A. TÓM TẮT LÝ THUYẾT

1. Kế thừa

Khái niệm là tiếp thu tiếp nhận những cái đã được xây dựng.

Nếu lớp B là lớp kế thừa từ lớp A thì toàn bộ các thành phần của lớp A cũng có mặt trong lớp B mà không cần phải khai báo trong lớp B. Lớp A được gọi là lớp cơ sở hay lớp cha, còn lớp B được gọi là lớp dẫn xuất hay lớp con.

Kế thừa có hai loại: Đơn kế thừa và đa kế thừa.

Ở chương này chúng ta chỉ nghiên cứu về đơn kế thừa còn đa kế thừa sẽ được thể hiện rõ hơn ở chương giao diện.

Cách khai báo lớp dẫn xuất kế thừa một lớp cơ sở:

Tên lớp dẫn xuất: Tên lớp cơ sở

Ví dụ: Lớp A kế thừa lớp B thì khai báo như sau:

class A:B

- Đơn kế thừa là một lớp dẫn xuất chỉ được kế thừa từ một lớp cơ sở

Ví dụ khai báo đơn kế thừa

```
class class1
{
    public class1()
    {
        //Code of contructor
    }
}
class class2 : class1 // Khai báo lớp Class2 kế thừa lớp Class1
{
    public class2()
    {
        //Code of contructor
    }
}
```

2. Từ khóa base

Sử dụng để truy nhập các thành viên của lớp cơ sở từ lớp dẫn xuất.

3. Từ khóa new

Khi khai báo phương thức trong lớp kế thừa mà có tên trùng với tên của phương thức (không có từ khóa abstract hay virtual) trong lớp cơ sở thì phải sử dụng từ khóa new để che giấu phương thức của lớp cơ sở.

Ví dụ:

```
class diem
{
    public void nhap()
    { }
}
class tamgiac : diem
{
    public new void nhap()
    { }
}
```

4. Tính đa hình

Bao gồm những đặc điểm sau:

+ Nó được hiểu như là khả năng sử dụng nhiều hình thức của một thực thể mà không cần quan tâm đến từng chi tiết cụ thể.

+ Phương thức là đa hình có từ khóa virtual ở trước và phương thức này phải được ghi đè ở lớp dẫn xuất.

+ Cho phép cài đặt phương thức của lớp dẫn xuất trong khi thi hành

Ví dụ: Khai báo phương thức đa hình ở lớp cơ sở

public **virtual** float dientich()

5. Ghi đè

Ghi đè phương thức khi có nhu cầu cho phép người lập trình thay đổi hay mở rộng các thành viên (phương thức) cùng tên của lớp cơ sở trong lớp dẫn xuất thì bạn sử dụng từ khóa override.

Ví dụ:

```
class hinh
{
    public virtual float dientich()
    { }
}
```

```
class hinhcn : hinh
{
    public override float dientich() // phương thức ghi đè
    { }
}
```

6. Nghiêm cấm kế thừa

Một lớp được cho là nghiêm cấm các lớp khác kế thừa khi nó được khai báo với từ khóa là sealed ngay trước từ khóa class. Do vậy bất kì lớp nào khai báo kế thừa lớp đó đều phát sinh lỗi khi biên dịch.

Ví dụ: Khai báo lớp nghiêm cấm kế thừa

sealed class sinhvien

7. Cách truy xuất protected

Bạn cần một lớp dẫn xuất để thâm nhập vào các thành phần của lớp cơ sở. Nhưng bạn lại không muốn khai báo các thành phần của lớp cơ sở đó là công khai (public). Cách tốt nhất là dùng cách truy xuất có bảo vệ protected trong định nghĩa của lớp cơ sở. Chúng đều được xử lí chung như bên trong phạm vi lớp dẫn xuất nhưng nó vẫn được coi là riêng ở mọi nơi khác. Cách truy nhập protected chỉ mở rộng cho lớp con.

8. Lớp trừu tượng

Là lớp có ít nhất một phương thức trừu tượng. Khi xây dựng lớp trừu tượng thì mọi thành viên được định nghĩa trong lớp trừu tượng đều sử dụng từ khóa abstract.

Phương thức trừu tượng không có sự thực thi. Phương thức này chỉ đơn giản tạo ra một tên phương thức và kí hiệu phương thức. Nó không định nghĩa phần thân, thay vào đó chúng được cài đặt trong phương thức ghi đè của lớp dẫn xuất.

