

Lab 01:

## LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG C#

### Với ứng dụng CONSOLE

#### A. MỤC TIÊU:

- ✓ Hướng dẫn sinh viên làm quen với ngôn ngữ lập trình C#: qua việc viết các ứng dụng **Console** trong vs. NET 2015/2017/ 2019.
- ✓ Xây dựng các lớp, tạo đối tượng, truy xuất các phương thức, ...
- ✓ Soạn thảo mã nguồn, biên dịch, debug, thực thi chương trình...
- ✓ Kế thừa trong lập trình hướng đối tượng trên C#.
- ✓ Tìm hiểu về sử dụng thư viện **LINQ** của .NET
- ✓ Khuyến khích sinh viên sử dụng chuẩn viết code C# ( *C# Coding Convetion* )
- ✓ Ngôn ngữ C#

- Kiểu dữ liệu: `bool`, `decimal`, `double`, `float`, `int`, `string`, `DateTime`, `bool?`, `decimal?`, `double?`, `float?`, `int?`, `long?`, `DateTime?`, `object`, `var`, `dynamic`...
- Chuyển đổi/ép kiểu dữ liệu: `as`, `is`, Thư viện `Convert`
- Vòng lặp: `for`, `foreach`, `while`, `do...while` và Lệnh điều khiển `break`, `continue`, `return`
- Một số phương thức của thư viện **Console**

```

Console.Write(<giá trị cần in ra màn hình>);
Console.WriteLine(); // Sử dụng lệnh in ra màn hình có xuống dòng
Console.ReadLine(); // Đọc dữ liệu chuỗi từ bàn phím cho đến khi gặp ký tự xuống dòng
Console.ReadKey(); // Dừng màn hình để xem kết quả.
    
```

- Coding Convention C#: Đưa ra các quy ước khi coding với ngôn ngữ lập trình C#, với các quy tắc này giúp tiết kiệm thời gian rất trong quá trình phát triển và bảo trì phần mềm.

Ví dụ: Đặt tên class dùng danh từ, đối với phương thức dùng động từ ...

Tên biến, tên phương thức thể hiện được ý nghĩa

Nên comment những đoạn code khó hiểu hoặc có chức năng đặc biệt

#### B. BÀI TẬP

**Bài tập 1A:** Viết chương trình cho phép người dùng nhập vào tổng số N sinh viên, mỗi sinh viên phải nhập vào các thông tin của sinh viên( Mã số Sinh viên, Họ tên sinh viên, Điểm TB và tên Khoa) sau đó lần lượt tạo các đối tượng sinh viên và đưa vào mảng **Student** theo những thông tin do user nhập vào (dùng vòng lặp for). Cuối cùng xuất ra danh sách chi tiết thông tin sinh viên.

