LAB 8. ĐA HÌNH BẰNG CÁCH THỰC THI GIAO DIỆN

THỜI LƯỢNG : 4 TIẾT

A. Mục tiêu

- Giúp sinh viên biết cách định nghĩa interface và tạo các lớp thực thi các interface đó.
- Cung cấp cho sinh viên một giải pháp khác để hiện thực tính đa hình.
- Sau khi hoàn thành bài thực hành này, sinh viên cần:
  - Biết cách tạo ra các interface
  - Hiểu rõ bản chất và cơ chế hoạt động của interface
  - Nắm vững cách thực thi một interface
  - Biết cách mở rộng một interface có sẵn để bổ sung thêm chức năng
  - Hiểu rõ bản chất của việc "đa kết thừa" bằng cách thực thi nhiều giao diện
  - Hiểu được tác dụng và sử dụng được một số interface có sẵn
  - Nhận biết được sự giống nhau và phân biệt được sự khác nhau giữa lớp trừu tượng (abstract class) và giao diện (interface)

B. Yêu cầu

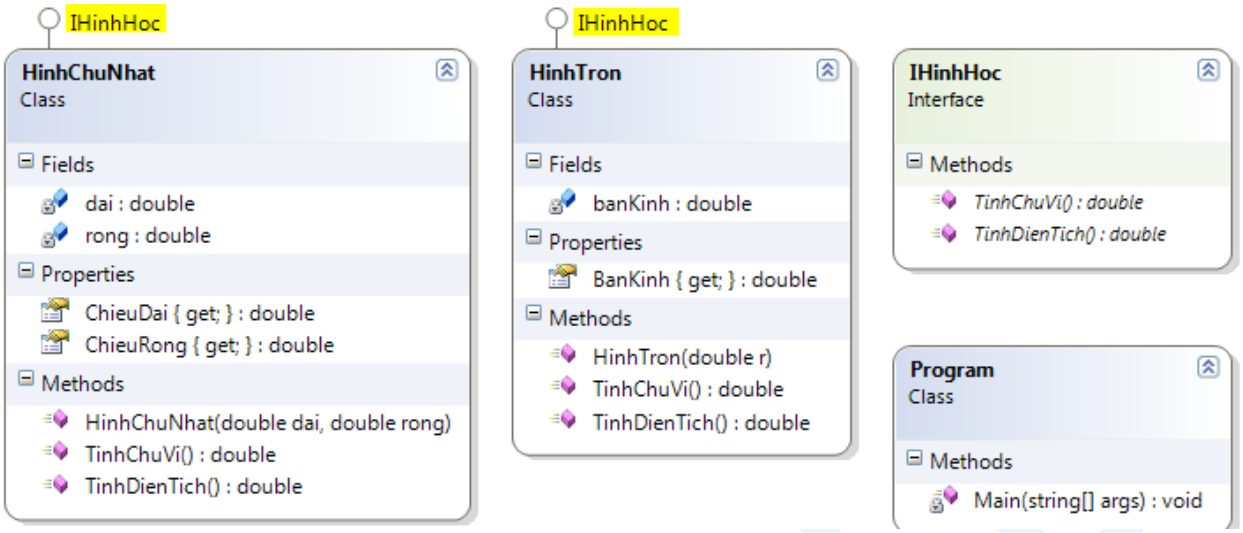
- Sinh viên phải trả lời đầy đủ các câu hỏi (nếu có) hoặc chụp màn hình kết quả, ghi lại vào tập tin Word theo yêu cầu ghi trong phần hướng dẫn thực hành.
- Đặt tên tập tin Word theo dạng: Lab08_MSSV_HoVaTen.rar.
- Tạo thư mục, đặt tên là MSSV_Lab08 để lưu tất cả bài làm trong bài thực hành này.
- Sinh viên phải hoàn thành tối thiểu 1 bài tập bắt buộc.
- Phải tạo một dự án riêng cho mỗi phần hoặc mỗi bài tập.
- Cuối buổi thực hành, nén thư mục trên & nộp bài qua email: phucnguyen5555@gmail.com

C. Hướng dẫn thực hành

1. Tạo một giao diện (interface) đơn giản

Phần này hướng dẫn cách tạo một giao diện đơn giản có tên là IHinhHoc. Interface này có hai phương thức (TinhChuVi,TinhDienTich) cần thiết đối với các đối tượng hình học 2D như hình dưới

Bước 1. Tạo một thư mục, đặt tên theo dạng MSSV-Lab08. Ví dụ: 1210234-Lab08.
Bước 2. Tạo dự án Console Application, đặt tên là Lab08_Bai01_GiaoDienDonGian.



Bước 3. Nhấp phải chuột vào tên project, chọn Add > New Item …

Bước 4. Trong cửa sổ Add new item, chọn Interface. Đặt tên cho giao diện là IHinhHoc.

Bước 5. Nhấn nút OK. Interface vừa tạo sẽ được hiển thị trong cửa sổ soạn thảo như sau

```
interface IHinhHoc
{
}
```

Bước 6. Cài đặt lại interface IHinhHoc như sau:

```
// Định nghĩa một giao diện chung cho các đối
// tượng hình học ( trong không gian 2 chiều)
public interface IHinhHoc
{
    // Khai báo nguyên mẫu hàm tính chu vi của hình
    double TinhChuVi();

    // Khai báo nguyên mẫu hàm tính diện tích
    double TinhDienTich();
}
```

2. Tạo các lớp thực thi giao diện

Phần này hướng dẫn cách tạo các lớp hình học và thực thi giao diện IHinhHoc vừa tạo.

Bước 7. Tạo một lớp mới, đặt tên là HinhTron. Cài đặt lớp này như sau:

```
// Lớp biểu diễn hình tròn có bán kính R.
// Nó thực thi giao diện IHinhHoc
public class HinhTron : IHinhHoc
{
    // Khai báo biến thành viên
```

```
    private double banKinh;

    // Định nghĩa thuộc tính
    public double BanKinh
    {
        get { return banKinh; }
    }

    // Định nghĩa phương thức khởi tạo
    public HinhTron(double r)
    {
        banKinh = r;
    }
}
```

Bước 8. Chọn menu Build > Build Solution để biên dịch chương trình. Hãy cho biết có lỗi xảy ra hay không? Nếu có, giải thích các lỗi đó. (Vào menu View > Error List để xem các lỗi)

Bước 9. Trong lớp HinhTron, bổ sung thêm hai phương thức sau:

```
    // Định nghĩa hàm tính chu vi. Lưu ý là
    // phải theo đúng nguyên mẫu trong giao diện
    public double TinhChuVi()
    {
        return 2 * banKinh * Math.PI;
    }

    // Định nghĩa hàm tính diện tích. Lưu ý là
    // phải theo đúng nguyên mẫu trong giao diện
    public double TinhDienTich()
    {
        return banKinh * banKinh * Math.PI;
    }
```

Bước 10. Biên dịch lại chương trình và kiểm tra xem còn lỗi hay không?

Bước 11. Tiếp tục tạo lớp HinhChuNhat và định nghĩa lớp này như sau:

```
// Lớp này biểu diễn hình chữ nhật.
// Nó thực thi giao diện IHinhHoc
public class HinhChuNhat : IHinhHoc
{
    // Khai báo biến thành viên
    private double dai;
    private double rong;

    // Định nghĩa thuộc tính
    public double ChieuDai
    {
        get { return dai; }
    }

    public double ChieuRong
```

```
        {
            get { return rong; }
        }

        // Định nghĩa phương thức khởi tạo
        public HinhChuNhat(double dai, double rong)
        {
            this.dai = dai;
            this.rong = rong;
        }

        // Định nghĩa hàm tính chu vi. Lưu ý là
        // phải theo đúng nguyên mẫu trong giao diện
        public double TinhChuVi()
        {
            return 2 * (dai + rong);
        }

        // Định nghĩa hàm tính diện tích. Lưu ý là
        // phải theo đúng nguyên mẫu trong giao diện
        public double TinhDienTich()
        {
            return dai * rong;
        }
    }
```

Bước 12. Biên dịch lại chương trình và kiểm tra xem có lỗi hay không? Nếu có, hãy sửa lỗi đó.

