Lab 01:

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG CƠ BẢN VỚI C#

A. MỤC TIÊU:

- ✓ Hướng dẫn sinh viên làm quen với ngôn ngữ lập trình C#: qua việc viết các ứng dụng **console** đơn giản
- ✓ Xây dựng các lớp, tạo đối tượng, truy xuất các phương thức, …
- ✓ Làm quen với môi trường lập trình VS .NET 2010/2012/2013
- ✓ Soạn thảo mã nguồn, biên dịch, debug, thực thi chương trình…
- ✓ Nâng cao: Sinh viên tự nghiên cứu về kế thừa trong lập trình hướng đối tượng trên C#.

B. NỘI DUNG:

Bài tập 1: Tạo lớp Student có các dữ liệu và phương thức sau:

- ✓ SID (mã số sinh viên)
- ✓ Tên sinh viên
- ✓ Khoa
- ✓ Điểm TB
- ✓ Thêm các property cho các dữ liệu thành viên trên
- ✓ Viết các phương thức hiển thị thông tin của sinh viên.

Tạo lớp Tester, trong lớp này chỉ chứa duy nhất hàm **main()**. Hàm cho phép người dùng nhập vào số **n** là số sinh viên, sau đó lần lượt tạo các đối tượng sinh viên và add vào danh sách sinh viên theo những thông tin do user nhập vào (dùng vòng lặp for). Cuối cùng xuất ra danh sách chi tiết thông tin sinh viên.

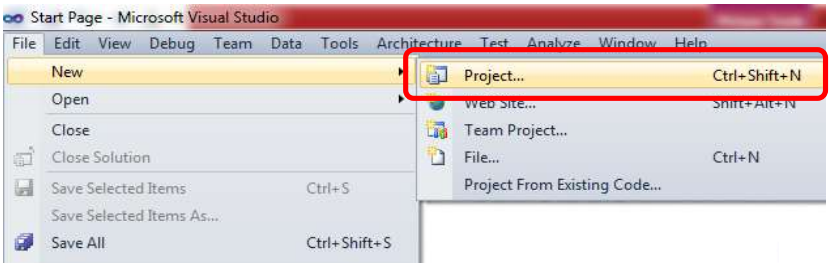
Yêu cầu:

- ✓ Sinh viên xây dựng chương trình theo nội dung mô tả bên trên.
- ✓ 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 trong VS .NET 2010:

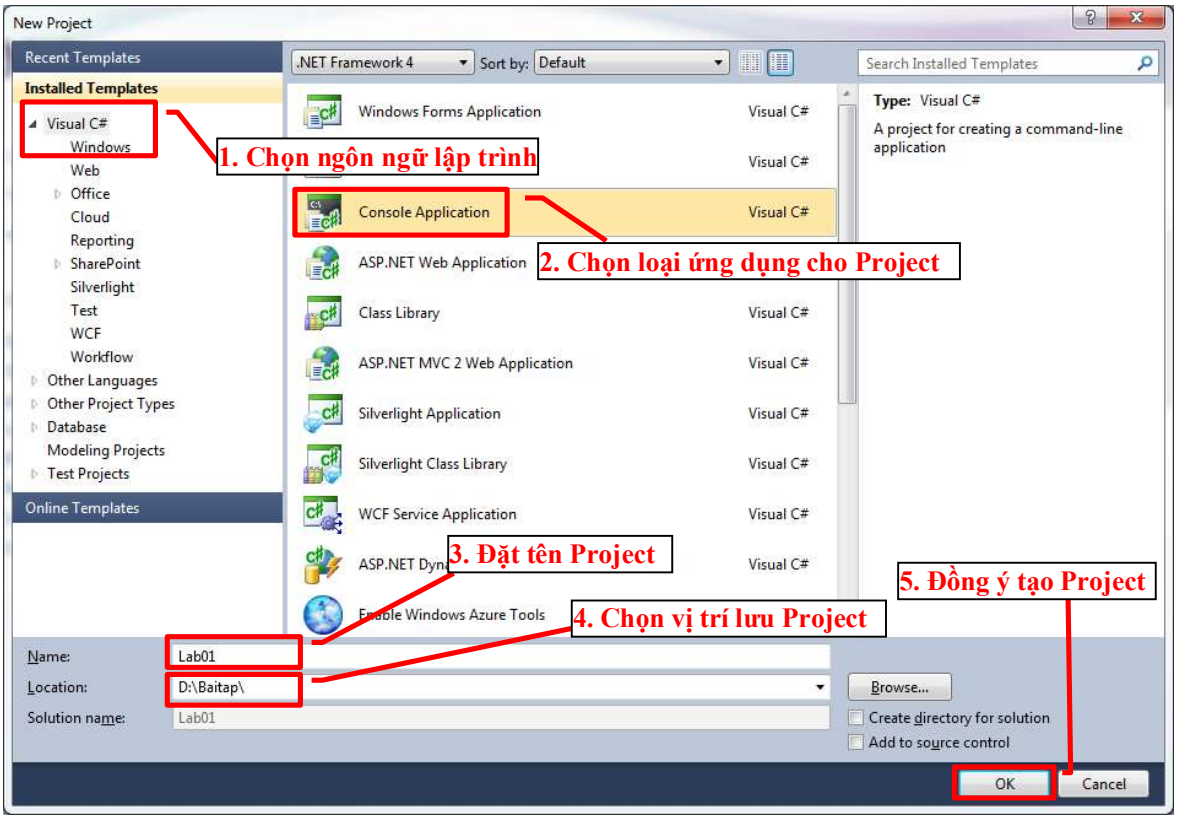
- ✓ Trong menu File chọn New → Poject hoặc nhấn tổ hợp phím (Ctrl+Shift+N), xuất hiện cửa sổ New Project.