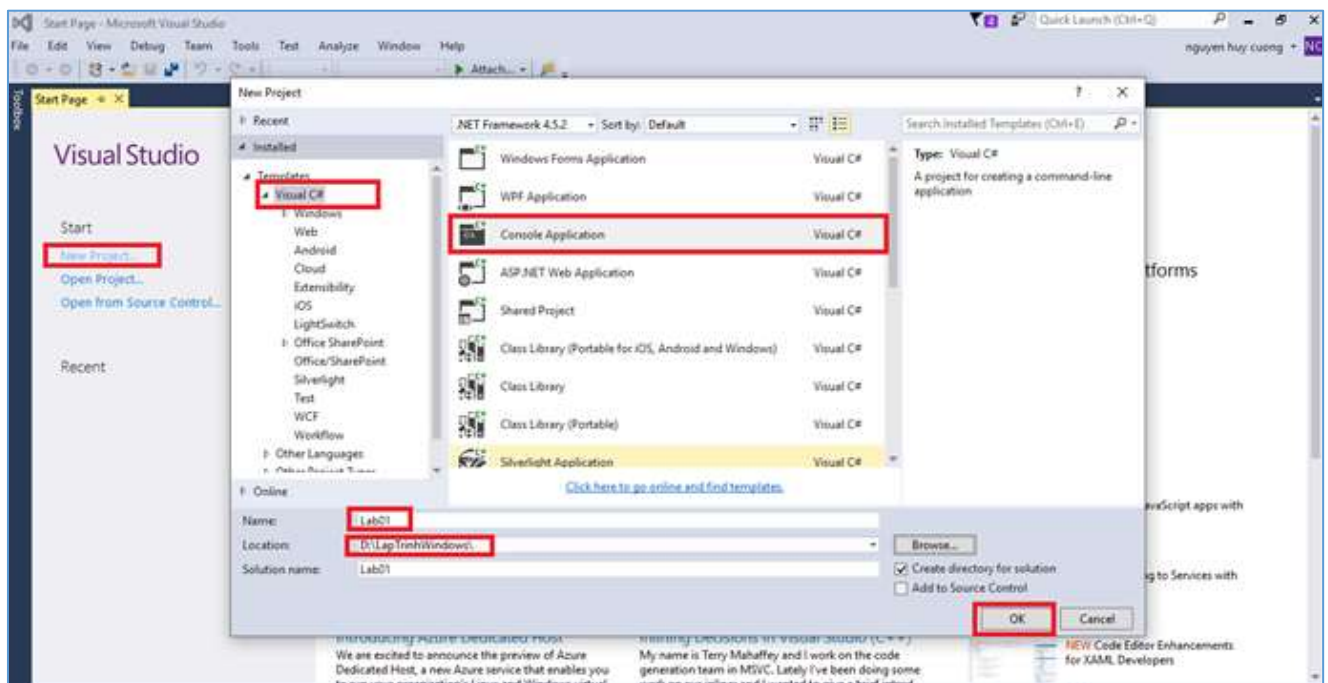
### Yêu cầu:

- ✓ Sinh viên xây dựng chương trình theo nội dung mô tả bên trên. Không viết lệnh nhập và xuất danh sách sinh viên trực tiếp trong hàm **Main()** mà hãy viết hai phương thức nhập và xuất thông tin sinh viên ở lớp **Student**.
- ✓ Compile & Build chương trình.
- ✓ Run chương trình ở hai chế độ debug và không debug.
- ✓ Chạy từng bước chương trình trong chế độ debug: dùng breakpoint hoặc chạy từng dòng lệnh. Kiểm tra những giá trị của các biến trong chương trình ở cửa sổ Watch.