Bước 13. Trong hàm Main, nhập đoạn mã sau để kiểm tra hoạt động của chương trình

```
// Tạo thể hiện của các lớp hình học
IHinhHoc tron = new HinhTron(10);
IHinhHoc chuNhat = new HinhChuNhat(15, 8);

// xuất chu vi của hai hình
Console.WriteLine("Chu vi hinh tron     : {0}", tron.TinhChuVi());
Console.WriteLine("Chu vi hinh chu nhat : {0}", chuNhat.TinhChuVi());

// xuất diện tích của hai hình
Console.WriteLine("Dien tich hinh tron     : {0}", tron.TinhDienTich());
Console.WriteLine("Dien tich hinh chu nhat : {0}", chuNhat.TinhDienTich());

Console.ReadKey();
```

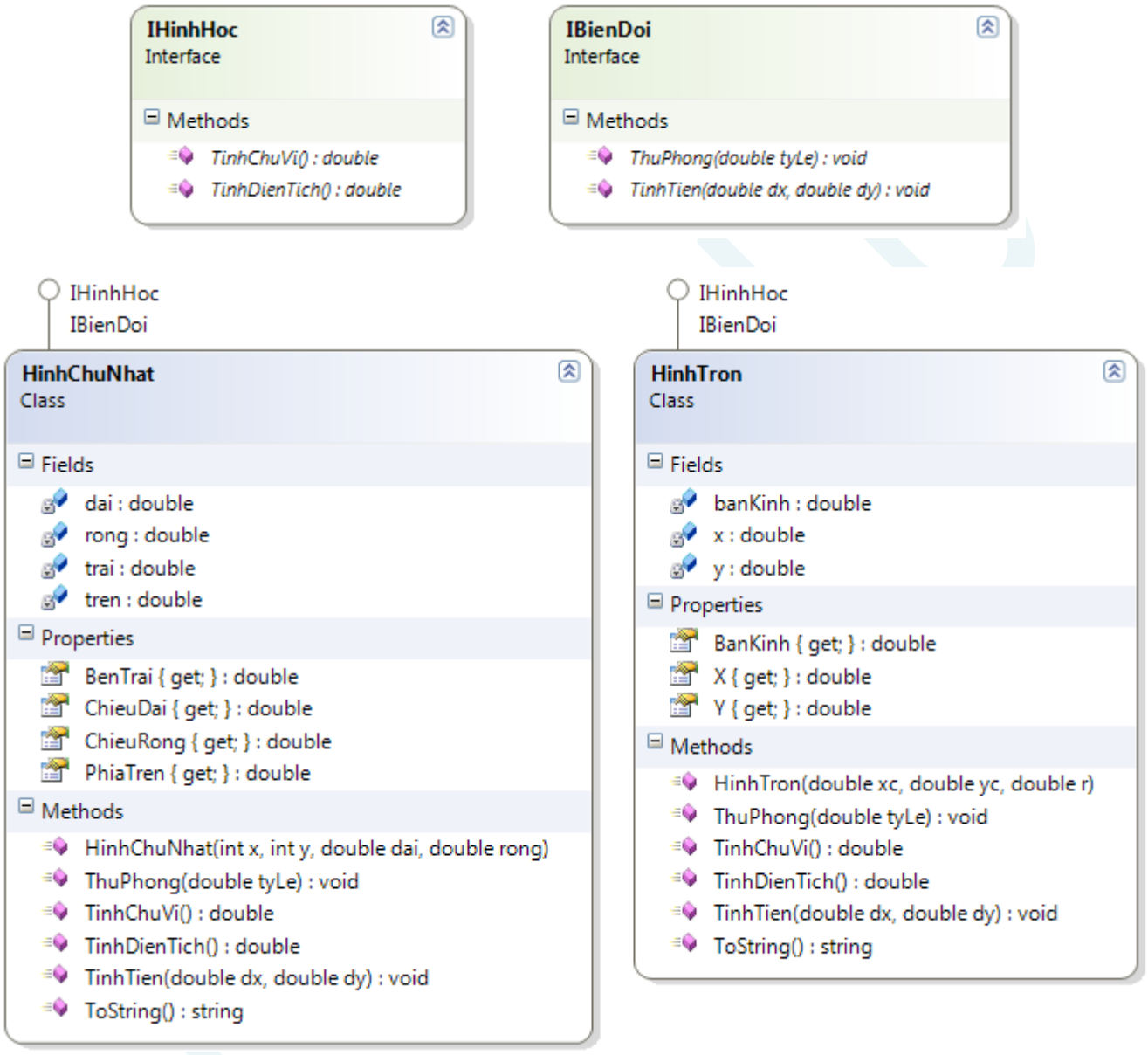
Bước 14. Chạy chương trình và ghi nhận kết quả. Cho biết kết quả có đúng hay không?

Bước 15. Tại sao trong interface IHinhHoc không định nghĩa nội dung phương thức TinhChuVi và TinhDienTich nhưng chương trình vẫn có thể tính toán và xuất ra kết quả?

Bước 16. So sánh với cách dùng lớp trừu tượng ở Lab06 và đưa ra nhận xét.

3. Thực thi nhiều giao diện

Một trong những điểm khác biệt quan trọng giữa abstract và interface là ở chổ: một lớp chỉ có thể kế thừa từ một lớp khác nhưng lại có thể thực thi nhiều giao diện. Phần này hướng dẫn cách tạo lớp thực thi nhiều giao diện để mở rộng thêm chức năng của đối tượng. Hình sau cho thấy sơ đồ các lớp và giao diện sẽ được tạo trong dự án này.



Bước 1. Tạo dự án Console Application mới, đặt tên là Lab08_Bai02_ThucThiNhieuGiaoDien

Bước 2. Tạo interface IHinhHoc như trong phần 1.

Bước 3. Tạo một interface mới, đặt tên là IBienDoi với các hàm phóng to/thu nhỏ và tịnh tiến

```
public interface IBienDoi
{
    // Khai báo nguyên mẫu hàm phóng to/thu nhỏ
```

```
        void ThuPhong(double tyLe);
        // Khai báo nguyên mẫu hàm tịnh tiến
        void TinhTien(double dx, double dy);
    }
```

Bước 4. Tiếp theo, cài đặt lớp HinhTron như sau:

```
// Lớp biểu diễn hình tròn có bán kính R.
// Lớp này thực thi giao diện IHinhHoc và IBienDoi
public class HinhTron : IHinhHoc, IBienDoi
{
    // Khai báo biến thành viên
    private double banKinh;
    private double x;       // Tọa độ của
    private double y;       // tâm hình tròn

    // Định nghĩa thuộc tính
    public double BanKinh
    {
        get { return banKinh; }
    }

    public double X
    {
        get { return x; }
    }

    public double Y
    {
        get { return y; }
    }

    // Định nghĩa phương thức khởi tạo
    public HinhTron(double xc, double yc, double r)
    {
        x = xc;
        y = yc;
        banKinh = r;
    }

    // Định nghĩa hàm tính chu vi. Lưu ý là
    // phải theo đúng nguyên mẫu trong giao diện
    public double TinhChuVi()
    {
        return 2 * banKinh * Math.PI;
    }

    // Định nghĩa hàm tính diện tích.
    public double TinhDienTich()
    {
        return banKinh * banKinh * Math.PI;
    }
```

```
        // Định nghĩa hàm phóng to, thu nhỏ hình tròn
        public void ThuPhong(double tyLe)
        {
            banKinh *= tyLe;
        }


        // Định nghĩa hàm tịnh tiến (dịch chuyển) hình tròn
        public void TinhTien(double dx, double dy)
        {
            x += dx;
            y += dy;
        }

        // Lấy chuỗi mô tả hình tròn
        public override string ToString()
        {
            return string.Format("Hinh tron tam O({0}, {1}), Ban kinh = {2}",
                                 x, y, banKinh);
        }
    }
```

Bước 5. Tạo và cài đặt lớp HinhChuNhat theo đoạn mã sau:

```
public class HinhChuNhat : IHinhHoc, IBienDoi
{
    // Khai báo biến thành viên
    private double tren;        // Tọa độ góc trên
    private double trai;        // & bên trái hình CN
    private double dai;
    private double rong;

    // Định nghĩa thuộc tính
    public double ChieuDai
    {
        get { return dai; }
    }
    public double ChieuRong
    {
        get { return rong; }
    }
    public double PhiaTren
    {
        get { return tren; }
    }
    public double BenTrai
    {
        get { return trai; }
    }

    // Định nghĩa phương thức khởi tạo
    public HinhChuNhat(int x, int y, double dai, double rong)
    {
```

```
            this.trai = x;
            this.tren = y;
            this.dai = dai;
            this.rong = rong;
        }

        // Định nghĩa hàm tính chu vi. Lưu ý là
        // phải theo đúng nguyên mẫu trong giao diện
        public double TinhChuVi()
        {
            return 2 * (dai + rong);
        }

        // Định nghĩa hàm tính diện tích. Lưu ý là
        // phải theo đúng nguyên mẫu trong giao diện
        public double TinhDienTich()
        {
            return dai * rong;
        }

        // Định nghĩa hàm phóng to, thu nhỏ hình tròn
        public void ThuPhong(double tyLe)
        {
            dai *= tyLe;
            rong *= tyLe;
        }

        // Định nghĩa hàm tịnh tiến (dịch chuyển) hình tròn
        public void TinhTien(double dx, double dy)
        {
            trai += dx;
            tren += dy;
        }

        // Lấy chuỗi mô tả hình chữ nhật
        public override string ToString()
        {
            return string.Format("Hinh chu nhat: Toa do ({0}, {1}), " +
                                 "Dai = {2}, Rong = {3}",
                                 trai, tren, dai, rong);
        }
    }
```

Bước 6. So sánh lớp HinhTron và HinhChuNhat trong dự án này với các lớp tương ứng trong dự án trước để tìm và chỉ ra các điểm khác biệt.

Bước 7. Trong hàm Main, nhập đoạn mã sau

```
// Tạo các đối tượng hình học
HinhTron tron = new HinhTron(2, 5, 10);
HinhChuNhat chuNhat = new HinhChuNhat(3, 1, 10, 7);
```

```
// Ép về kiểu IBienDoi
IBienDoi tronBd = tron, chuNhatBd = chuNhat;

// Ép về kiểu IHinhHoc
IHinhHoc hinhTron = tron, hinhCn = chuNhat;
```

Bước 8. Trong đoạn mã trên, tại sao có thể ép kiểu từ HinhTron, HinhChuNhat sang kiểu IBienDoi và IHinhHoc?

