Bài 3. Lập trình hướng đối tượng

4 tính chất của LTHĐT:

- ✓ ***Tính đóng gói***
- ✓ ***Tính kế thừa***
- ✓ ***Tính đa hình***
- ✓ ***Tính trừu tượng***

1. Lớp trong lập trình hướng đối tượng (class)

1.1. Khái niệm

- Lập trình HĐT là phương pháp thiết kế và phát triển phần mềm dựa trên kiến trúc lớp (class) và đối tượng (object).
- Lớp là một kiểu dữ liệu mới.
- Đối tượng là một biến có kiểu thuộc một lớp đã định nghĩa.

Ví dụ: Lớp XeOto, Lớp DienThoai, Lớp SinhVien,…

1.2. Khai báo lớp

Cú pháp:

[phạm vi] class <tên lớp>

{<các thành phần của lớp>}

Trong đó: **[phạm vi]** có thể là:

- private (mặc định): không cho phép truy cập từ lớp khác vào lớp hiện tại.
- public: cho phép truy cập từ các lớp khác vào lớp hiện tại.
- protected: cho phép truy cập tới lớp kế thừa (lớp con) từ lớp hiện tại.

1.3. Tạo một đối tượng (thể hiện) kiểu lớp

Cú pháp:

<tên lớp> <tên đối tượng>=new <tên lớp>([DS tham số]);

Ví dụ: Tạo đối tượng **NV** thuộc lớp **NhanVien**

```
NhanVien NV=new NhanVien();
```

1.4. Truy cập vào một thành phần của lớp

Cú pháp:

<tên đối tượng>.<tên thành phần lớp>;

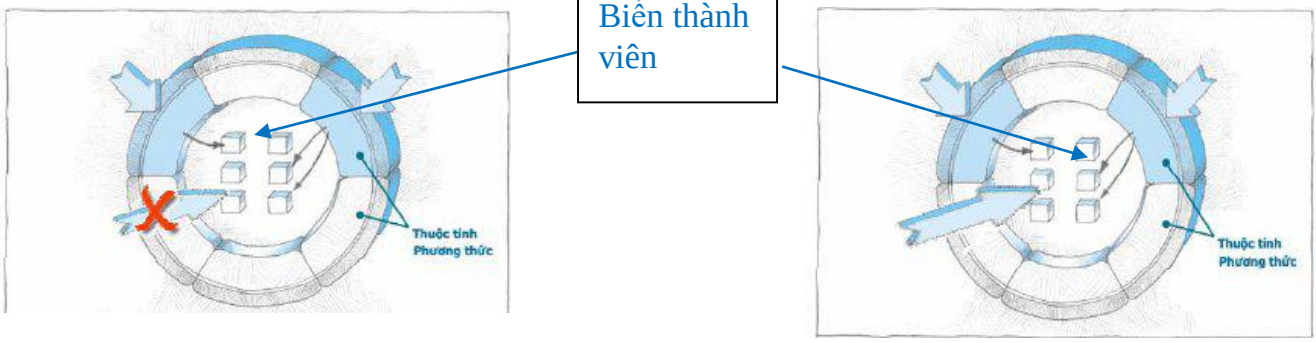
Ví dụ:

```
NV.Nhap();  //Nhap() là 1 phương thức
```

2. Các thành phần của lớp

Có 3 thành phần: biến thành viên, thuộc tính và phương thức.

Đối tượng



Khai báo **private** cho biến thành viên
(không cho phép truy cập từ ngoài vào)

Khai báo **public** cho biến thành viên
(cho phép truy cập từ ngoài vào)

2.1. Biến thành viên (Field)

Dùng để lưu trữ dữ liệu của một lớp.

2.2. Thuộc tính (Properties)

a. Định nghĩa:

- Là thành phần được sử dụng để truy xuất đến các biến thành viên (Field) được khai báo **private** trong **class**.
- Mỗi thuộc tính chỉ truy xuất đến một biến thành viên duy nhất.
- Dùng **properties** để:
  - ✓ Che dấu biến thành viên (field), không cho phép người dùng truy cập trực tiếp vào field.***(T/c đóng gói)***
  - ✓ Tạo thuộc tính chỉ đọc (get) hoặc chỉ ghi (set).
  - ✓ Đảm bảo tính toàn vẹn dữ liệu thông qua kiểm tra lỗi.

b. Khai báo:

- Lấy và thiết lập các giá trị từ thuộc tính:

```
public  <kiểu> <tên thuộc tính>
 {
   get {return <tên biến>;}   //lấy giá trị
   set {<tên biến>=value;}   //gán giá trị
 }
```

- Trong một số trường hợp, người dùng chỉ dùng thuộc tính để lấy và thiết lập giá trị mà không có phép tính logic nào thì có thể sử dụng các thuộc tính được triển khai tự động.

```
public  <kiểu> <tên thuộc tính> { get; set;}
```