### Hướng dẫn:

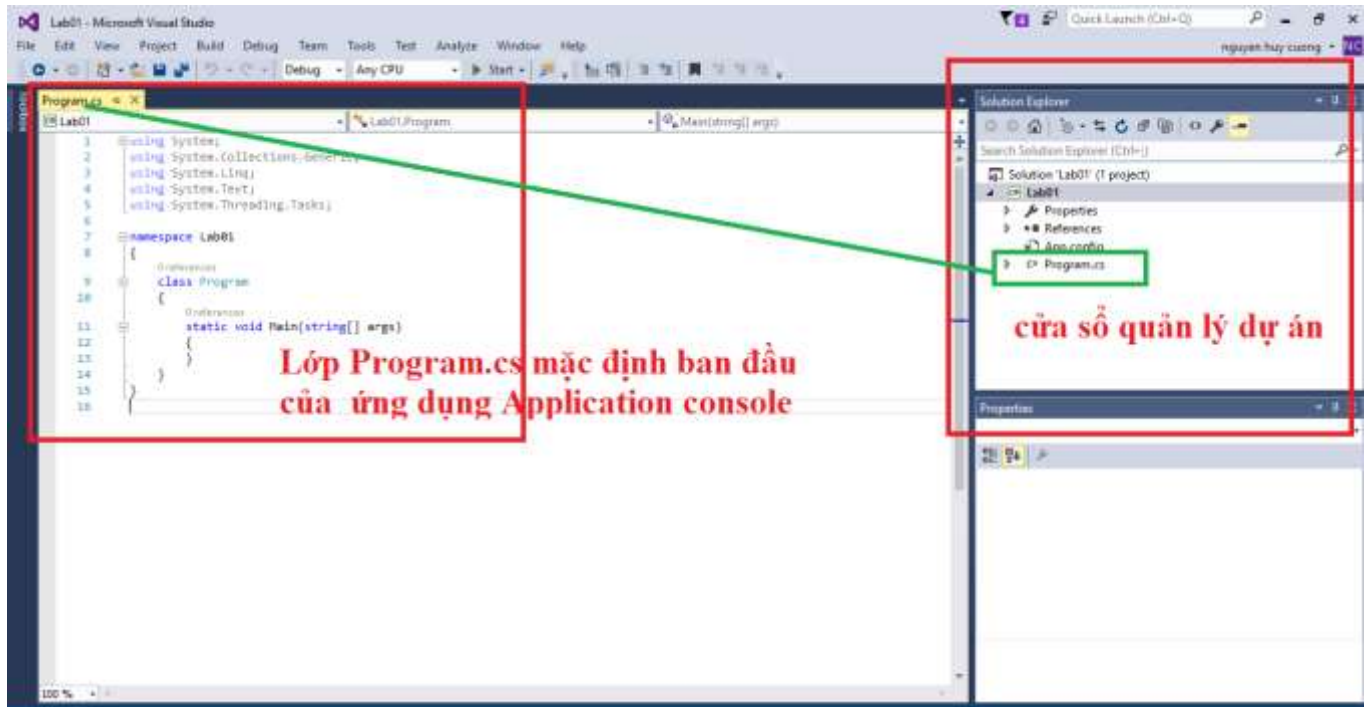
#### Bước 1: Tạo project mới trong VS 2015

- Mở Visual Studio 2015, chọn **New Project**, Chọn ngôn ngữ **Visual C#**, Loại project là **Console Application**
- Đặt tên project là **Lab01** và lưu



Hình 1: Tạo một project console c# mới trong VS .NET 2015

- Click OK để đồng ý tạo project, kết quả chúng ta được một ứng dụng console như sau:



Hình 2: Màn hình làm việc của Project

- Lưu lại dự Án File/Save All

**Bước 2:** Tạo class Student, Khai báo thuộc tính, phương thức (Right Click vào Lab01 Solution, chọn Add/ Class đặt tên là Student.cs)

```
class Student
{
    //1. Tao thuoc tinh
    private string studentID; //mã số sinh viên
    private string fullName; // họ tên sinh viên
    private float averageScore; //điểm trung bình
    private string faculty; //khoa

    //2. Tao cac Property (tip: chọn Quick Actions And Refactorings từ thuộc tính)
    public string StudentID
    {
        get
        {
            return studentID;
        }
        set
        {
            studentID = value;
        }
    }
    public string FullName
    {
        get
        {
            return fullName;
        }
        set
        {
            fullName = value;
        }
    }
}
```

```

public float AverageScore
{
    get
    {
        return averageScore;
    }

    set
    {
        averageScore = value;
    }
}
public string Faculty
{
    get
    {
        return faculty;
    }

    set
    {
        faculty = value;
    }
}
// 3. Tao constructor mặc định không tham số
public Student()
{
}
// 4. Tao constuctor có tham số
public Student(string id, string name, float score, string faculty)
{
    StudentID = id;
    FullName = name;
    AverageScore = score;
    Faculty = faculty;
}
// 5. Viết các phương thức nhập, xuất sinh viên
public void Input()
{
    Console.Write("Nhập MSSV:");
    StudentID = Console.ReadLine();

    Console.Write("Nhập Họ tên Sinh viên:");
    FullName = Console.ReadLine();

    Console.Write("Nhập Điểm TB:");
    AverageScore = float.Parse(Console.ReadLine()); //ép sang kiểu float

    Console.Write("Nhập Khoa:");
    Faculty = Console.ReadLine();
}
public void Show()
{
    Console.WriteLine("MSSV:{0} Họ Tên:{1} Khoa:{2} ĐiểmTB:{3}", this.StudentID,
this.fullName, this.Faculty, this.AverageScore);
}
}

