe = Int32.Parse(tbCost.Text);
        tbCount.Text = (num - 1).ToString();
        {
            Phims.Add(Phim);
            MessageBox.Show("Nhập thành công!!, còn lại " + tbCount.Text + " lần nhập");
            tbName.Text = "";
            tbCost.Text = "";
            cbRoom1.CheckState = CheckState.Unchecked;
            cbRoom2.CheckState = CheckState.Unchecked;
            cbRoom3.CheckState = CheckState.Unchecked;
            tbTotal.Enabled = false;
        }
        if (tbCount.Text == "0")
        {
            MessageBox.Show("Thực hiện đầy thông tin vào file ....", "Cảnh báo", MessageBoxButtons.OK);
            SerializeJson(Phims, "input5.txt");
            SerializeJson(Phims, "output5.txt");
            //DisableInput();
            //bt3_Read.Enabled = true;
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}
```

- Hàm 'DeserializeJson(string Filepath)': Đọc và chuyển đổi chuỗi JSON từ tệp tin thành danh sách các đối tượng phim
- Hàm 'SerializeJson(object obj, string Filepath)': Chuyển đổi danh sách các đối tượng phim thành chuỗi JSON và lưu vào tệp tin tại đường dẫn được chỉ định

```
private List<CPhim> DeserializeJson(string Filepath)
{
    string json = File.ReadAllText(Filepath);
    List<CPhim> Phims = JsonSerializer.Deserialize<List<CPhim>>(json);
    return Phims;
}

2 references
private void SerializeJson(object obj, string Filepath)
{
    string json = JsonSerializer.Serialize(obj);
    File.WriteAllText(Filepath, json);
}
```

- Hàm kích hoạt khi được click của button "Đặt vé". Mở form mới để thực hiện quá trình đặt vé cho các phim

```
private void btDatVe_Click(object sender, EventArgs e)
{
    var Form = new Bai05_DatVe();
    Form.Show();
}
```

- Hàm 'RankPhim(List<CPhim> cphimList)': Sắp xếp danh sách phim theo thứ tự giảm dần của tổng doanh thu và gán xếp hạng cho mỗi phim

```
private static void RankPhim(List<CPhim> cphimList)
{
    for (int i = 0; i < cphimList.Count - 1; i++)
    {
        for (int j = 0; j < cphimList.Count - 1 - i; j++)
        {
            if (cphimList[j].TongTien < cphimList[j + 1].TongTien)
            {
                CPhim temp = cphimList[j];
                cphimList[j] = cphimList[j + 1];
                cphimList[j + 1] = temp;
            }
        }
    }

    for (int i = 0; i < cphimList.Count; i++)
    {
        cphimList[i].Rank = i + 1;
    }
}
```

- Hàm kích hoạt khi được click của button "Read"
 - Đọc thông tin từ tệp tin và hiển thị ra giao diện sau khi sắp xếp và tính toán các thông tin liên quan đến doanh thu và xếp hạng của mỗi phim
 - Chèn vào **ProgressBar** progressbar1 để cho người dùng theo dõi quá trình đọc file

```
private void bt3_Read_Click(object sender, EventArgs e)
{
    progressBar1.Value = 0;
    try
    {
        Phims = DeserializeJson("output5.txt");
        RankPhim(Phims);
        rtb1_Show.Clear();
        int progressStep = 100 / Phims.Count;
        foreach (var c in Phims)
        {
            progressBar1.Value += progressStep;
            progressBar1.Refresh();

            rtb1_Show.Text += "Ten phim: " + c.TenPhim + '\n';
            double vetong = 0;
            double veban = 0;
            foreach (CPhim.CPhong c2 in c.Phong)
            {
                vetong += 15;
                veban += c2.suat.Length;
            }
            rtb1_Show.Text += "So ve ban duoc: " + (vetong - veban).ToString() + '\n' + "So ve ton: " + veban.ToString() + '\n';
            rtb1_Show.Text += "Ti le ve ban duoc: " + (((vetong - veban) / vetong) * 100).ToString("0.00") + "%" + '\n';
        }
    }
    catch (JsonException)
    {
        MessageBox.Show("File thông kê (output5.txt) hiện không có dữ liệu để trích xuất hoặc lỗi về Json ");
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}
```

PHÂN ĐẶT VÉ

Thực hiện đặt vé sau đó lưu dữ liệu vé vào output5.txt