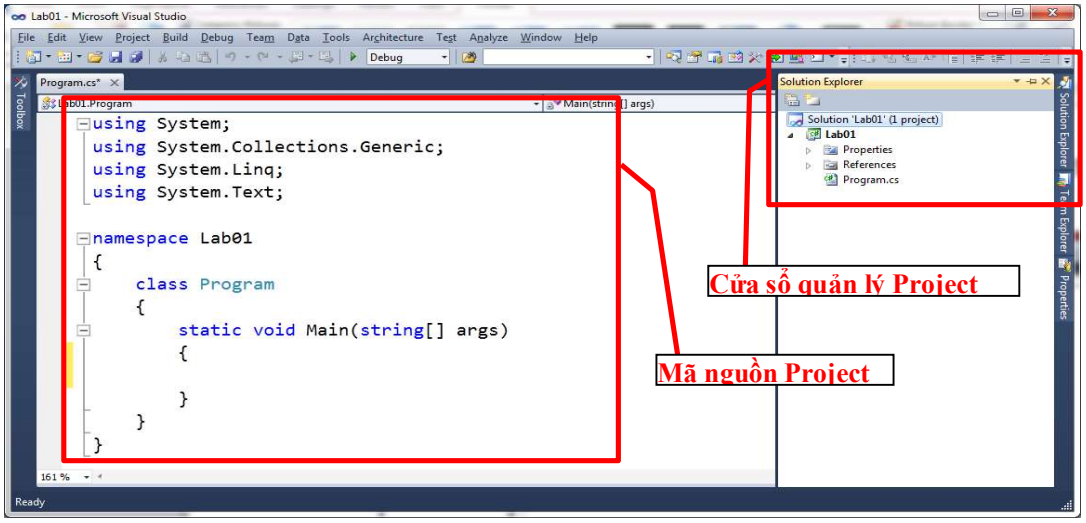
Hình 1: Màn hình tạo Project

✓ Trong cửa sổ **New Project**: chọn

- Project type là Visual C# - Windows
- Chọn templates là Console Application
- Nhập tên project vào phần Name: **Lab01**
- Khai báo đường dẫn lưu trữ trong Location…
- Khai báo tên Project…



⇨ Click O**K** để đồng ý tạo project, kết quả chúng ta được một ứng dụng console như sau:



Hình 3: Màn hình làm việc của Project

Bước 2: Xây dựng các lớp theo yêu cầu

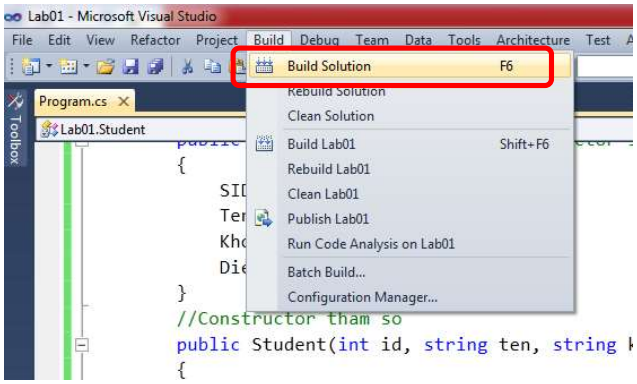
- ✓ Xóa lớp Program mặc định tạo lớp Student gồm các thuộc tính, Cosntructor, các phương thức như sau:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace Lab01
{
    class Student
    {
        private int SID;
        private string TenSV;
        private string Khoa;
        private float  DiemTB;
        public Student() //Constructor mac dinh
        {
            SID = 1;
            TenSV = "Nguyen Van Nam";
            Khoa = "CNTT";
            DiemTB = 7;
        }
        public Student(Student stu) //Constructor sao chep
        {
            SID = stu.SID;
            TenSV = stu.TenSV;
            Khoa = stu.Khoa;
            DiemTB =stu.DiemTB;
        }
        //Constructor tham so
        public Student(int id, string ten, string kh, float dtb)
        {
            SID = id;
            TenSV = ten;
            Khoa = kh;
            DiemTB = dtb;
        }
        //Cac Property cho tung thuoc tinh cua lop
        public int StudentID //Property dai dien cho thuoc tinh SID
        {
            get { return SID; } //Lay du lieu
            set { SID = value;}  //Gan du lieu
        }
        public String Name
        {
            get{ return TenSV; }
            set{ TenSV = value; }
        }
        public String Faculty
        {
            get { return Khoa; }
            set { Khoa = value; }
        }
        public float Mark
        {
            get { return DiemTB; }
            set { DiemTB = value; }
        }
        //Phuong thuc hien thi du lieu
        public void Show()
        {
            Console.WriteLine("MSSV:{0}", this.SID);
            Console.WriteLine("Ten SV:{0}", this.TenSV);
            Console.WriteLine("Khoa:{0}", this.Khoa);
            Console.WriteLine("Diem TB:{0}", this.DiemTB);
```