```

**Bước 3:** Viết code trong hàm **Main()** để cho phép người dùng nhập vào tổng số N sinh viên

```

static void Main(string[] args)
{
    Console.OutputEncoding = Encoding.Unicode; //Để sử dụng tiếng việt
    Console.InputEncoding = Encoding.Unicode;
    //Nhập tổng số sinh viên N, Convert kiểu dữ liệu sang biến N kiểu int
}

```

```

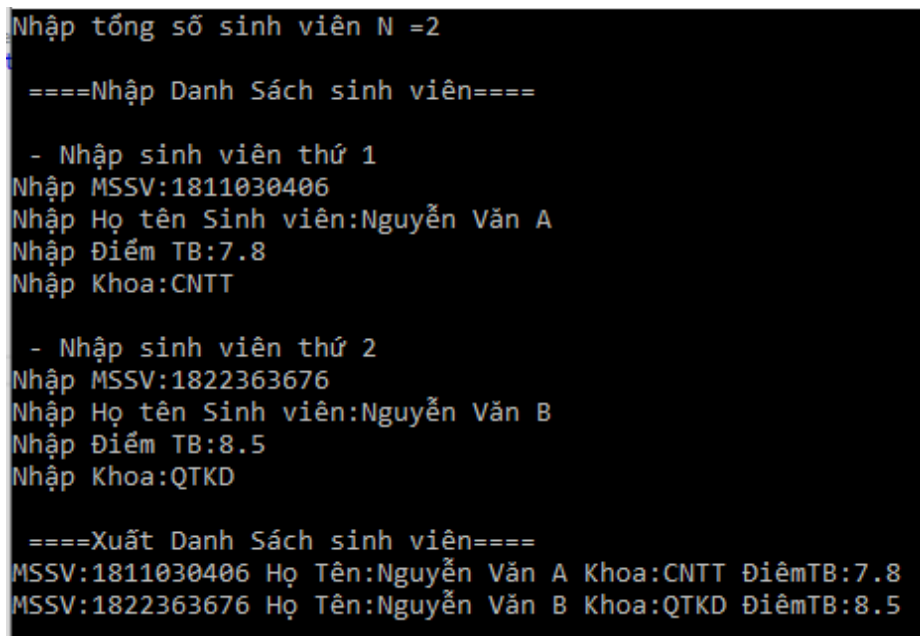
Console.WriteLine("Nhập tổng số sinh viên N =");
int N = Convert.ToInt32(Console.ReadLine());

Student[] arrStudents = new Student[N];
Console.WriteLine("\n ====Nhập Danh Sách sinh viên====");
for (int i = 0; i < N; i++)
{
    Console.WriteLine("\n - Nhập sinh viên thứ {0}", i+1);
    arrStudents[i] = new Student();
    arrStudents[i].Input();
}

Console.WriteLine("\n ====Xuất Danh Sách sinh viên====");
foreach (Student sv in arrStudents)
{
    sv.Show();
}
Console.ReadKey(); //Dừng màn hình kiểm tra kết quả
}
    
```

**Bước 4:** Biên dịch và chạy chương trình

- ✓ Để biên dịch chương trình chọn menu **Build**, rồi chọn **Build Solution** (hoặc dùng phím tắt **F6** hoặc click phải vào Solution Explorer chọn **Build**). VS.NET sẽ thông báo biên dịch thành công hay gặp lỗi cú pháp.
- ✓ Để chạy chương trình click vào biểu tượng **Start** trên VS (hoặc chọn **Debug / Debug Without Debugging** hoặc dùng **Ctrl +F5**)



