MÔN : LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG
Bài thực hành số 8.4 : Xây dựng chương trình ghi/đọc hệ thống đối tượng

I. Mục tiêu :

- Giúp SV làm quen với cách thức viết code để ghi/đọc hệ thống đối tượng gồm nhiều đối tượng có mối quan hệ tham khảo lẫn nhau sao cho dễ dàng, trong sáng, an toàn và tin cậy nhất.

II. Nội dung :

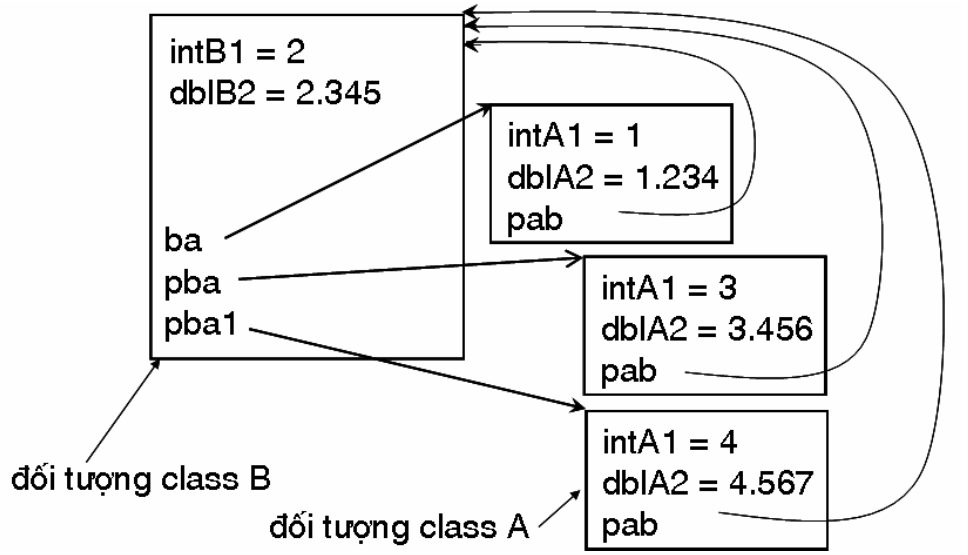
- Viết code để ghi/đọc đối tượng b có trạng thái cụ thể như hình vẽ dưới đây.

III. Chuẩn đầu ra :

- Sinh viên nắm vững và lập trình thành thạo các đoạn code để đọc/ghi đối tượng cần thiết trong chương trình của mình.

IV. Qui trình :

Giả sử ta có hệ thống các đối tượng với trạng thái và mối quan hệ giữa chúng cụ thể như sau. Lưu ý chúng có mối quan hệ bao gộp dạng vòng :



1. Chạy VS .Net, chọn menu File.New.Project để hiển thị cửa sổ New Project.
2. Mở rộng mục Visual C# trong TreeView "Project Types", chọn mục Windows, chọn icon "Console Application" trong listbox "Templates" bên phải, thiết lập thư mục chứa Project trong listbox "Location", nhập tên Project vào textbox "Name:" (td. WRObject), click button OK để tạo Project theo các thông số đã khai báo.
3. Ngay sau Project vừa được tạo ra, cửa sổ soạn code cho chương trình được hiển thị. Thêm lệnh các using sau đây vào đầu file :

```
using System.IO;
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Binary;
```