```
        }
    }
    class Tester
    {
        public static void Main()
        {
            Student[] DSSV;
            int n;
            Console.Write("Nhap so luong SV:");
            n = int.Parse(Console.ReadLine());
            DSSV = new Student[n]; //tao mang n phan tu
            Console.WriteLine("\n ====NHAP DS SINH VIEN====");
            for (int i = 0; i < n; i++) //Lap n lan nhap thong tin sv
            {
                DSSV[i] = new Student();
                Console.Write("Nhap MaSV {0}:",i+1);
                DSSV[i].StudentID =int.Parse(Console.ReadLine());
                Console.Write("Ho ten SV:");
                DSSV[i].Name = Console.ReadLine();
                Console.Write("Nhap khoa:");
                DSSV[i].Faculty = Console.ReadLine();
                Console.Write("Nhap Diem TB:");
                DSSV[i].Mark = float.Parse(Console.ReadLine());
            }
            //Xuat DS Sinh vien
            Console.WriteLine("\n ====XUAT DS SINH VIEN====");
            foreach (Student sv in DSSV)
                sv.Show();
            Console.ReadLine();
        }
    }
}
```

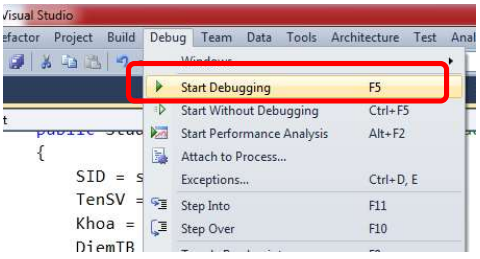
Bước 3: Biên dịch và chạy chương trình:

- ✓ Chức năng Build Solution: F6



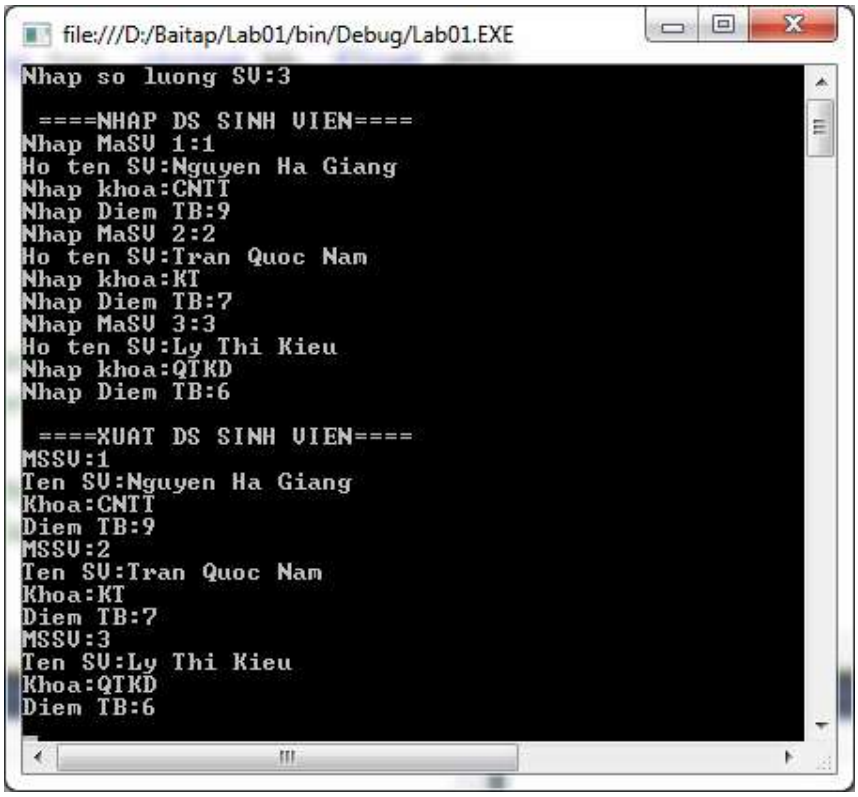
Hình 4: Chức năng Build

- ✓ Chức năng Run with Debug: F5



Hình 5: Chức năng Debug/ without Debug

- ✓ Chức năng Run without Debug: Ctrl + F5



Hình 6: Màn hình chương trình khi thực hiện

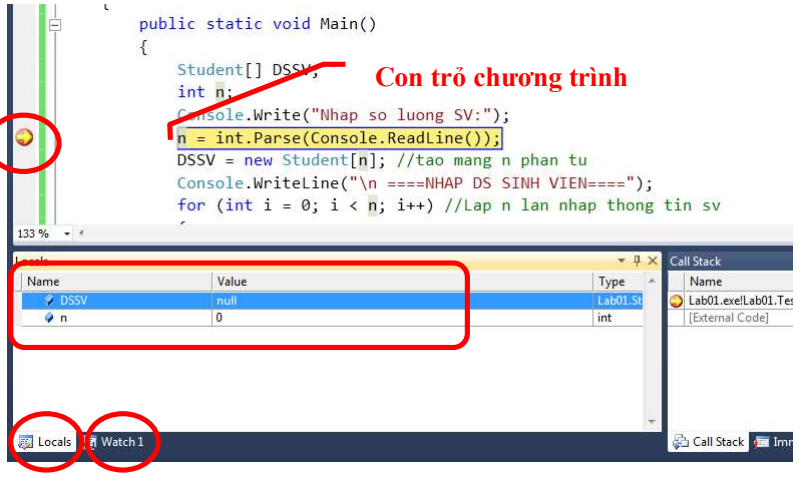
✓ Sinh viên chạy debug chương trình: sử dụng các chức năng breakpoint, xem giá trị của các biến trong cửa sổ Locals hoặc nhập các biến vào cửa sổ Watch1 để xem giá trị hiện tại của nó.

Cách thức chạy từng bước chương trình trong chế độ Debug:

- Cách thứ nhất chèn cách breakpoint vào một dòng lệnh nào đó: trong màn hình soạn thảo, di chuyển con trỏ văn bản tới dòng cần dừng nhấn <**F9**> hay kích chuột vào lề trái của dòng đó sẽ xuất hiện ký hiệu breakpoint