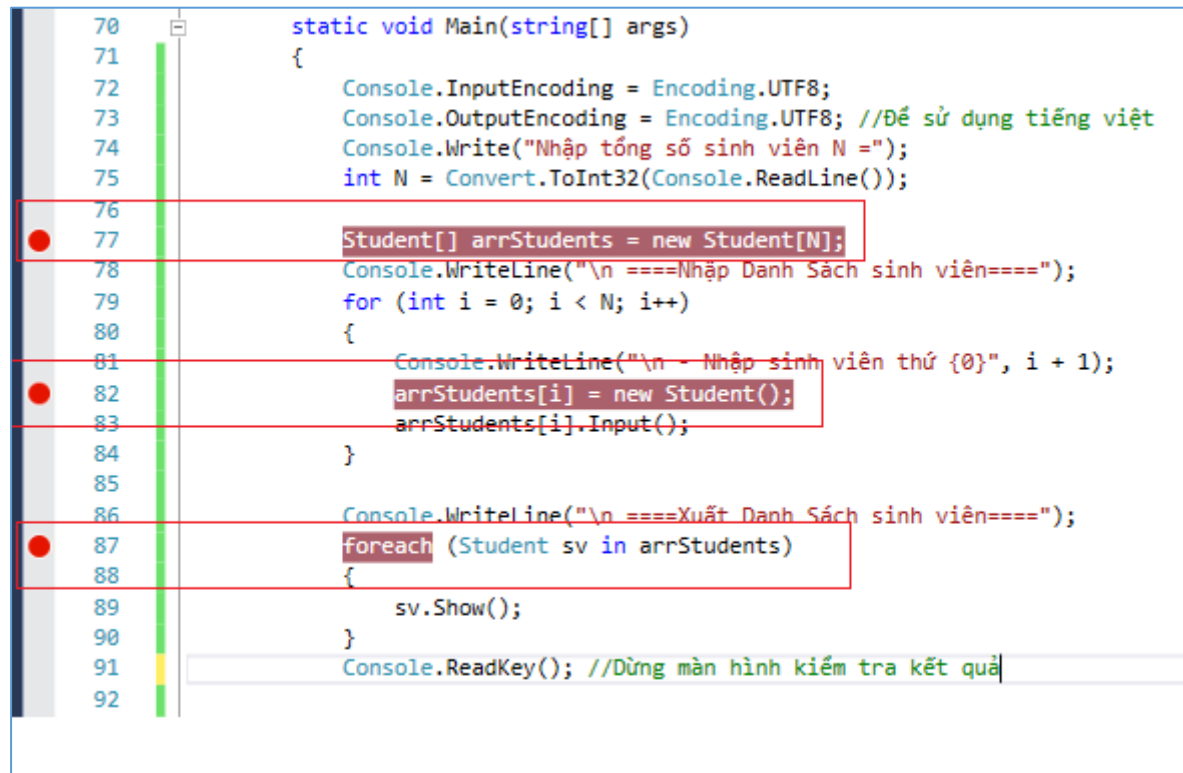
```

Nhập tổng số sinh viên N =2
====Nhập Danh Sách sinh viên====
- Nhập sinh viên thứ 1
Nhập MSSV:1811030406
Nhập Họ tên Sinh viên:Nguyễn Văn A
Nhập Điểm TB:7.8
Nhập Khoa:CNTT
- Nhập sinh viên thứ 2
Nhập MSSV:1822363676
Nhập Họ tên Sinh viên:Nguyễn Văn B
Nhập Điểm TB:8.5
Nhập Khoa:QTKD
====Xuất Danh Sách sinh viên====
MSSV:1811030406 Họ Tên:Nguyễn Văn A Khoa:CNTT ĐiểmTB:7.8
MSSV:1822363676 Họ Tên:Nguyễn Văn B Khoa:QTKD ĐiểmTB:8.5
    
```

Hình 3: Kết quả màn hình console chương trình khi thực hiện

- ✓ Để Debug chương trình sử dụng F5(hoặc vào Debug / Start Debugging) kết hợp với việc đặt các breakpoint
  - Tạo breakpoint bằng cách đơn giản nhất là click chuột vào đầu dòng code (như trong hình). Để hủy breakpoint, chỉ cần click chuột vào breakpoint đó một lần nữa. Ngoài ra

các bạn cũng có thể tạo/hủy breakpoint bằng phím F9.