Bước 9. Trong hàm Main, bổ sung tiếp đoạn mã sau:

```
// Xuất thông tin các hình trước phóng to
Console.WriteLine("=================================================");
Console.WriteLine("Truoc khi phong to");
Console.WriteLine();

Console.WriteLine(tronBd);
Console.WriteLine("Chu vi hinh tron     : {0}", hinhTron.TinhChuVi());
Console.WriteLine("Dien tich hinh tron  : {0}", hinhTron.TinhDienTich());
Console.WriteLine();

Console.WriteLine(chuNhatBd);
Console.WriteLine("Chu vi hinh chu nhat    : {0}", hinhCn.TinhChuVi());
Console.WriteLine("Dien tich hinh chu nhat : {0}", hinhCn.TinhDienTich());
Console.WriteLine();
```

Bước 10. Chạy chương trình và ghi nhận kết quả. Cho biết kết quả có đúng hay không?

Bước 11. Nhập tiếp đoạn mã sau để kiểm tra hàm phóng to/thu nhỏ.

```
// Gọi hàm phóng to gấp đôi
tronBd.ThuPhong(2);
chuNhatBd.ThuPhong(2);

// Xuất thông tin các hình sau phóng to
Console.WriteLine("=================================================");
Console.WriteLine("Sau khi phong to");
Console.WriteLine();

Console.WriteLine(tronBd);
Console.WriteLine("Chu vi hinh tron     : {0}", hinhTron.TinhChuVi());
Console.WriteLine("Dien tich hinh tron  : {0}", hinhTron.TinhDienTich());
Console.WriteLine();

Console.WriteLine(chuNhatBd);
Console.WriteLine("Chu vi hinh chu nhat    : {0}", hinhCn.TinhChuVi());
Console.WriteLine("Dien tich hinh chu nhat : {0}", hinhCn.TinhDienTich());
Console.WriteLine();
```

Bước 12. Chạy chương trình và ghi nhận kết quả. So sánh với kết quả bước 10 và nhận xét.

Bước 13. Tiếp tục gọi và kiểm tra hàm tịnh tiến (dịch chuyển) đối tượng hình học

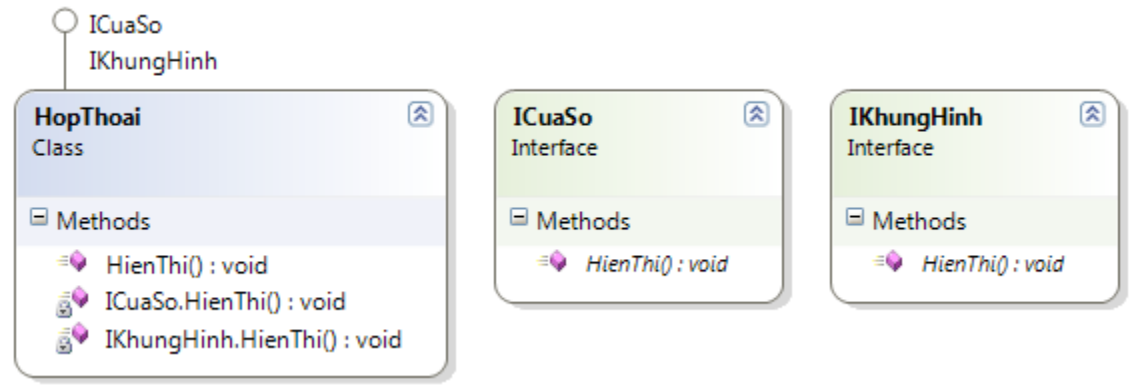
```
// Gọi hàm tịnh tiến theo vector (-5, 19)
tronBd.TinhTien(-5, 19);
chuNhatBd.TinhTien(-5, 19);

// Xuất thông tin các hình sau khi tịnh tiến

Console.WriteLine("Sau khi tinh tien");
Console.WriteLine();

Console.WriteLine(tronBd);
Console.WriteLine("Chu vi hinh tron     : {0}", hinhTron.TinhChuVi());
Console.WriteLine("Dien tich hinh tron  : {0}", hinhTron.TinhDienTich());
Console.WriteLine();

Console.WriteLine(chuNhatBd);
Console.WriteLine("Chu vi hinh chu nhat    : {0}", hinhCn.TinhChuVi());
Console.WriteLine("Dien tich hinh chu nhat : {0}", hinhCn.TinhDienTich());
Console.WriteLine();
```

Bước 14. Chạy chương trình và ghi nhận kết quả. So sánh với kết quả bước 12 và nhận xét.
Bước 15. Có thể gọi phương thức hinhTron.TinhTien(-5, 19) được không? Tại sao?
Bước 16. Có thể gọi hàm tron.TinhTien(-5, 19) hoặc tron.TinhChuVi() được không? Vì sao?
Bước 17. Tại sao lệnh Console.WriteLine(tronBd) có thể xuất thông tin của hình tròn?

4. Hiểu rõ hơn việc thực thi giao diện

Những phần trên cho thấy cách định nghĩa một giao diện, tạo các lớp thực thi một giao diện và định nghĩa các phương thức theo đúng các nguyên mẫu hàm trong các giao diện. Phần này hướng dẫn cách tạo lớp thực thi một giao diện, định nghĩa phương thức theo nguyên mẫu trong giao diện nhưng có cách thực thi khác nhau tùy thuộc vào đối tượng gọi phương thức.

Hình sau cho thấy sơ đồ các lớp và giao diện sẽ được cài đặt trong dự án này.



Bước 1. Tạo dự án Console Application mới, đặt tên là Lab08_Bai03_DaHinhPhuongThuc.
Bước 2. Tạo một interface mới với tên IcuaSo (cửa sổ) như sau:

```
public interface ICuaSo
{
    void HienThi();
}
```

Bước 3. Tạo lớp HopThoai (hộp thoại) thực thi interface IcuaSo

```
public class HopThoai : ICuaSo
{
    public void HienThi()
    {
        Console.WriteLine("Hien thi hop thoai");
    }
}
```

Bước 4. Trong hàm Main, nhập đoạn mã sau

```
HopThoai hop = new HopThoai();
ICuaSo cua = hop;

hop.HienThi();
cua.HienThi();
```

Bước 5. Chạy chương trình và ghi nhận kết quả.

Bước 6. Cài đặt lại lớp HopThoai như sau:

```
public class HopThoai : ICuaSo
{
    void ICuaSo.HienThi()
    {
        Console.WriteLine("Hien thi cua so");
    }

    public void HienThi()
    {
        Console.WriteLine("Hien thi hop thoai");
    }
}
```

Bước 7. Chạy chương trình và ghi nhận kết quả. So sánh với kết quả ở bước 5 và giải thích.

Thậm chí, một lớp có thể thực thi nhiều giao diện có phương thức trùng tên. Việc chỉ rõ tên của interface trước tên phương thức còn có tác dụng tránh gây nhập nhằng do việc trùng tên gây ra.

Bước 8. Bổ sung thêm một interface mới, đặt tên là IkhungHinh (khung hình) như sau

```
public interface IKhungHinh
{
    void HienThi();
}
```

Bước 9. Cài đặt lại lớp HopThoai như sau:

```
public class HopThoai : ICuaSo, IKhungHinh
{
    void ICuaSo.HienThi()
    {
        Console.WriteLine("Hien thi cua so");
    }

    void IKhungHinh.HienThi()
    {
        Console.WriteLine("Hien thi khung hinh");
    }

    public void HienThi()
    {
        Console.WriteLine("Hien thi hop thoai");
    }
}
```

Bước 10. Cập nhật lại mã nguồn trong hàm Main như sau:

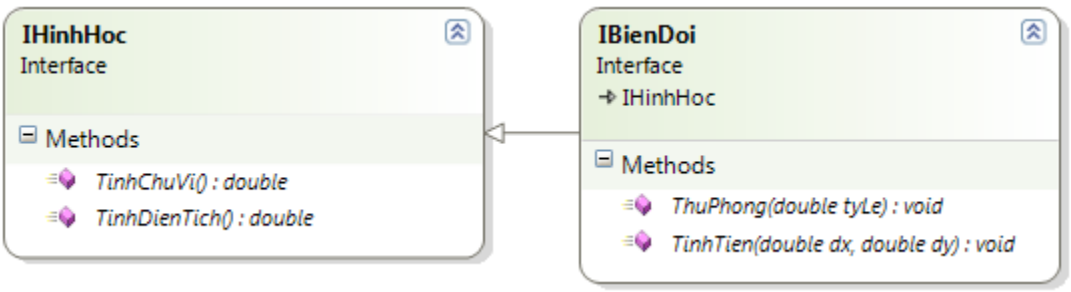
```
HopThoai hop = new HopThoai();
ICuaSo cua = hop;
IKhungHinh hinh = hop;

hop.HienThi();
cua.HienThi();
hinh.HienThi();

Console.ReadKey();
```