  Khi chạy debug thì chọn <**F5**> chương trình sẽ thực hiện và dừng tại breakpoint, muốn chạy tiếp thì tiếp tục nhấn <**F5**>

  Để remove breakpoint thì di chuyển con trỏ văn bản vào dòng đó và nhấn <**F9**>



Hình 7: Màn hình chương trình chạy debug dừng tại một breakpoint.

- Cách thứ hai chạy từng dòng lệnh, bắt đầu từ hàm Main():

Nhấn <**F10**>: chương trình sẽ chạy debug vào lệnh đầu tiên của hàm Main

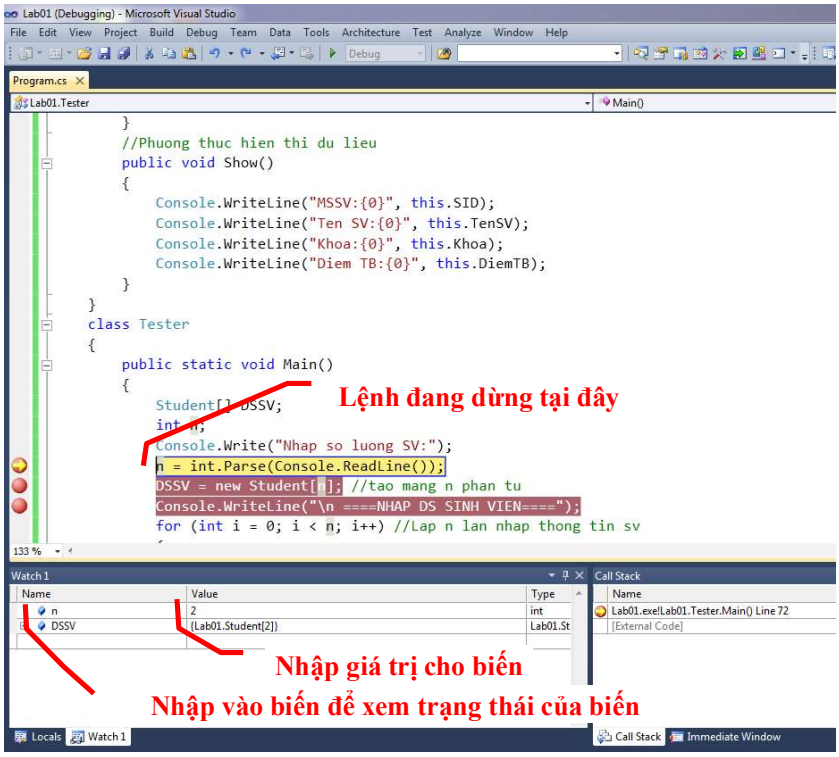
Chạy từng dòng lệnh thì nhấn <**F10**>

Vào trong thân của một lời gọi hàm <**F11**>

Thoát ra khỏi hàm nào đó thì nhấn <**Shift + F11**>

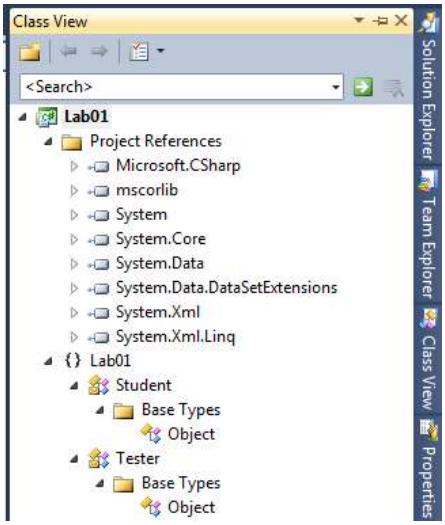
- Để kết thúc debug ở bất cứ nơi nào thì nhấn: <**Shift + F5**>

Trong quá trình debug có thể kết hợp với của sổ Locals hay Watch1 để xem thông tin chi tiết của các biến, đối tượng trong chương trình.



Hình 8: Màn hình trạng thái debug chương trình

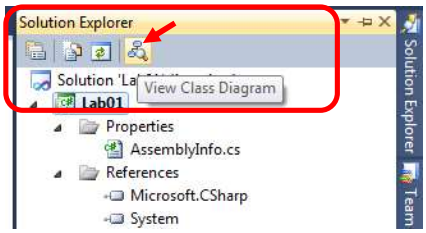
✓ Sử dụng cửa sổ **Class View** (Ctrl + W, C) để quản lý các lớp có trong chương trình



Hình 9: Cửa sổ Class View của project

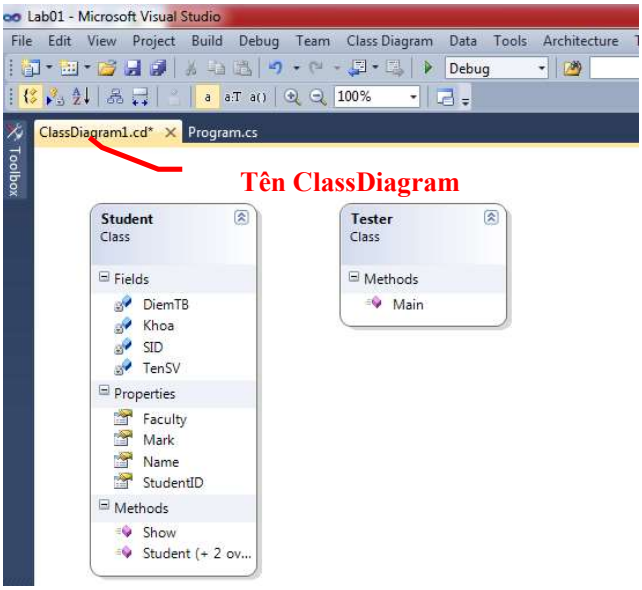
✓ Sử dụng chức năng **View Class Diagram** để xem các lớp có trong chương trình:

- Trong cửa sổ Solution Explorer chọn button **View Class Diagram**



Hình 10: Chọn chức năng View Class Diagram

- Sau khi chọn chức năng này, VS.NET 2010 sẽ khởi tạo ra class diagram và lưu vào một file có phần mở rộng là ***.cd**



Hình 11: Class diagram do VS.NET phát sinh theo source code chương trình

Lưu ý: Sinh viên có thể bổ sung các thuộc tính như địa chỉ, số điện thoại, hoặc chi tiết các cột điểm…

Bài tập 2:

- ✓ Trong lớp Student trên không dùng Property cho các dữ liệu mà thay vào dùng phương thức getter và setter.