```
public partial class Bai05_DatVe : Form
{
    List<CPhim> Phims = new List<CPhim>();
    static CPhim SelectedPhim = new CPhim();
    static CPhim.CPhong SelectedPhong = new CPhim.CPhong();
    public Bai05_DatVe()
    {
        InitializeComponent();
        try
        {
            Phims = DeserializeJson("input5.txt");
            foreach (var Phim in Phims)
            {
                cb2_Movie.Items.Add(Phim.TenPhim);
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
    private List<CPhim> DeserializeJson(string Filepath)
    {
        string json = File.ReadAllText(Filepath);
        List<CPhim> Phims = JsonSerializer.Deserialize<List<CPhim>>(json);
        return Phims;
    }
}
```

```

private void SerializeJson(object obj, string Filepath)
{
    string json = JsonSerializer.Serialize(obj);
    File.WriteAllText(Filepath, json);
}

private void cb2_Movie_SelectionChangeCommitted(object sender, EventArgs e)
{
    cb1_Theater.Items.Clear();
    if (cb2_Movie.SelectedItem != null)
    {
        foreach (CPhim Phim in Phims)
        {
            if (Phim.TenPhim == cb2_Movie.SelectedItem.ToString())
            {
                //MessageBox.Show("1");
                SelectedPhim = Phim;
                break;
            }
        }

        if (SelectedPhim != null && SelectedPhim.Phong != null)
        {
            //MessageBox.Show("2");
            foreach (CPhim.CPhong phong in SelectedPhim.Phong)
            {
                cb1_Theater.Items.Add(phong.TenPhong);
            }
        }
        cb1_Theater.Enabled = true;
    }
}

private void cb1_Theater_SelectionChangeCommitted(object sender, EventArgs e)
{
    clb1_Seats.Items.Clear();
    if (cb1_Theater.SelectedItem != null && SelectedPhim != null && SelectedPhim.Phong != null)
    {
        //MessageBox.Show("3"); ;
        foreach (CPhim.CPhong c in SelectedPhim.Phong)
        {
            if (c.TenPhong == cb1_Theater.SelectedItem.ToString())
            {
                //MessageBox.Show("4"); ;
                SelectedPhong = c;
                break;
            }
        }

        if (SelectedPhong.suat != null)
        {
            clb1_Seats.Items.AddRange(SelectedPhong.suat);
        }
    }
}

private void clb1_Seats_ItemCheck(object sender, ItemCheckEventArgs e)
{
    if (clb1_Seats.CheckedItems.Count == 1)
    {
        if (e.NewValue == CheckState.Unchecked)
            bt1_Confirm.Enabled = false;
    }
}

```

```

    }
    else
        bt1_Confirm.Enabled = true;
}

private void bt1_Confirm_Click(object sender, EventArgs e)
{
    long tong = 0; // tính tổng tiền từ checked seats
    long cost = SelectedPhim.GiaVe;
    foreach (string c in clb1_Seats.CheckedItems)
    {
        if (new[] { "A1", "A5", "B1", "B5", "C1", "C5" }.Contains(c))
        {
            tong += cost * 1 / 4;
        }
        else if (new[] { "A2", "A3", "A4", "C2", "C3", "C4" }.Contains(c))
        {
            tong += cost * 1;
        }
        else
        {
            tong += cost * 2;
        }
    }
    string s = "Ho va ten: " + tb1_Name.Text;
    s += System.Environment.NewLine + "Các vé đã chọn: ";
    foreach (string c in clb1_Seats.CheckedItems)
        s += c + " ";
    s += System.Environment.NewLine;
    s += "Phòng chiếu: " + cb1_Theater.Text;
    s += System.Environment.NewLine;
    s += "Số tiền phải trả: " + tong.ToString();