Ví dụ 1: khai báo thuộc tính **HSLuong** truy xuất biến thành viên **_HSLuong**

```
double _HSLuong;
public double HSLuong
   {
     get {return _HSLuong;}
     set {_HSLuong=value;}
   }
```

Có thể khai báo ngắn gọn như sau:

```
public double HSLuong {get; set;} //Tự sinh biến thành viên ngầm định
```

Ví dụ 2: khai báo thuộc tính **HSLuong** với giá trị phải >=1 (kiểm tra tính hợp lệ của dữ liệu)

```
public double HSLuong
 {
   get{return _HSLuong;}
   set{
        if (value>=1)
          _HSLuong=value;
      }
 }
```

2.3. Phương thức (Method):

Là một chương trình con dùng để thực hiện một công việc gì đó.

a. Phương thức khởi tạo (Constructor)

- **Đặc điểm:**
  - ✓ Là phương thức đầu tiên được triệu gọi và chỉ gọi một lần khi khởi tạo đối tượng bằng lệnh **new.**
  - ✓ Nó nhằm khởi tạo các giá trị ban đầu cho biến thành viên.
  - ✓ Tên của nó trùng với tên lớp.
  - ✓ **Constructor** không tham số sẽ được tự động sinh ra khi không định nghĩa, khi đó các biến thành viên sẽ được gán giá trị mặc định.
- **Khai báo Constructor**

[private/public] <tên lớp> ([các tham số])

{

// khởi tạo giá trị cho các biến thành viên

}

Ví dụ:

```
class NhanVien
   {
     //khai báo 2 thuộc tính
        public int maso {get; set;}
        public string hoten {get; set;}
```

```
//phương thức khởi tạo không TS
   public NhanVien()
     {  maso=0; hoten=“”; }
// phương thức khởi tạo có TS
  public NhanVien(int ms, string ht)
   {
     maso=ms; hoten=ht;
   }
```

2 phương thức cùng tên nhưng khác tham số
➔ Nạp chồng phương thức (Overloading)

+ Gọi phương thức khởi tạo:

```
- NhanVien P1 =new NhanVien(); //gọi phương thức khởi tạo không TS

- NhanVien P2 =new NhanVien(10,”Lê Thị Hà”); //gọi phương thức khởi tạo có TS
```

b. phương thức xử lý, tính toán

- **Phương thức trả về giá trị**

[private/public] <kiểu> <Tên PT>([DS tham số])

{

<Khối lệnh>;

return(giá trị);

}

+ Gọi phương thức:

<biến>=<Tên đối tượng>.<Tên phương thức>([DS đối số]);

Ví dụ:

```
public double TinhLuong()
   {
      double kq=HSLuong*850000;
      return kq;
  }
```

- **Phương thức kiểu void (*không trả về giá trị*)**

[private/public] void <Tên PT>([DS tham số])

{

<Khối lệnh>;

}

+ Gọi phương thức:

<Tên đối tượng>.<Tên phương thức>([DS đối số]);

Ví dụ:

```
public void  Chao()
  {
      Console.WriteLine(“Chào các bạn!”);
  }
```

Bài tập

Bài 1: Xây dựng lớp GiaiPTB1 có các thuộc tính:

- double a
- double b

- Viết các phương thức khởi tạo và các phương thức khác phù hợp để có thể giải phương trình bậc 1 với 2 tham số tương ứng với 2 thuộc tính.
- Viết mã trong phương thức Main ở lớp Program để gọi các phương thức trên.

Code demo:

- Thêm class **GiaiPTB1**: Project → Add New Item → Chọn template là class → nhập tên class là **GiaiPTB1.cs** → Add

```
class GiaiPTB1
 {
  //Thuộc tính hệ số a, b
     public double a{ get; set; }
     public double b{ get; set; }
  //Constructor không có TS
     public GiaiPTB1()
         { a = 0; b = 0; }
        //Constructor có TS
     public GiaiPTB1(double a1, double b1)
         { a = a1; b = b1; }
     //Phương thức nhập hệ số
     public void Nhapheso()
     {
        Console.Write("He so a=");
        a = double.Parse(Console.ReadLine());
        Console.Write("He so b=");
        b = double.Parse(Console.ReadLine());
     }
     //Phương thức giải phương trình
     public string GiaiPT()
     {
        string kq;
        if (a != 0)
           kq = "X=" + (-b / a);
        else if (b == 0)
           kq = "PT vo so nghiep";
        else
           kq = "PTVN";
        return (kq);
     }
 }
```

```
class Program
  {
    static void Main(string[] args)
    {
      //Sử dụng constructor không có TS
      GiaiPTB1 pt1 = new GiaiPTB1();
      pt1.Nhapheso();
      Console.Write(pt1.GiaiPT());
      Console.WriteLine();
      //Sử dụng constructor có tham số
      GiaiPTB1 pt2 = new GiaiPTB1(1, 2);
      Console.Write(pt2.GiaiPT());
      Console.WriteLine();
      Console.Read();
    }
  }
```

Bài 2: Sửa lại Bài 1 trên với 2 phương thức **Nhapheso()** và **GiaiPT()** có tham số.