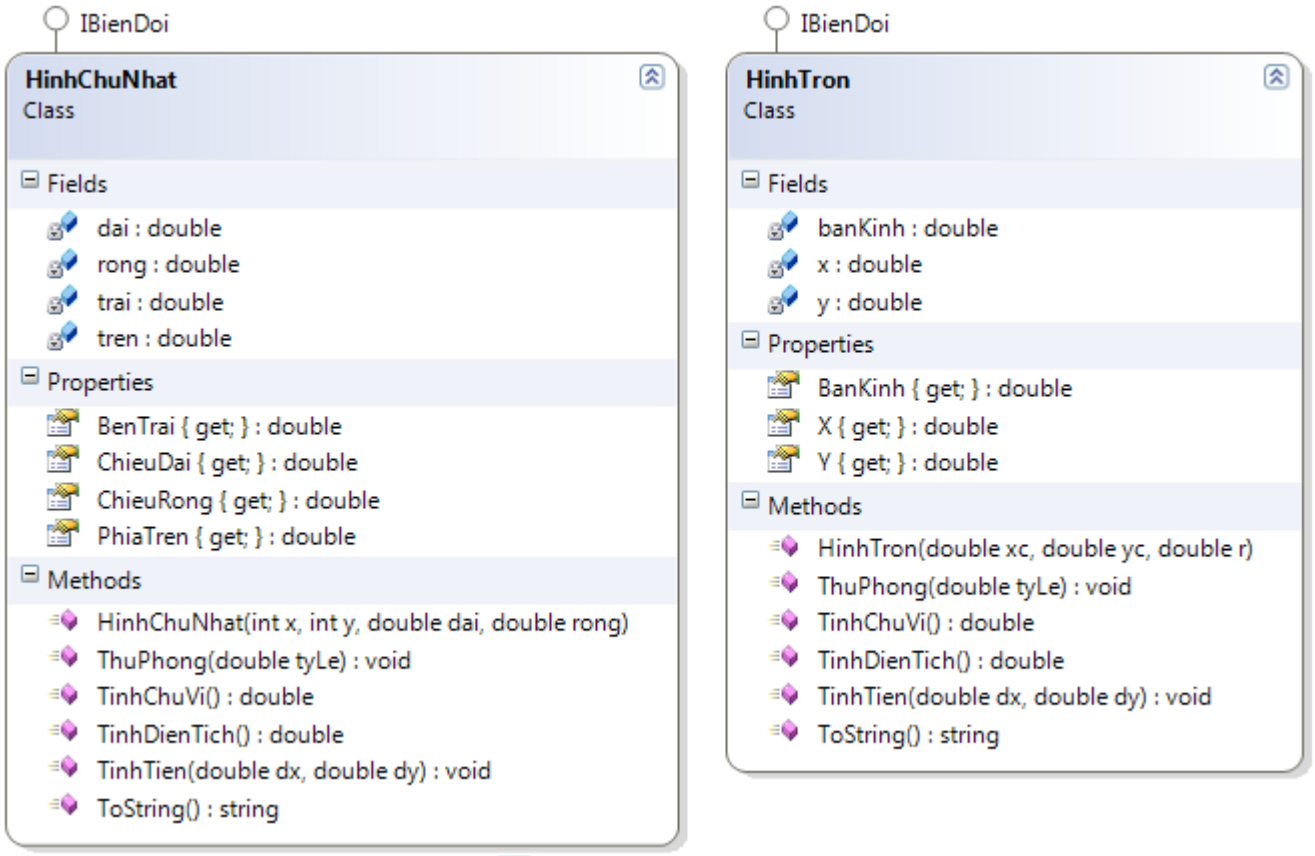
Bước 11. Chạy lại chương trình và ghi nhận kết quả.

Bước 12. Qua dự án này, bạn rút ra được điều gì?

5. Giao diện kế thừa giao diện

Ngoài cách bổ sung chức năng cho đối tượng bằng cách thực thi nhiều giao diện, ta có thể tạo một giao diện mới, kế thừa từ 1 giao diện đã có và bổ sung thêm các phương thức mới như hình dưới.





Bước 1. Tạo dự án Console Application mới, đặt tên là Lab08_Bai04_KeThuaGiaoDien

Bước 2. Tạo interface IHinhHoc như trong phần 1.

Bước 3. Tạo interface mới, đặt tên là IBienDoi, kế thừa từ IHinhHoc như sau

```
public interface IBienDoi : IHinhHoc
{
    // Khai báo nguyên mẫu hàm phóng to/thu nhỏ
    void ThuPhong(double tyLe);

    // Khai báo nguyên mẫu hàm tịnh tiến
    void TinhTien(double dx, double dy);
}
```

Bước 4. Cài đặt các lớp HinhTron và HinhChuNhat tương tự như trong Lab08_Bai02. Chỉ khác ở chổ những lớp này chỉ thực thi một interface IBienDoi.

```
public class HinhChuNhat : IBienDoi

public class HinhTron : IBienDoi
```

Bước 5. Sao chép nội dung hàm Main từ Lab08_Bai02 sang hàm Main của dự án này.

Bước 6. Chạy chương trình và ghi nhận kết quả. So sánh với Lab08_Bai02 và đưa ra nhận xét.

6. Sử dụng interface có sẵn

Trong .Net Framework đã xây dựng sẵn rất nhiều giao diện, sử dụng cho nhiều mục đích khác nhau. Phần này hướng dẫn cách sử dụng một giao diện thông dụng, đó là IComparable. Giao diện này chỉ chứa một phương thức int CompareTo(object other).

Bước 7. Tiếp tục với dự án Lab08_Bai04_KeThuaGiaoDien. Sửa đổi hai lớp HinhTron và HinhChuNhat để chúng thực thi interface IComparable.

```
public class HinhChuNhat : IBienDoi , IComparable

public class HinhTron : IBienDoi , IComparable
```

Bước 8. Trong cả hai lớp HinhTron và HinhChuNhat, bổ sung thêm phương thức sau

```
// So sánh hình hiện tại với một hình khác
// Trả về -1 nếu diện tích hình hiện tại nhỏ
// hơn hình khác, trả về 0 nếu diện tích bằng
// nhau và trả về 1 nếu ngược lại
public int CompareTo(object hinhKhac)
{
    IHinhHoc hinh = (IHinhHoc)hinhKhac;
    return this.TinhDienTich().CompareTo(hinh.TinhDienTich());
}
```

Bước 9. Trong hàm Main, nhập đoạn mã sau

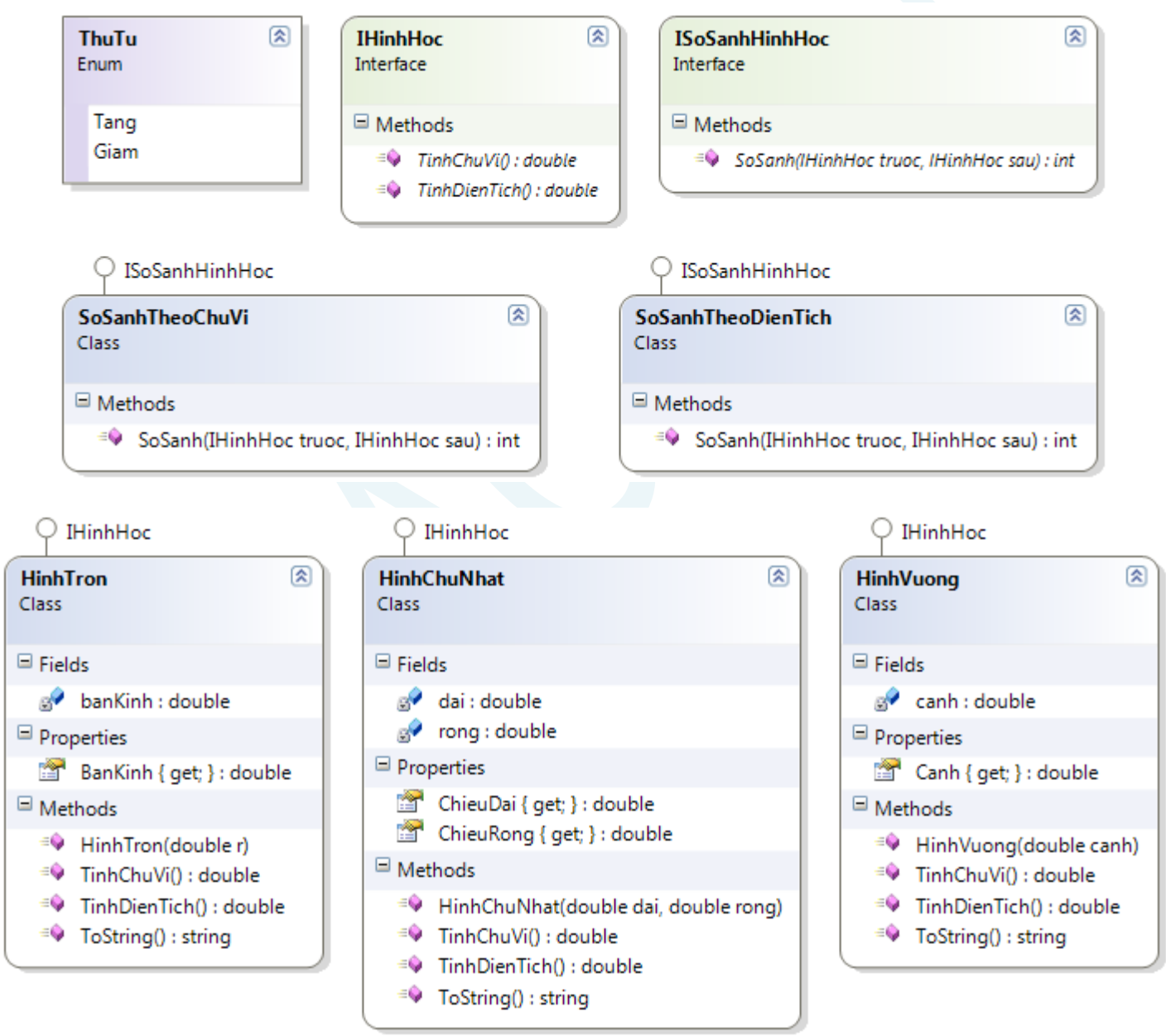
```
// Tạo thể hiện của các lớp hình học
IComparable tron = new HinhTron(10);
IComparable chuNhat = new HinhChuNhat(15, 8);

// xuất diện tích của hai hình
Console.WriteLine("Dien tich hinh tron     : {0}", ((IHinhHoc)tron).TinhDienTich());
Console.WriteLine("Dien tich hinh chu nhat : {0}", ((IHinhHoc)chuNhat).TinhDienTich());

// so sánh hai hình theo diện tích
int ketQua = tron.CompareTo(chuNhat);

switch (ketQua)
{
    case - 1: Console.WriteLine("Hinh tron co dien tich nho hon");
        break;
    case 0: Console.WriteLine("Hai hinh co dien tich bang nhau");
        break;
    case 1: Console.WriteLine("Hinh tron co dien tich lon hon");
        break;
}
```

Bước 10. Chạy chương trình và ghi nhận kết quả.