    if (MessageBox.Show(s, "Warning !!", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.No) //
        thông báo lần cuối chắc chắn hay ko ?
    {
        return;
    }
    else
    {
        MessageBox.Show("Bạn đã đặt vé thành công.", "Congratulations", MessageBoxButtons.OK,
            MessageBoxIcon.Information);
        SelectedPhim.TongTien += tong;
        HashSet<string> temp = SelectedPhong.suat.ToHashSet();
        foreach (string c in clb1_Seats.CheckedItems)
        {
            temp.Remove(c);
        }
        SelectedPhong.suat = temp.ToArray();
        foreach (var c in SelectedPhim.Phong)
        {
            if (c.TenPhong == SelectedPhong.TenPhong)
            {
                SelectedPhim.Phong.Remove(c);
                SelectedPhim.Phong.Add(SelectedPhong);
                break;
            }
        }
        foreach (var c in Phims)
        {
            if (SelectedPhim.TenPhim == c.TenPhim)
            {

```

```

        Phims.Remove(c);
        Phims.Add(SelectedPhim);
        break;
    }
}
SerializeJson(Phims, "output5.txt");
// blank các ô -> đẹp
tb1_Name.Text = "";
cb2_Movie.Text = "";
cb1_Theater.Text = "";
cb1_Theater.Enabled = false;
SelectedPhim = null;
SelectedPhong = new CPhim.CPhong();
cb1_Theater.Items.Clear();
clb1_Seats.Items.Clear();
}
}

private void bt2_Reset_Click(object sender, EventArgs e)
{
    tb1_Name.Text = "";
    cb2_Movie.Text = "";
    cb1_Theater.Text = "";
    cb1_Theater.Enabled = false;
}

private void bt3_Exit_Click(object sender, EventArgs e)
{
    this.Close();
}
}

```

Bài 06 – Hôm nay ăn gì? (phiên bản số 2)

Lấy ý tưởng từ bài 8 - bài thực hành số 1, tuy nhiên cách thức nhập, đọc thông tin về món ăn được cấu trúc ở dạng như sau:

MonAn (**IDMA**, TenMonAn, HinhAnh, IDNCC)

NguoIDung (**IDNCC**, HoVaTen, QuyenHan)

Tìm hiểu về các sử dụng **SQLite**, thực hiện nhập dữ liệu trên vào cơ sở dữ liệu **SQLite**, thực hiện các câu truy vấn để trích xuất thông tin. Chương trình cho phép hiển thị danh sách các món ăn hiện tại được đọc từ cơ sở dữ liệu (sử dụng **ListView** hay **TreeView**...).

Kết quả cuối cùng của ứng dụng là ngẫu nhiên chọn ra 1 món ăn, hình ảnh của món ăn đó và tên người đóng góp món ăn.

Giao diện

```
private SQLiteConnection conn;
private Random rnd;
static string connstring = "Bai07_Database.db";
static int CountMonAn = 1;
static int CountNguoiDung = 1;

1 reference
public Bai06()
{
    InitializeComponent();
    CheckIfDataExists("Bai07_Database.db");
    dataGridView2.DataSource = LoadJointList(connstring);
}
```

Các biến static và hàm khởi tạo Form

- Hàm kích hoạt khi được click của button "button1": hàm này thực hiện việc thêm một món ăn mới và một người dùng mới vào **cơ sở dữ liệu SQLite**. Sau đó, cập nhật hai DataGridView (dataGridView1 và dataGridView2) để hiển thị danh sách món ăn và danh sách gắn kết giữa món ăn và người dùng