Hình 4: Đặt các breakpoint hỗ trợ quá trình Debug

- Nhấn F5 để bắt đầu Debug, Tại các vị trí BreakPoint chương trình sẽ dừng lại. Muốn kiểm tra được sự thay đổi giá trị của các biến.
  - Sử dụng Step Over / Step Into / Step Out trong quá trình Debug chương trình
- Step Over (F10):** Chạy step by step, lướt qua hàm (chỉ nhận giá trị return của hàm).
- Step Into (F11):** Chạy step by step, đi vào nội dung của các hàm con.

**Step Out (continue):** “Nhảy” đến breakpoint kế tiếp. Nếu không còn breakpoint nào thì sẽ kết thúc debug. Ngoài ra nó còn có chức năng chạy lướt qua hàm con hiện tại.

### Bài tập 1B:

- ✓ Sử dụng Collection là **List** để chứa danh sách sinh viên thay thế cho mảng sinh viên trong bài tập 1. Chạy lại chương trình theo yêu cầu trên.
- ✓ Viết tiếp chương trình thực hiện một số yêu cầu sau bằng cách sử dụng LINQ. (Khuyến khích sinh viên viết ra các hàm cho dễ quản lý, những phần chung nên gom vào một hàm)

1.1 Xuất ra thông tin của các SV đều thuộc khoa “CNTT” (nếu có)

1.2 Xuất ra thông tin các sinh viên có điểm TB lớn hơn bằng 5 (nếu có).

1.3 Xuất ra danh sách các sinh viên được sắp xếp theo điểm trung bình tăng dần

1.4 Xuất ra danh sách sinh viên có điểm TB lớn hơn bằng 5 và thuộc khoa “CNTT” (nếu có)

1.5 Xuất ra danh sách sinh viên có điểm trung bình cao nhất và thuộc khoa “CNTT” (nếu có)

## Hướng Dẫn

1. Sử dụng **List** để thay thế mảng (phương thức **add** để đưa đối tượng vào list)

```
static void Main(string[] args)
{
    Console.OutputEncoding = Encoding.Unicode;
    Console.InputEncoding = Encoding.Unicode; //Để sử dụng tiếng việt
    List<Student> listStudents = new List<Student>();
    Console.WriteLine("Nhập tổng số sinh viên N =");
    int N = Convert.ToInt32(Console.ReadLine());

    Console.WriteLine("\n ====Nhập Danh Sách sinh viên====");
    for (int i = 0; i < N; i++)
    {
        Console.WriteLine("\n - Nhập sinh viên thứ {0}", i + 1);
        Student temp = new Student();
        temp.Input();
        listStudents.Add(temp);
    }

    Console.WriteLine("\n ====Xuất Danh Sách sinh viên====");
    foreach (Student sv in listStudents)
    {
        sv.Show();
    }
    Console.ReadKey(); //Dừng màn hình kiểm tra kết quả
}
```

2. Sử dụng **Cú pháp truy vấn (query syntax)** hoặc **cú pháp phương thức (method syntax)** trong **LINQ**. Danh sách truy vấn trong **LINQ**

Loại	Phương thức sử dụng
Lọc dữ liệu	Where
Chọn dữ liệu	Select, SelectMany
Phân vùng dữ liệu	Take, Skip, TakeWhile, SkipWhile
Kết hợp	Join, GroupJoin
Sắp Xếp	OrderBy / ThenBy, Reverse
Gom nhóm	GroupBy
Toán tử tập hợp	Distinct, Union, Intersect, Except
Chuyển đổi kiểu dữ liệu	ToSequence, ToArray, ToList, ToDictionary, ToLookup, OfType, Cast
Phần tử	First, FirstOrDefault, Last, LastOrDefault, Single, SingleOrDefault, ElementAt, ElementAtOrDefault, DefaultIfEmpty
Các hàm tính toán kết hợp	Count, LongCount, Sum, Min, Max, Average, Aggregate
Toán tử phát sinh	Range, Repeat, Empty
Toán tử lượng hóa	Any, All, Contains
Trộn	Concat