Bước 11. Tìm hiểu interface ICloneable, áp dụng cho các lớp HinhTron và HinhChuNhat để định nghĩa phương thức tạo bản sao của các đối tượng.

7. Áp dụng interface vào bài toán quản lý các đối tượng hình học

Phần này hướng dẫn cách áp dụng interface vào bài toán quản lý các đối tượng hình học đã được cài đặt trong Lab05_Bai02 và Lab06_Bai02. Qua ví dụ này, sinh viên cần nắm vững và đối sánh giữa các cách cài đặt để đưa ra nhận xét. Sơ đồ các lớp và giao diện được cho như sau:



Bước 1. Tạo dự án Console Application, đặt tên là Lab08_Bai05_QuanLyHinhHoc.

Bước 2. Tạo và cài đặt enum ThuTu với các hằng Tăng = 1, Giảm = -1.

Bước 3. Tạo và cài đặt interface IHinhHoc như trong phần 1 của Lab này.

Bước 4. Tạo và cài đặt interface ISoSanhHinhHoc có 1 phương thức như sau

```
// Định nghĩa giao diện làm khuôn mẫu
// cho các lớp thực hiện việc so sánh
// hai đối tượng hình học
public interface ISoSanhHinhHoc
{
    // Khai báo nguyên mẫu hàm so sánh hai hình
    int SoSanh(IHinhHoc truoc, IHinhHoc sau);
}
```

Bước 5. Tham khảo các phần trước để định nghĩa các lớp HinhTron, HinhChuNhat, HinhVuong theo sơ đồ lớp ở trên.

Bước 6. Cài đặt phương thức ToString của lớp HinhTron như sau:

```
public override string ToString()
{
    return string.Format("{0}{1}{2}{3}",
        "Hinh tron".PadRight(20),
        banKinh.ToString().PadRight(20),
        TinhChuVi().ToString().PadRight(19),
        TinhDienTich().ToString().PadRight(19));
}
```

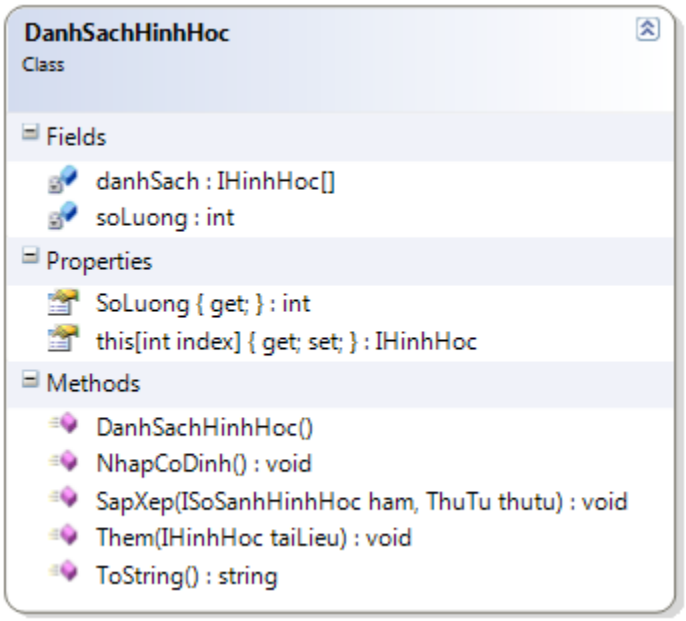
Bước 7. Cài đặt phương thức ToString của lớp HinhChuNhat như sau:

```
// Lấy chuỗi mô tả hình chữ nhật
public override string ToString()
{
    return string.Format("{0}{1}{2}{3}{4}",
        "Hinh chu nhat".PadRight(20),
        dai.ToString().PadRight(10),
        rong.ToString().PadRight(10),
        TinhChuVi().ToString().PadRight(19),
        TinhDienTich().ToString().PadRight(19));
}
```

Bước 8. Cài đặt phương thức ToString của lớp HinhTron như sau:

```
// Lấy chuỗi mô tả hình vuông
public override string ToString()
{
    return string.Format("{0}{1}{2}{3}",
        "Hinh vuong".PadRight(20),
        canh.ToString().PadRight(20),
        TinhChuVi().ToString().PadRight(19),
        TinhDienTich().ToString().PadRight(19));
}
```

Bước 9. Cài đặt thêm lớp DanhSachHinhHoc theo sơ đồ lớp sau:



Bước 10. Trong đó, phương thức SapXep được viết như sau:

```
// Sắp xếp danh sách hình học
public void SapXep(ISoSanhHinhHoc ham, ThuTu thutu)
{
    int chieu = (int) thutu;

    for (int i = 0; i < soLuong - 1; i++)
        for (int j = i + 1; j < soLuong; j++)
            if (ham.SoSanh(danhSach[i], danhSach[j]) * chieu > 0)
            {
                IHinhHoc tam = danhSach[i];
                danhSach[i] = danhSach[j];
                danhSach[j] = tam;
            }
}
```

Bước 11. Trong hàm Main, nhập đoạn mã sau

```
// Tạo danh sách các hình học
DanhSachHinhHoc cacHinh = new DanhSachHinhHoc();

// Gọi hàm nhập các hình học
cacHinh.NhapCoDinh();

// xuất danh sách các hình học
Console.WriteLine(cacHinh);
Console.WriteLine();
```

Bước 12. Chạy chương trình và ghi nhận kết quả

Bước 13. Cài đặt lớp SoSanhTheoChuVi như sau

```
public class SoSanhTheoChuVi : ISoSanhHinhHoc
{
    public int SoSanh(IHinhHoc truoc, IHinhHoc sau)
    {
        return truoc.TinhChuVi().CompareTo(sau.TinhChuVi());
    }
}
```

Bước 14. Trong hàm Main, bổ sung thêm đoạn mã dưới đây

```
// Tạo tiêu chí sắp xếp theo chu vi
ISoSanhHinhHoc dieuKien = new SoSanhTheoChuVi();

// Gọi hàm sắp xếp tăng theo chu vi
cacHinh.SapXep(dieuKien, ThuTu.Tang);

// Xuất lại danh sách
Console.WriteLine("========================================");
Console.WriteLine("Sau khi sap tang theo chu vi");

Console.WriteLine(cacHinh);
```

Bước 15. Chạy chương trình và ghi nhận kết quả.

Bước 16. Cài đặt lớp SoSanhTheoDienTich như sau

```
public class SoSanhTheoDienTich : ISoSanhHinhHoc
{
    public int SoSanh(IHinhHoc truoc, IHinhHoc sau)
    {
        double x = truoc.TinhDienTich();
        double y = sau.TinhDienTich();

        return x.CompareTo(y);
    }
}
```

Bước 17. Trong hàm Main, bổ sung thêm đoạn mã dưới đây

```
dieuKien = new SoSanhTheoDienTich();

// Gọi hàm sắp xếp giảm theo diện tích
cacHinh.SapXep(dieuKien, ThuTu.Giam);

// Xuất lại danh sách
Console.WriteLine("========================================");
Console.WriteLine("Sau khi sap giam theo dien tich");

Console.WriteLine(cacHinh);
```