```
private void button1_Click(object sender, EventArgs e)
{
    using (SQLiteConnection conn = new SQLiteConnection("Data source=" + connstring + ";Version = 3"))
    {
        //conn.Open();
        //SQLiteCommand delete = new SQLiteCommand("DELETE FROM MONAN ", conn);
        //delete.ExecuteNonQuery();
        //delete = new SQLiteCommand("DELETE FROM NGUOIDUNG ", conn);
        //delete.ExecuteNonQuery();
        //conn.Close();
        string query = "INSERT INTO MonAn ([IDMA], [TenMonAn], [HinhAnh], [IDNCC]) VALUES (@IDMA, @TenMonAn, @HinhAnh, @IDNCC)";
        using (SQLiteCommand cmd = new SQLiteCommand(query, conn))
        {
            conn.Open();

            cmd.Parameters.AddWithValue("@IDMA", "IDMA" + CountMonAn.ToString());
            cmd.Parameters.AddWithValue("@TenMonAn", textBox1.Text);
            cmd.Parameters.AddWithValue("@HinhAnh", textBox2.Text);
            cmd.Parameters.AddWithValue("@IDNCC", "IDNCC" + CountNguoiDung.ToString());

            cmd.ExecuteNonQuery();
            CountMonAn++;
        }
        string query1 = "INSERT INTO NguoiDung ([IDNCC], [HoVaTen], [QuyenHan]) VALUES (@IDNCC, @HoVaTen, @QuyenHan)";
        using (SQLiteCommand cmd = new SQLiteCommand(query1, conn))
        {
            cmd.Parameters.AddWithValue("@IDNCC", "IDNCC" + CountNguoiDung.ToString());
            cmd.Parameters.AddWithValue("@HoVaTen", textBox3.Text);
            cmd.Parameters.AddWithValue("@QuyenHan", textBox4.Text);

            cmd.ExecuteNonQuery();
            CountNguoiDung++;
        }
    }
    dataGridView1.DataSource = LoadMonAnList(connstring);
    dataGridView2.DataSource = LoadJointList(connstring);
}
```

- Hàm CheckIfDataExists(string filepath): kiểm tra xem tập tin cơ sở dữ liệu đã tồn tại hay không
 - Nếu không tồn tại, hàm này sẽ tạo một tập tin mới và tạo các bảng cần thiết trong cơ sở dữ liệu (bảng MonAn và NguoiDung)
 - Nếu tập tin đã tồn tại, hàm sẽ tải dữ liệu vào dataGridView1 và cập nhật số lượng người dùng và món ăn

```
1 reference
private void CheckIfDataExists(string filepath)
{
    if (!File.Exists(filepath))
    {
        SQLiteConnection.CreateFile(filepath);
        using (SQLiteConnection conn = new SQLiteConnection("Data source=" + filepath + ";Version = 3"))
        {
            string command_createtb_MonAn = "CREATE TABLE MonAn (IDMA TEXT (10) PRIMARY KEY NOT NULL UNIQUE, TenMonAn TEXT NOT NULL, " +
            " HinhAnh TEXT, IDNCC TEXT REFERENCES NguoiDung (IDNCC) );";
            using (SQLiteCommand cmd = new SQLiteCommand(command_createtb_MonAn, conn))
            {
                conn.Open();
                cmd.ExecuteNonQuery();
            }
        }
    }
    else
    {
        dataGridView1.DataSource = LoadMonAnList(filepath);
        CountNguoiDung = LoadNguoiDungList(filepath).Rows.Count + 1;
        CountMonAn = dataGridView1.RowCount + 1;
    }
}
```

- Hàm LoadNguoiDungList(string filepath): tải danh sách người dùng từ cơ sở dữ liệu và trả về một DataTable chứa dữ liệu.

```

2 references
private DataTable LoadNguoiDungList(string filepath)
{
    DataTable dta = new DataTable();

    using (SQLiteConnection conn = new SQLiteConnection("Data source=" + filepath + ";Version = 3"))
    {
        string query = "SELECT * FROM NguoiDung";
        using (SQLiteCommand cmd = new SQLiteCommand(query, conn))
        {
            conn.Open();

            SQLiteDataReader reader = cmd.ExecuteReader();

            dta.Load(reader);
        }
    }
    return dta;
}

```

- Hàm LoadMonAnList(string filepath): Tải danh sách món ăn từ cơ sở dữ liệu và trả về một DataTable chứa dữ liệu.