Bài 3: Tạo lớp **Student** gồm các biến thành viên: SID (mã số sinh viên), Tên sinh viên, Khoa, Điểm TB.

- Viết các thuộc tính để truy cập an toàn các biến thành viên này.
- Viết các phương thức khởi tạo không tham số, có tham số.
- Viết các phương thức nhập, xuất thông tin của một sinh viên.
- Viết mã trong phương thức **Main** ở lớp **Program** để gọi đủ các phương thức đã viết.
- Viết mã trong phương thức **Main** ở lớp **Program** để nhập vào một danh sách n sinh viên. In danh sách các sinh viên lên màn hình.

```
class Student
  {       //Thuộc tính
      public int SID {get;set;}
      public String TenSV {get;set;}
      public String Khoa {get;set;}
      public float DiemTB {get;set;}

    public Student()     //Constructor khong tham so
        {
          SID = 1;
          TenSV = "Nguyen Van Nam";
          Khoa = "CNTT";
          DiemTB = 7;
        }
```

```
    //Constructor co tham so
    public Student(int id, string ten, string kh, float dtb)
        {
          SID = id;
          TenSV = ten;
          Khoa = kh;
          DiemTB = dtb;
        }
     //Phuong thuc nhap du lieu
      public void Input()
      {
          Console.WriteLine("Vao ma SV:");
          SID = Int32.Parse(Console.ReadLine());
          Console.WriteLine("Vao ho ten SV:");
          TenSV = Console.ReadLine();
          Console.WriteLine("Vao khoa:");
          Khoa = Console.ReadLine();
          Console.WriteLine("Vao diem TB:");
          DiemTB = float.Parse(Console.ReadLine());
       }
    //Phuong thuc hien thi du lieu
      public void Show()
      {
          Console.WriteLine("MSSV:{0}", SID);
          Console.WriteLine("Ten SV:{0}", TenSV);
          Console.WriteLine("Khoa:{0}", Khoa);
          Console.WriteLine("Diem TB:{0}", DiemTB);
      }
}
```

```
class Program
  {
      static void Main(string[] args)
      {
        //Sử dụng constructor không có TS
          Student SV1 = new Student();
          SV.Input();
          SV.Show();
       /*  Sử dụng constructor có TS
          Student SV1 = new Student(2,"Le Thu Ha","Khoa CNTT",8);
          SV1.Show();  */

          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++)
       {
           DSSV[i] = new Student();
           Console.Write("Nhap MaSV {0}:", i + 1);
           DSSV[i].SID = int.Parse(Console.ReadLine());
           Console.Write("Ho ten SV:");
           DSSV[i].TenSV = Console.ReadLine();
           Console.Write("Nhap khoa:");
           DSSV[i].Khoa = Console.ReadLine();
           Console.Write("Nhap Diem TB:");
           DSSV[i].DiemTB = float.Parse(Console.ReadLine());
        }
       //Xuat DS Sinh vien
       Console.WriteLine("\n ====XUAT DS SINH VIEN====");
       foreach (Student sv in DSSV)
         sv.Show();
       Console.ReadLine();
    }
}
```

Bài 4: Sửa lại bài 3:

a- Không viết lệnh nhập xuất danh sách sinh viên trực tiếp trong hàm Main() mà hãy viết hàm nhập danh sách sinh viên **NhapDS()** và hàm xuất danh sách sinh viên **XuatDS()**. Sau đó trong hàm Main() gọi các hàm NhapDS(), XuatDS().

b- Sử dụng lớp collection là **List** để chứa danh sách sinh viên thay thế cho mảng sinh viên trong bài 3 trên.

Bài 5: Xây dựng lớp HOCSINH bao gồm các biến thành viên: Hoten(string), Tuoi(int), DiemT(int) , DiemL(int) , DiemH(int).

- Viết các thuộc tính để truy cập an toàn các biến thành viên này.
- Viết các phương thức khởi tạo không tham số, có tham số.
- Viết các phương thức nhập, xuất, xếp loại học sinh theo điểm trung bình của 1 học sinh.
- Viết các phương thức nhập, xuất một danh sách n học sinh.
- Viết mã trong phương thức Main ở lớp Program để gọi đủ các phương thức đã viết.