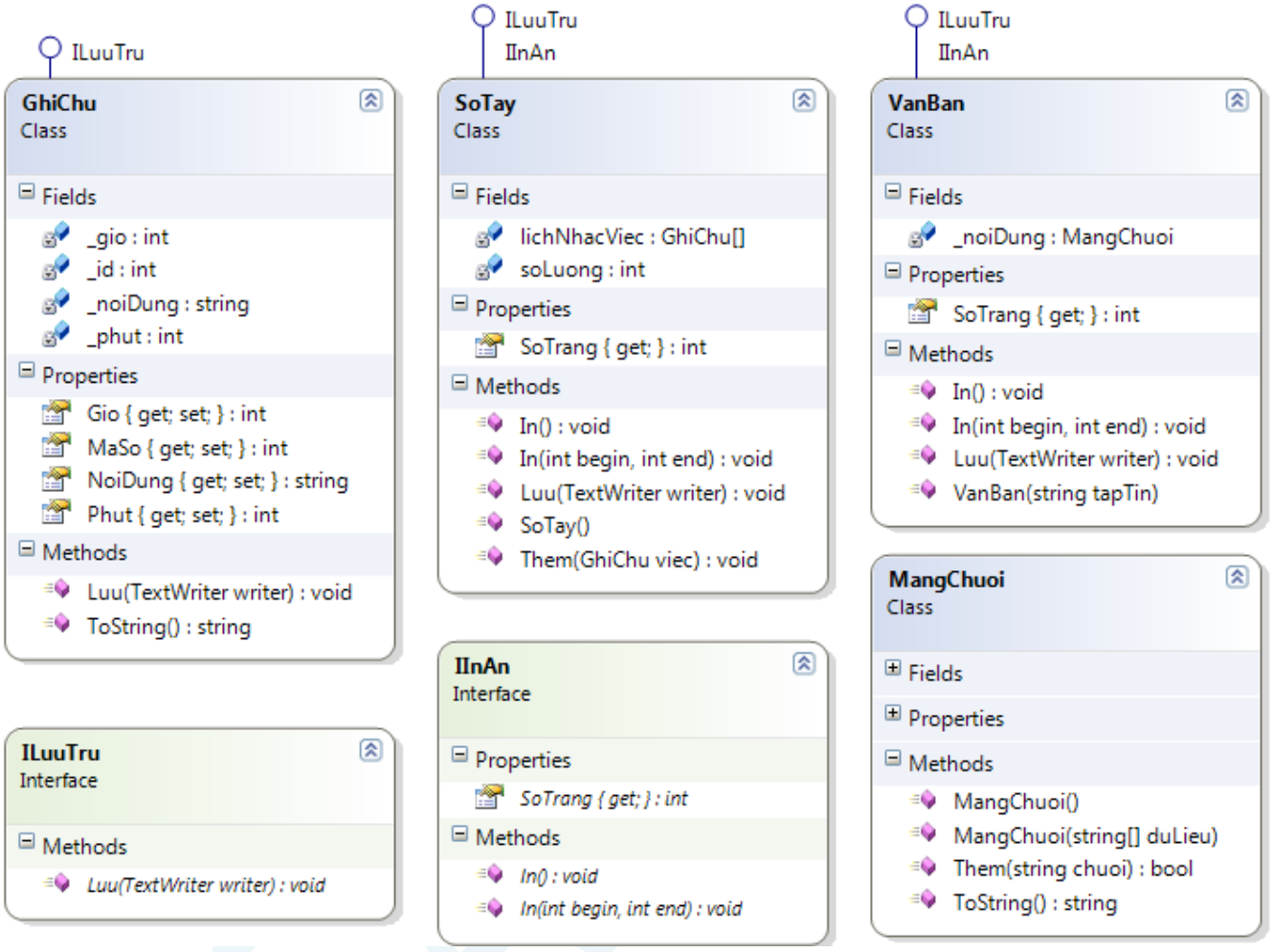
Bước 18. Chạy chương trình và ghi nhận kết quả.
Bước 19. Mô tả lại một cách vắn tắt hoạt động của chương trình trên.
Bước 20. Giải thích cách xử lý của phương thức SapXep.

D. Bài tập bắt buộc

1. Tiếp tục dự án quản lý hình học Lab08_Bai05_QuanLyHinhHoc, hãy cài đặt các chức năng sau
   a. Định nghĩa một interface mới, đặt tên là IDieuKienTimKiem. Interface này có một phương thức: bool KiemTra(IHinhHoc hinh).
   b. Cài đặt các lớp thực thi interface vừa tạo để kiểm tra một đối tượng hình học có phải là hình vuông, hình tròn hay hình chữ nhật.
   c. Cài đặt lớp thực thi interface trên để kiểm tra diện tích của hình có bằng giá trị cho trước.
   d. Trong lớp DanhSachHinhHoc, cài đặt phương thức tìm các đối tượng hình học theo nguyên mẫu: public DanhSachHinhHoc TimKiem(IDieuKienTimKiem ham).
   e. Bổ sung thêm enum LoaiHinh với 3 loại hình học ở trên.
   f. Trong interface IHinhHoc, bổ sung phương thức LoaiHinh LayKieuHinh( ). Cài đặt (hay thực thi) phương thức này ở các lớp hình học.
   g. Tính tổng diện tích các hình học
   h. Tính tổng chu vi các hình học
   i. Tính tổng chu vi của các hình vuông
   j. Tìm hình theo loại
   k. Tìm diện tích của hình lớn nhất.
   l. Tìm hình chữ nhật có diện tích lớn nhất
   m. Tìm hình có giá trị chu vi lớn hơn giá trị diện tích (không tính đơn vị đo)
   n. Xóa hình tại vị trí (viTri) cho trước
   o. Xóa một hình học (hinh) cho trước
   p. Xóa các hình thỏa điều kiện: void XoaTheoDK(IDieuKienTimKiem dieuKien)
   q. Đếm số lượng hình học mỗi loại
   r. Tìm hình có chu vi nhỏ nhất nhưng không phải là hình vuông
   s. Xuất danh sách hình chữ nhật tăng dần theo chu vi
   t. Liệt kê các hình chữ vuông hoặc hình tròn giảm dần theo diện tích
   u. Tìm hình tròn có bán kính nhỏ nhất
   v. Tìm hình tròn có khả năng nội tiếp một hình vuông (hinhVuong) cho trước (R = cạnh/2).
   w. Đếm số lượng hình vuông có diện tích lớn hơn diện tích mọi hình tròn.
   x. Tính diện tích trung bình của các hình chữ nhật
   y. Tính độ chênh lệch nhỏ nhất về diện tích giữa các hình.
   z. Cài đặt lớp Menu để xử lý các chức năng trên

Gợi ý: Với mỗi điều kiện tìm kiếm/sắp xếp khác nhau, tạo ra một lớp mới để kiểm tra điều kiện.

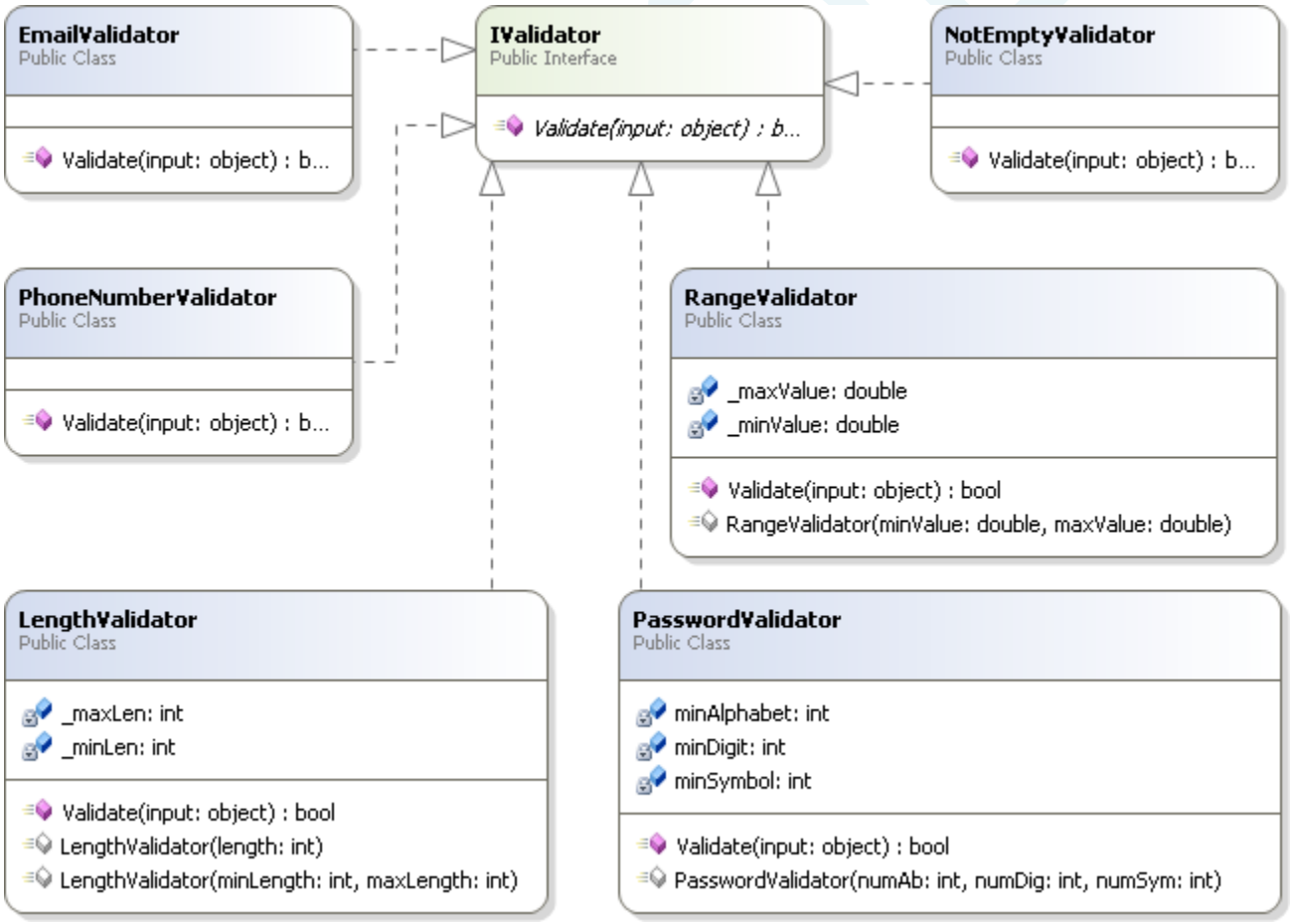
2. Viết chương trình quản lý việc in ấn và lưu trữ tài liệu theo sơ đồ lớp và yêu cầu dưới đây