```

2 references
private DataTable LoadMonAnList(string filepath)
{
    DataTable dta = new DataTable();

    using (SQLiteConnection conn = new SQLiteConnection("Data source=" + filepath + ";Version = 3"))
    {
        string query = "SELECT IDMA, TENMONAN, HINHANH, IDNCC FROM MonAn";
        using (SQLiteCommand cmd = new SQLiteCommand(query, conn))
        {
            conn.Open();

            SQLiteDataReader reader = cmd.ExecuteReader();

            dta.Load(reader);
        }
    }
    return dta;
}
2 references

```

- Hàm LoadJointList(string filepath): Tải danh sách gắn kết giữa món ăn và người dùng từ cơ sở dữ liệu và trả về một DataTable chứa dữ liệu.

```

3 references
private DataTable LoadJointList(string filepath)
{
    DataTable dta = new DataTable();

    using (SQLiteConnection conn = new SQLiteConnection("Data source=" + filepath + ";Version = 3"))
    {
        string query = "SELECT TenMonAn, HinhAnh, HoVaTen FROM MonAn JOIN NguoiDung ON MonAn.IDNCC = NguoiDung.IDNCC";
        using (SQLiteCommand cmd = new SQLiteCommand(query, conn))
        {
            conn.Open();

            SQLiteDataReader reader = cmd.ExecuteReader();

            dta.Load(reader);
        }
    }

    return dta;
}

```

- Hai hàm kích hoạt khi được click của 2 button “Danh sách người dùng” và “Danh sách món ăn”: tải danh sách các món ăn/ người dùng từ cơ sở dữ liệu và hiển thị trên dataGridView1

```

1 reference
private void button4_Click(object sender, EventArgs e)
{
    dataGridView1.DataSource = LoadNguoiDungList(connstring);
}

```

```

1 reference
private void button3_Click(object sender, EventArgs e)
{
    dataGridView1.DataSource = LoadMonAnList(connstring);
}

```

1 reference

- Hàm IsImageFile(string filePath): kiểm tra xem một tập tin có phải là tập tin hình ảnh hay không bằng cách kiểm tra phần mở rộng của tập tin

```

private bool IsImageFile(string filePath)
{
    string extension = Path.GetExtension(filePath).ToLower(); // lay duoi file
    return extension == ".jpg" || extension == ".jpeg" || extension == ".png" || extension == ".gif" || extension == ".bmp";
}

```

- Hàm kích hoạt khi được click của button “Random”: tải danh sách gắn kết giữa món ăn và người dùng, sau đó chọn một hàng ngẫu nhiên từ DataTable và hiển thị hình ảnh và thông tin về món ăn và người cung cấp trong các điều khiển hình ảnh và MessageBox

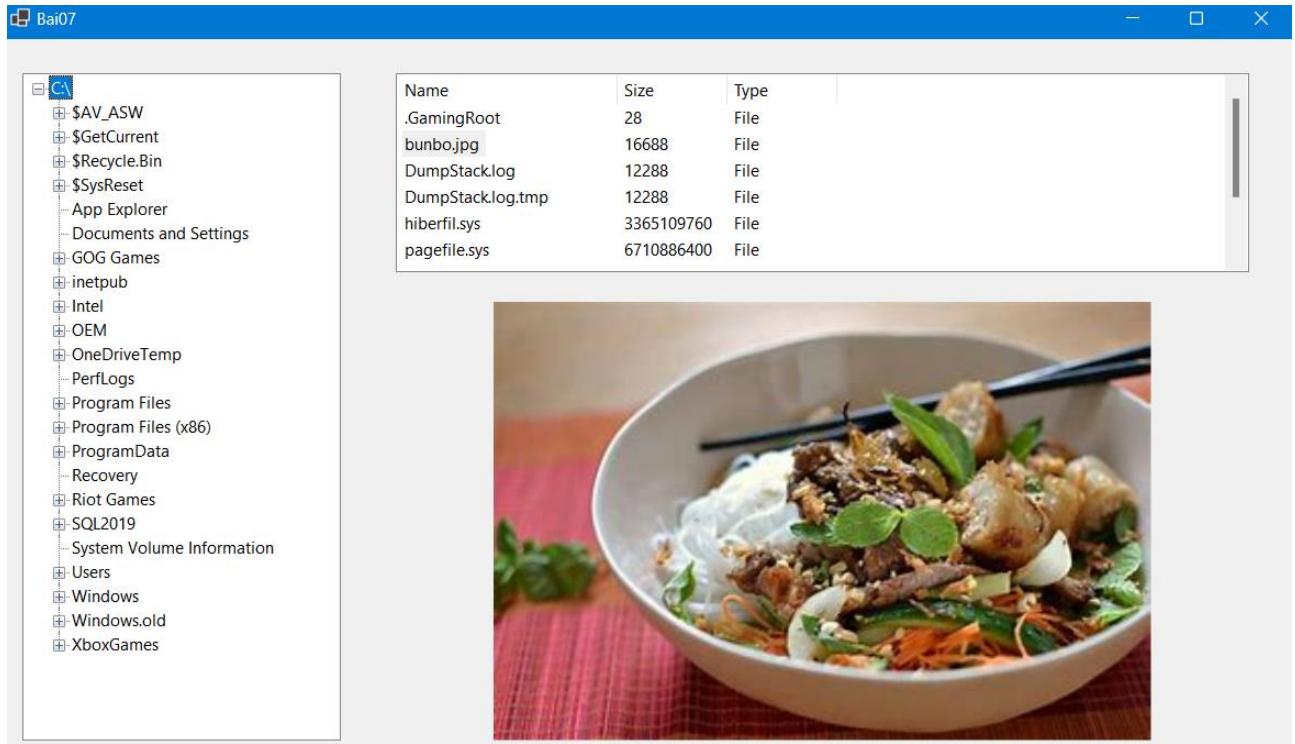
```