- ✓ Không viết lệnh nhập xuất danh sách sinh viên trực tiếp trong hàm Main() mà hãy viết hàm nhập 1 sinh viên Nhap1SV(), hàm nhập danh sách sinh viên NhapDS() và hàm xuất danh sách sinh viên XuatDS(), gọi hàm Show() đã viết. Sau đó trong hàm Main() gọi các hàm NhapDS(), XuatDS().
- ✓ Viết lại chương trình theo yêu cầu trên.

Bài tập 3:

- ✓ Viết lại chương trình trên theo cách tạo thêm một lớp là People làm lớp cơ sở cho lớp Student. Sinh viên chọn những field thích hợp để đưa lên lớp cơ sở People…
- ✓ Viết lại phần hàm Main trong lớp Tester theo yêu cầu sau:
  - Sử dụng lớp collection là **List** để chứa danh sách sinh viên thay thế cho mảng sinh viên trong bài tập trên

- Sử dụng lớp collection là **ArrayList** để chứa danh sách sinh viên thay vì mảng.

✓ Tìm hiểu chức năng hỗ trợ **Refactor** trong VS.NET, sử dụng chức năng **Encapsulate Fields** để tạo các property trong khi viết chương trình C#.

Hướng dẫn: Sử dụng chức năng Refactor: Encapsulate Fields bằng cách khi tạo các field của lớp: click phải chuột vào tên của field cần tạo property, menu ngữ cảnh xuất hiện, chọn chức năng Refactor và chọn mục Encapsulate Fields. Một dialog Encapsulate Field xuất hiện cho phép mô tả property. Bước cuối cùng chọn OK, VS.NET sẽ phát sinh code tạo property.

Bài tập 4:

Thiết kế chương trình quản lý các đối tượng sau trong một Viện khoa học: nhà khoa học, nhà quản lý và nhân viên phòng thí nghiệm. Một nhà khoa học cũng có thể làm công tác quản lý. Các thành phần dữ liệu của các đối tượng trên:

- ✓ Nhà khoa học: họ tên, năm sinh, bằng cấp, chức vụ, số bài báo đã công bố, số ngày công trong tháng, bậc lương
- ✓ Nhà quản lý: họ tên, năm sinh, bằng cấp, chức vụ, số ngày công trong tháng, bậc lương
- ✓ Nhân viên phòng thí nghiệm: họ tên, năm sinh, bằng cấp, lương trong tháng.

Biết rằng nhân viên phòng thí nghiệm lãnh lương khoán, còn lương của nhà khoa học và nhà quản lý bằng số ngày công trong tháng * bậc lương. Nhập, xuất danh sách nhân viên và in tổng lương đã chi trả cho từng loại đối tượng.

-- Hết Lab 01 --