a. Cài đặt lớp MangChuoi để lưu một mảng các chuỗi.
b. Cài đặt hai giao diện ILuuTru (lưu trữ) và IInAn (in ấn).
c. Cài đặt lớp VanBan (văn bản) trong đó, nội dung của văn bản là một mảng các chuỗi. Ta giả sử số trang của văn bản bằng số chuỗi.
   - Phương thức khởi tạo có tham số là đường dẫn của một tập tin văn bản. Để đọc hết nội dung tập tin và lấy ra các dòng, ta dùng phương thức: System.IO.File.ReadAllLines( ).
   - Phương thức In( ) và In(begin, end) sẽ xuất nội dung văn bản ra màn hình. Begin và end là số trang bắt đầu & kết thúc.
   - Phương thức Luu(writer) sẽ lưu các chuỗi vào tập tin. Để lưu một chuỗi, dùng lệnh sau: writer.WriteLine(chuoi).
d. Cài đặt lớp GhiChu (biểu diễn cho một công việc phải làm).
e. Cài đặt lớp SoTay (sổ tay, sổ ghi chú hay lịch nhắc việc). Các phương thức In và Lưu tương tự như lớp VanBan. Riêng begin, end là mã số nhỏ nhất và lớn nhất của các ghi chú muốn in.
f. Trong hàm Main, tạo 1 thể hiện của lớp SoTay, trong đó chứa 10 đối tượng có kiểu GhiChu.

g. Lưu sổ tay vào tập tin sotay.txt. Gợi ý: sử dụng lệnh sau để tạo thể hiện của lớp TextWriter: TextWriter writer = System.IO.File.CreateText("sotay.txt");
h. Gọi phương thức In( ) để xuất mọi ghi chú ra màn hình.
i. Gọi phương thức In(2, 7) để xuất các ghi chú có mã số từ 2 đến 7 ra màn hình.
j. Trong dự án hiện tại, tạo 4 tập tin văn bản với nội dung bất kỳ. Trong đó, mỗi dòng trong tập tin có tối thiểu là 50 ký tự, tối đa 80 ký tự, tối thiểu 10 dòng và tối đa 25 dòng.
k. Tạo 4 thể hiện của lớp VanBan ứng với 4 tập tin vừa tạo.
l. Gọi hàm để xuất nội dung của từng văn bản ra màn hình.
m. Lưu nội dung của 4 văn bản trong câu l vào chung một tập tin.
n. Lưu nội dung của sổ tay ở câu f và 4 văn bản ở câu l vào chung một tập tin.
o. In danh sách những công việc phải làm trong khoảng thời gian từ 10 tới 11 giờ.
p. In những công việc phải làm tính từ thời điểm hiện tại.

3. Cài đặt các lớp thực hiện việc xác nhận tính hợp lệ của dữ liệu theo sơ đồ lớp sau:



Trong đó:

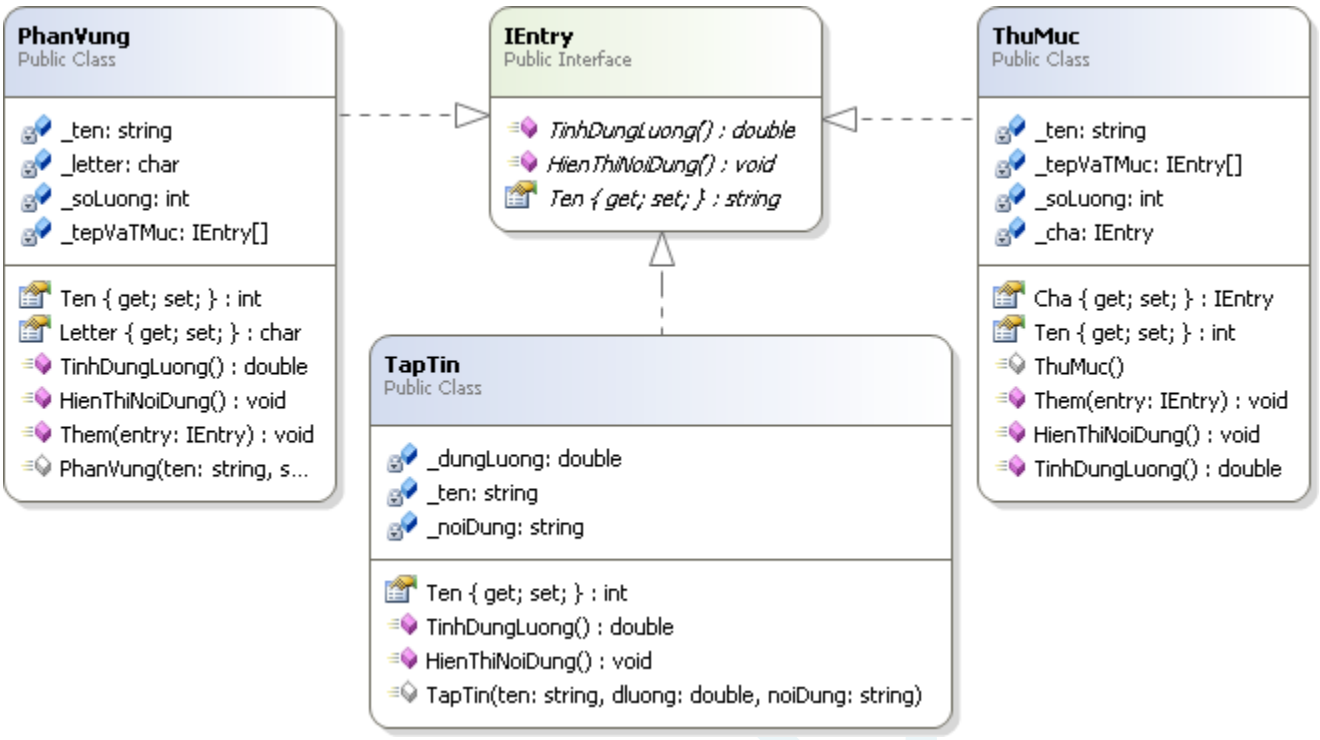
- Mũi tên nét đứt nối giữa A và B dùng để ám chỉ lớp A thực thi interface B.

- Phương thức Validate(input) dùng để kiểm tra đối tượng input có hợp lệ hay không.
- EmailValidator dùng để kiểm tra chuỗi input có phải là một địa chỉ email hợp lệ.
- PhoneNumberValidator dùng để kiểm tra chuỗi input có phải là một số điện thoại hợp lệ. Ta giả sử một số điện thoại là hợp lệ nếu nó có chiều dài 10 hoặc 11 chữ số, bắt đầu bởi số 0.
- NotEmptyValidator dùng để kiểm tra một chuỗi là khác null, khác khoảng trắng.
- RangeValidator dùng để kiểm tra một số input có nằm trong đoạn cho trước hay không.
- LengthValidator dùng để kiểm tra chuỗi input có chiều dài nằm trong đoạn cho trước không
- PasswordValidator dùng để kiểm tra chuỗi input có thể mà một mật khẩu hợp lệ hay không. minAlphabet, minDigit, minSymbol tương ứng là số chữ cái, số chữ số, số ký tự đặc biệt tối thiểu phải có.

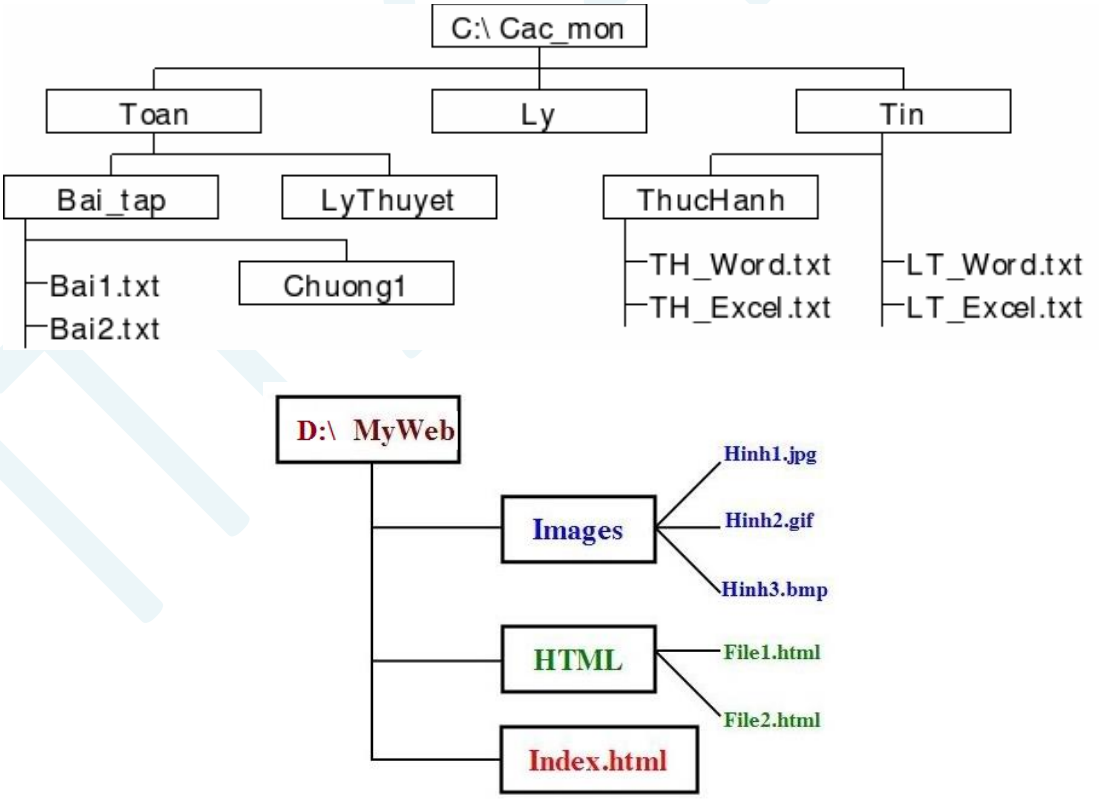
Thực hiện tiếp các yêu cầu sau:

a. Cài đặt lớp Profile (thông tin cá nhân) theo hình bên.
b. Tạo một thể hiện của lớp Profile chứa thông tin cá nhân của chính bạn.
c. Sử dụng các lớp đã cài đặt để kiểm tra các thông tin cá nhân theo yêu cầu sau:
   - ✓ Những thông tin dạng chuỗi phải khác rỗng và khác khoảng trắng.
   - ✓ Tuổi (Age) phải nằm trong đoạn từ 15 tới 55
   - ✓ Lương (Salary) phải nằm trong đoạn 1 triệu tới 100 triệu
   - ✓ Địa chỉ mail (Email) phải hợp lệ
   - ✓ Mật khẩu (Password) phải có tối thiểu 3 chữ số, 2 ký tự đặc biệt và 3 chữ cái
   - ✓ Tên đăng nhập (Username) phải dài ít nhất 10 ký tự
   - ✓ Họ và tên (FullName) phải có từ 15 tới 30 ký tự

4. Viết chương trình quản lý các tập tin thư mục theo mô tả sau.
   a. Một phân vùng hay ổ đĩa chỉ có thể chứa các tập tin hoặc thư mục.
   b. Một thư mục cũng có thể chứa các tập tin hoặc thư mục khác.
   c. Phân vùng hay thư mục không thể chứa một phân vùng khác
   d. Phương thức HienThiNoiDung dùng để xuất nội dung tập tin ra màn hình hoặc xuất danh sách thư mục và tập tin được chứa trong ThuMuc hay PhanVung hiện tại. Yêu cầu: xuất đầy đủ thông tin của tập tin, thư mục, ổ đĩa với định dạng và canh lề.
   e. Phương thức TinhDungLuong dùng để lấy dung lượng của tập tin hoặc tính tổng dung lượng các tập tin & thư mục trong một thư mục hay phân vùng.
   f. _tepVaTMuc là danh sách các tập tin hay thư mục chứa trong thư mục hay trong phân vùng
   g. _soLuong là số lượng tập tin hay thư mục chứa trong thư mục khác hay trong phân vùng.



Tạo thể hiện của các lớp để biểu diễn cây thư mục sau (Dung lượng & nội dung tập tin nhập tùy ý):

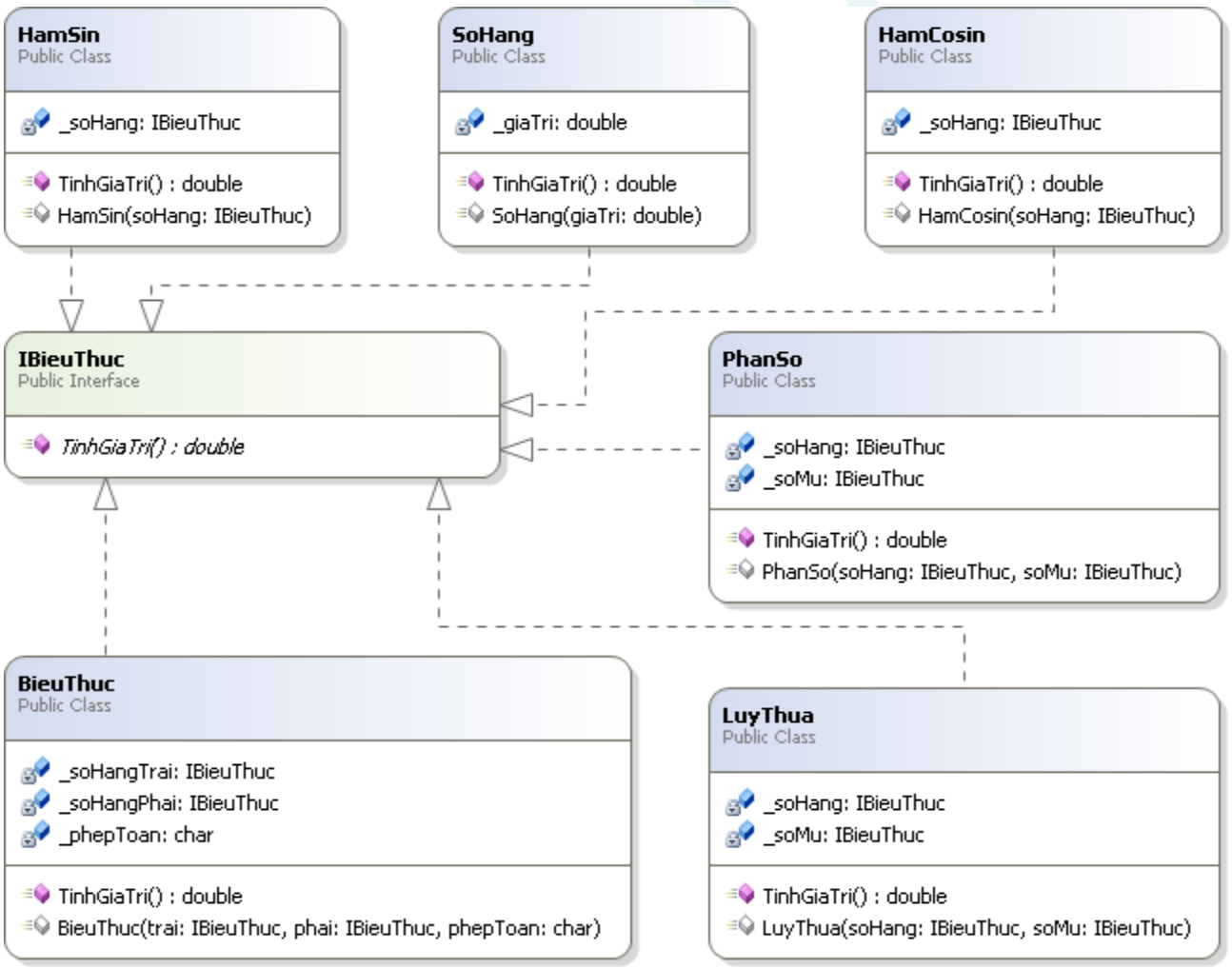


E. Bài tập làm thêm

1. Bổ sung vào dự án Lab08_Bai05_QuanLyHinhHoc tính năng tô màu cho các đối tượng hình học.

2. Viết chương trình quản các đối tượng hình học 3 chiều (3D) như hình cầu, hình hộp chữ nhật, hình nón, hình trụ, hình chóp, … Trong đó có hàm tính thể tích của các hình và tô màu cho hình.

3. Cài đặt các lớp để biểu diễn một biểu thức toán học và tính giá trị của biểu thức đó.



Trong đó, _phepToan trong lớp BieuThuc là các phép toán cơ bản: +, -, *, /.

a. Thử nghiệm với biểu thức sau

$$\frac{0.09 \times \left(\sin 65 + e^{\frac{2}{\pi}}\right) \times \left[\frac{\cos 110 \times 2.75^{15}}{9.11 \times \frac{13}{4} + \frac{\pi^3}{e}} + 10 \times \cos 70\right]}{2 \times \frac{\pi}{\sin \frac{\pi}{5}} + 3 \times \left[12 + e^{\pi} \times \frac{3 \times (9 \times 2.5 - 35)}{6 \times \pi - 77}\right]}$$

b. Cài đặt thêm các lớp để tính Logarit, Logarit nêpe, căn bậc 2, tan, cotan, giai thừa … Sau đó, thử nghiệm với biểu thức sau

$$\frac{0.09 \times \left(\sin 65 + e^{\frac{2}{\pi}}\right) \times \left[\frac{\cos 110 \times 2.75^{15}}{9.11 \times \frac{13}{4} + \frac{\pi^3}{e}} + \ln(10 \times \cos 70)\right]}{2 \times tan\ \frac{\pi}{\sin \frac{\pi}{5}} + 3^{7+\frac{1}{2}} \times \left[12! + \sqrt{e^{\pi}} \times \frac{3 \times (9 \times 2.5 - 35)}{6 \times \pi - 77}\right]}$$

4. Viết chương trình giải các phương trình bậc nhất, bậc hai, bậc ba, phương trình trùng phương và hệ tuyến tính bậc nhất 2 ẩn.