1 reference
private void button2_Click(object sender, EventArgs e)
{
    DataTable dt = LoadJointList(connstring);
    DataRow row = dt.Rows[RandomNumberGenerator.GetInt32(dt.Rows.Count)];
    string s = row["HinhAnh"].ToString();
    if (IsImageFile(s)) // check co phai image ko
    {
        pictureBox1.Image = Image.FromFile(s);
    }
    MessageBox.Show($"Món ăn hôm nay là {row["TenMonAn"]}, Hình ảnh = {row["HinhAnh"]}, Người cung cấp = {row["HoVaTen"]}");
}

```

Bài 07 – Duyệt thư mục

Viết ứng dụng cho phép duyệt tất cả file có trong máy tính, hiển thị danh sách cái file, thư mục. Cho phép đi đến folder tiếp theo khi nhấp đúp chuột và hiển thị nội dung của file khi bấm chọn.



Giao diện

- Hàm Bai07(): phương thức khởi tạo của lớp Bai07, được kích hoạt khi một đối tượng của lớp này được tạo ra
 - Gọi InitializeComponent() để khởi tạo các thành phần giao diện người dùng được thiết kế bằng giao diện người dùng Windows Form
 - Hiển thị một hộp thoại cảnh báo thông báo rằng form đang được xử lý
 - Khởi tạo ListView (FileListView) để hiển thị thông tin về các tập tin
 - Khởi tạo TreeView (FolderTreeView) để hiển thị cây thư mục của các ổ đĩa trên máy tính

```

public Bai07()
{
    InitializeComponent();
    MessageBox.Show("Doi 1 ti form dang duoc xu ly", "Warning");
    // khoi tao list view
    FileListView.View = View.Details;
    FileListView.Columns.Add("Name", 200);
    FileListView.Columns.Add("Size", 100);
    FileListView.Columns.Add("Type", 100);

    // khoi tao treeview (files)
    DriveInfo[] drives = DriveInfo.GetDrives();

    foreach (DriveInfo drive in drives)
    {
        TreeNode driveNode = new TreeNode(drive.Name); // tao treenode
        driveNode.Tag = drive.RootDirectory;
        FolderTreeView.Nodes.Add(driveNode);
        CungCapInfoTreeView(drive.RootDirectory, driveNode);
    }
}

```

- Hàm CungCapInfoTreeView(DirectoryInfo directory, TreeNode parentNode): sử dụng để cung cấp thông tin về cây thư mục trong TreeView → đệ quy qua tất cả các thư mục con và thêm chúng vào TreeView

2 references