4. Viết code cho hàm Main và các hàm dịch vụ khác nhau sau :

```
class Program {
    static String fbuf;
    static void Main(string[] args) { //điểm nhập của chương trình
        //xây dựng hệ thống đối tượng và ghi lên file
        Create_SaveObject();
        //đọc lại hệ thống đối tượng
```

```
        //ReadObject();
    }
    //hàm xây dựng hệ thống đối tượng và ghi lên file
    public static void Create_SaveObject() {
        //khở tạo đối tượng b theo hình ở silde 24
        B b = new B();
        b.init(2,2.345);
        b.Setba(1,1.234,b);
        b.Setpba(3,3.1416,b);
        b.Setpba1(4,4.567,b);
        //ghi đối tượng b dùng kỹ thuật Serialization
        try {
            //1. định nghĩa đối tượng FileStream miêu tả file chứa kết quả
            FileStream fs = new FileStream("c:\\data.obj", FileMode.Create);
            //2. tạo đối tượng BinaryFormatter phục vụ ghi đối tượng
            BinaryFormatter formatter = new BinaryFormatter();
            //3. gọi tác vụ Serialize  của formatter để ghi đối tượng
            formatter.Serialize(fs,b);
            //4. đóng file lại
            fs.Flush(); fs.Close();
        } catch (Exception e) { Console.WriteLine(e.ToString()); }
    }
    //hàm đọc đối tượng b dùng kỹ thuật Serialization
    public static void ReadObject() {
     try {
        //1. định nghĩa đối tượng FileStream miêu tả file chứa dữ liệu đã có
        FileStream fs = new FileStream("c:\\data.obj", FileMode.Open);
        //2. tạo đối tượng BinaryFormatter phục vụ đọc đối tượng
        BinaryFormatter formatter = new BinaryFormatter();
        //3. gọi tác vụ Deserialize để đọc đối tượng từ file vào
        B b = (B) formatter.Deserialize(fs);
        //4. đóng file lại
        fs.Close();
      } catch (Exception e) { Console.WriteLine(e.ToString()); }
    } //hết hàm Main
} //hết class program
```

5. Ấn phải chuột vào phần tử gốc của cây Project trong cửa sổ Solution Explorer, chọn option Add.Class, đặt tên là A.cs để tạo ra file đặc tả class A. Khi cửa sổ hiển thị mã nguồn của class A hiển thị, đặc tả class A như đoạn code dưới đây :

```
//thêm lệnh using sau ở đầu file
using System.Runtime.Serialization;
namespace WRObject {
    [Serializable]
    public class A {
        //định nghĩa các thuộc tính dữ liệu
        private int intA1;
        private double dblA2;
        private B pab;
        //định nghĩa các tác vụ
        public A() {}
        public void init(int a1, double a2, B p) {
            this.intA1 = a1;
```

```
            this.dblA2 = a2;
            this.pab = p;
        }
    }
}
```

6. Ấn phải chuột vào phần tử gốc của cây Project trong cửa sổ Solution Explorer, chọn option Add.Class, đặt tên là B.cs để tạo ra file đặc tả class B. Khi cửa sổ hiển thị mã nguồn của class B hiển thị, đặc tả class B như đoạn code dưới đây :

```
//thêm lệnh using sau ở đầu file
using System.Runtime.Serialization;
namespace WRObject  {
    [Serializable]
    public class B {
        //định nghĩa các thuộc tính dữ liệu
        private    int intB1;
        private    double dblB2;
        private    A ba;
        private   A pba;
        private   A pba1;
        //định nghĩa các tác vụ
        public B() { }
public void init(int b1, double b2) {
    this.intB1 = b1; this.dblB2 = b2;
 ba = new A(); pba = new A(); pba1 = new A();
}
public void Setba (int a1, double a2, B b) {
 this.ba.init (a1,a2,b);
}
public void Setpba (int a1, double a2, B b) {
 this.pba.init (a1,a2,b);
}
public void Setpba1 (int a1, double a2, B b) {
 this.pba1.init (a1,a2,b);
}
} //hết class B
} //hết namespace WRObject
```

7. Chọn menu Debug.Start Debugging để dịch và chạy ứng dụng. Hệ thống đối tượng sẽ được tạo ra và lưu lên file c:\data.obj.
8. Hiển thị cửa sổ soạn mã nguồn file Program.cs, chú thích lệnh gọi Create_SaveObject(); và bỏ chú thích lệnh gọi ReadObject();. Dời chuột về lệnh "B b = (B) formatter.Deserialize(fs);" trong hàm ReadObject(), click chuột vào lệnh trái của lệnh này để thiết lập điểm dừng. Chọn menu Debug.Start Debugging để dịch và chạy ứng dụng. Ứng dụng sẽ dừng ở lệnh dừng. Dời chuột về biến b, ta thấy cửa sổ hiển thị giá trị của biến này lúc này là null.
9. Ấn F10 để thực hiện đúng lệnh này rồi dừng lại, dời chuột về biến b, rồi mở rộng cây phân cấp miêu tả nội dung biến b và thấy nó chứa đúng thông tin như slide 14, nghĩa là chương trình đã đọc được toàn bộ hệ thống đối tượng đã lưu giữ trước đây.