- Các yêu cầu 1.1 đến 1.5 đều xuất ra danh sách, nên chúng ta nên tách hàm nhập - xuất sinh viên riêng để tái sử dụng



```
private static List<Student> NhapDSSinhVien()
{
    List<Student> listStudents = new List<Student>();
    Console.WriteLine("Nhập tổng số sinh viên N =");
    int N = Convert.ToInt32(Console.ReadLine());

    Console.WriteLine("\n ====Nhập Danh Sách sinh viên====");
    for (int i = 0; i < N; i++)
    {
        Console.WriteLine("\n - Nhập sinh viên thứ {0}", i + 1);
        Student temp = new Student();
        temp.Input();
        listStudents.Add(temp);
    }
    return listStudents;
}

4 references
private static void XuatDSSinhVien(List<Student> listStudent)
{
    Console.WriteLine("\n ====Xuất Danh Sách sinh viên====");
    foreach (Student sv in listStudent)
    {
        sv.Show();
    }
}

static void Main(string[] args)
{
    Console.OutputEncoding = Encoding.Unicode;
    Console.InputEncoding = Encoding.Unicode; //Để sử dụng tiếng việt

    //Nhập Sinh Viên - xuất danh sách sinh viên
    List<Student> listStudent = NhapDSSinhVien();
    XuatDSSinhVien(listStudent);
    Console.ReadKey(); //Dừng màn hình kiểm tra kết quả
}
```

- Thực hiện các câu query theo yêu cầu bằng LINQ ( chỉ cần chọn 1 cách các bạn quen thuộc để thực hiện, trong các HD tiếp theo tôi sử dụng cú pháp phương thức)

```
//1.1 Xuất ra thông tin của các SV đều thuộc khoa "CNTT" (nếu có)
//cách 1: Sử dụng cú pháp truy vấn
List<Student> listStudentCNTT = (from s in listStudent
                                where s.Faculty == "CNTT"
                                select s).ToList();

//Cách 2: Sử dụng cú pháp phương thức
List<Student> listStudentCNTT = listStudent.Where(p => p.Faculty == "CNTT").ToList();

//xuất thông tin
if (listStudentCNTT.Count() == 0)
    Console.WriteLine("Không có sinh viên thuộc khoa CNTT");
else
    XuatDSSinhVien(listStudentCNTT);
```

- Thực hiện 1.2 Sử dụng Where Lấy danh sách sinh viên có điểm >=5



```
// 1.2 Xuất ra thông tin các sinh viên có điểm TB lớn hơn bằng 5(nếu có).
Console.WriteLine("1.2 Xuất ra thông tin các sinh viên có điểm TB lớn hơn bằng 5");
List<Student> listStudentResult = listStudent.Where(p => p.AverageScore >=5).ToList();
if (listStudentResult.Count() == 0)
    Console.WriteLine("Không có sinh viên có điểm >=5");
else
    XuatDSSinhVien(listStudentResult);
```

- Thực hiện 1.3 Sử dụng OrderBy để sắp tăng dần

```
// 1.3 Xuất ra danh sách các sinh viên được sắp xếp theo điểm trung bình tăng dần
Console.WriteLine("1.3 Sắp xếp sinh viên có điểm tăng dần");
List<Student> listStudentSort = listStudent.OrderBy(p => p.AverageScore).ToList();
XuatDSSinhVien(listStudentSort);
```

- Thực hiện 1.4 và 1.5 (SV tự thực hiện)