```

private void CungCapInfoTreeView(DirectoryInfo directory, TreeNode parentNode)
{
    try
    {
        foreach (DirectoryInfo dir in directory.GetDirectories())
        {
            TreeNode node = new TreeNode(dir.Name);
            node.Tag = dir;
            parentNode.Nodes.Add(node);
            CungCapInfoTreeView(dir, node);
        }
    }
    catch (UnauthorizedAccessException) //bat loi
    {
    }
}

```

- Hàm `FolderTreeView_NodeMouseDoubleClick(object sender, TreeNodeMouseClickEventArgs e)`: được kích hoạt khi người dùng nhấp đúp chuột vào một nút thư mục trong `TreeView`
 - Lấy thư mục được chọn
 - Xóa tất cả các mục trong `ListView`
 - Liệt kê các tập tin trong thư mục được chọn và thêm chúng vào `ListView`

```
1 reference
private void FolderTreeView_NodeMouseDoubleClick(object sender, TreeNodeMouseClickEventArgs e)
{
    TreeNode SelectedNode = e.Node;
    if (SelectedNode.Tag is DirectoryInfo) //check tag có gan đường dẫn?
    {
        DirectoryInfo SelectedDirectory = (DirectoryInfo)SelectedNode.Tag;
        FileListView.Items.Clear();
        try
        {
            foreach (var file in SelectedDirectory.GetFiles())
            {
                ListViewItem item = new ListViewItem(file.Name);
                item.SubItems.Add(file.Length.ToString());
                item.SubItems.Add("File");
                FileListView.Items.Add(item);
            }
        }
        catch (UnauthorizedAccessException) // bắt lỗi
        {
        }
    }
}
```

- Hàm `FileListView_SelectedIndexChanged(object sender, EventArgs e)`: được kích hoạt khi một mục được chọn trong `ListView`
 - Kiểm tra xem có mục nào được chọn không
 - Lấy đường dẫn của tập tin được chọn
 - Nếu tập tin tồn tại, kiểm tra xem nó có phải là một hình ảnh hay không. Nếu là hình ảnh, hiển thị hình ảnh trong `pictureBox1`. Nếu không phải → đọc nội dung của tập tin và hiển thị nội dung trong một hộp thoại cảnh báo


```

1 reference
private void FileListView_SelectedIndexChanged(object sender, EventArgs e)
{
    if (FileListView.SelectedItems.Count > 0)
    {
        string SelectedFile = FileListView.SelectedItems[0].Text;
        string SelectedPath = Path.Combine(FolderTreeView.SelectedNode.FullPath, SelectedFile);

        if (File.Exists(SelectedPath))
        {
            try
            {
                if (IsImageFile(SelectedPath)) // check có phải image ko
                {
                    pictureBox1.Image = Image.FromFile(SelectedPath);
                }
                else
                {
                    using (var streamReader = new StreamReader(SelectedPath))
                    {
                        string content = streamReader.ReadToEnd();
                        MessageBox.Show(content, "Nội dung file");
                    }
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Lỗi: {ex.Message}", "Warning");
            }
        }
    }
}

```

- Hàm IsImageFile(string filePath): kiểm tra xem một tập tin có phải là hình ảnh không bằng cách kiểm tra phần mở rộng của tập tin
 - **true** nếu tập tin là một hình ảnh (có phần mở rộng là .jpg, .jpeg, .png, .gif hoặc .bmp)
 - **false** nếu không phải

```

1 reference
private bool IsImageFile(string filePath)
{
    string extension = Path.GetExtension(filePath).ToLower(); // lấy đuôi file
    return extension == ".jpg" || extension == ".jpeg" || extension == ".png" || extension == ".gif" || extension == ".bmp";
}

```