**Bài tập 2:** Tạo thêm 1 project ở trong cùng solution Lab01 có tên là “**Lab01-02**”.

- ✓ Viết lại chương trình từ bài tập 1 trên theo cách tạo thêm một lớp là **Person** làm lớp cơ sở cho lớp **Student**. Chọn Mã số, Họ tên làm field để đưa lên lớp **Person**.
- ✓ Thêm thông tin lớp giảng viên **Teacher** kế thừa từ lớp **Person**. Mỗi giảng viên đều có Mã số, Họ Tên và Địa chỉ.
- ✓ Ở hàm **Main()** cho phép nhập vào tổng số N đối tượng được đưa vào **List** danh sách **Person**, ở mỗi lần nhập liệu user có quyền chọn là nhập thông tin cho **Student** hoặc **Teacher**. Xuất danh sách thông tin vừa nhập liệu trên.
- ✓ Thực hiện một số yêu cầu sau trên List danh sách nhập liệu
  - 2.1 Tìm kiếm danh sách các Sinh Viên thuộc khoa “**CNTT**” nếu có.
  - 2.2 Xuất ra danh sách tất cả sinh viên có điểm trung bình nhỏ hơn 5 và thuộc khoa “**CNTT**”.
  - 2.3 Xuất ra danh sách các giáo viên có địa chỉ chứa thông tin “**Quận 9**” nếu có
  - 2.4 Tìm kiếm giáo viên có mã giảng viên là CHN060286. Xuất ra thông tin giáo viên tìm được nếu có.
  - 2.5 Tìm danh sách sinh viên có điểm trung bình cao nhất và thuộc khoa “**CNTT**”

**Bài tập 3:** Thêm 1 project ở trong cùng solution Lab01 có tên là “**Lab01-03**”.

Công ty địa ốc D cần xây dựng chương trình quản lý thông tin về các khu đất do công ty cung ứng với các thông tin cần quản lý

- Khu Đất: Địa Điểm, Giá Bán (Đơn vị tính: VND) và Diện Tích (m2)

Thực hiện các yêu cầu sau:

- ✓ Xây dựng các lớp với chức năng thừa kế
- ✓ Nhập xuất danh sách các Khu Đất của công ty.

- ✓ Xuất ra danh sách thông tin các khu đất có diện tích được sắp xếp tăng dần.
- ✓ Xuất ra danh sách các khu đất có giá bán  $< 1$  tỷ và diện tích  $\geq 60\text{m}^2$  (nếu có).
- ✓ Tính đơn giá trung bình  $1\text{m}^2$  của tất cả các khu đất có diện tích lớn hơn  $1000\text{m}^2$  (nếu có).

**Bài tập 4:**

Công ty địa ốc D muốn mở rộng kinh doanh thêm về thị trường Nhà phố và Chung Cư. Với tất cả các thông tin cần quản lý như sau:

- Khu Đất: Địa Điểm, Giá Bán (Đơn vị tính: VND) và Diện Tích ( $\text{m}^2$ )
- Nhà Phố: Địa Điểm, Giá Bán (Đơn vị tính: VND), Diện tích ( $\text{m}^2$ ), Năm Xây dựng, Số tầng
- Chung Cư: Địa Điểm, Giá Bán (Đơn Vị Tính: VND), Diện Tích ( $\text{m}^2$ ), Tầng

Thực hiện các yêu cầu sau:

- ✓ Xây dựng các lớp với chức năng thừa kế
- ✓ Nhập xuất danh sách các thông tin (Khu đất, Nhà phố, Chung Cư) cần quản lý.
- ✓ Xuất tổng giá bán cho 3 loại (Khu đất, Nhà phố, Chung Cư) của công ty D.
- ✓ Xuất danh sách các khu đất có diện tích  $> 100\text{m}^2$  hoặc là nhà phố mà có diện tích  $> 60\text{m}^2$  và năm xây dựng  $\geq 2020$  (nếu có).
- ✓ Nhập vào các thông tin cần tìm kiếm (địa điểm, giá, diện tích). Xuất thông tin danh sách tất cả các nhà phố hoặc chung cư phù hợp yêu cầu. (có địa điểm chứa chuỗi tìm kiếm không phân biệt hoa thường, có giá  $\leq$  giá tìm kiếm, và diện tích  $\geq$  diện tích cần tìm kiếm)

-----**Hết Lab 01**-----