Để khai báo lớp trừu tượng bạn sử dụng từ khóa **abstract** trước từ khóa **class** như cú pháp sau:

abstract *class baseclass*

{

// Code of members

}

9. Sự khác nhau giữa phương thức đa hình với phương thức trừu tượng

Phương thức đa hình phần thân được định nghĩa tổng quát, ở lớp dẫn xuất có thể thay đổi phương thức đó. Phương thức trừu tượng phần thân không được định nghĩa và nó được cài đặt trong phương thức của lớp dẫn xuất.

10. Gốc của tất cả các lớp (Lớp Object)

Trong C #, các lớp kế thừa tạo thành cây phân cấp và lớp cao nhất (hay lớp cơ bản nhất) chính là lớp Object. Bất cứ kiểu dữ liệu nào thì cũng được dẫn xuất từ lớp System.Object. Lớp Object cung cấp một số các phương thức dùng cho lớp dẫn xuất có thể thực hiện việc phủ quyết. Sau đây là bảng tóm tắt các phương thức của lớp Object.

Ví dụ minh họa việc kế thừa phương thức ToString() của lớp Object

```
using System;
namespace Object
{
    public class Object
    {
        private int value;
        public Object(int val)
        {
            value = val;
        }
        public virtual string ToString()
        {
            return value.ToString();
        }
    }
    class tester
    {
        static void Main(string[] args)
        {
            int i = 5;
            Console.WriteLine("Gia tri cua i:{0}", i.ToString());
            Object A = new Object(7);
            Console.WriteLine("Gia tri cua A:{0}", A.ToString());
            Console.ReadLine();
        }
    }
}
```

Kết quả sau khi thực hiện chương trình:

Giá trị của i: 5
Giá trị của A: 7

11. Boxing và Unboxing

Boxing là tiến trình chuyển đổi một kiểu giá trị thành kiểu Object. Nó là ngầm định khi ta cung cấp một giá trị tham chiếu đến giá trị này và giá trị được chuyển đổi ngầm định.

Ví dụ: Minh họa boxing

```
using System;
class Boxing
{
    public static void Main()
    {
        int i = 123;
        Console.WriteLine("The object value:{0}", i);
        Console.ReadLine();
    }
}
```

Unboxing: trả kết quả của một đối tượng về kiểu giá trị, ta thực hiện "mở" tường minh nó. Ta thiết lập theo 2 bước sau:

1. Chắc chắn rằng đối tượng là thể hiện của một giá trị đã được boxing
2. Sao chép giá trị từ thể hiện này thành giá trị của biến.

Ví dụ: Minh họa boxing và unboxing

```
using System;
public class Unxing
{
    public static void Main()
    {
        int i = 123;
        object o = i;
        int j = (int)o;
        Console.WriteLine("j:{0}", j);
        Console.ReadLine();
    }
}
```

12. Lớp lồng nhau

Lớp được khai báo trong thân của một lớp được gọi là lớp nội hay lớp lồng, lớp kia là lớp ngoài. Lớp lồng có khả năng truy cập tới các thành viên của lớp ngoài. Một phương thức của lớp lồng có thể truy xuất đến thành viên dữ liệu **private** của lớp ngoài. Hơn nữa lớp nội ẩn trong lớp ngoài so với các lớp khác, nó có thể là thành viên kiểu **private** của lớp ngoài. Lớp lồng được khai báo với từ khóa **internal** trước từ khóa class.

Cú pháp:

public class A

{

internal class B

}

B. BÀI TẬP MẪU

I. Kế thừa

1. Xây dựng lớp dẫn xuất thừa kế từ lớp cơ sở.

Bài 1

Xây dựng lớp có tên là lương để tính lương cho cán bộ với các thông tin sau: Họ tên, lương cơ bản, hệ số lương, với các phương thức: Khởi tạo không tham số dùng để khởi tạo lương cơ bản là 450, hệ số lương là 2,3. Phương thức thiết lập 2 tham số dùng để khởi tạo lương cơ bản, hệ số lương.

Sau đó kế thừa lớp có tên là lương được xây dựng ở trên dùng để tính lương mới cho các cán bộ với việc bổ sung thêm hệ số phụ cấp, lương được tính lại như sau:

Lương=lương cơ bản * hệ số lương * hệ số phụ cấp.

a. Hướng dẫn:

Bài toán này được xây dựng gồm có 2 đối tượng: **class luong** và **class luongmoi**. Nhìn vào bài toán trên ta thấy cả hai lớp có những thuộc tính và phương thức giống nhau như: lương cơ bản, hệ số lương, các phương thức khởi tạo, tính lương. Theo như lập trình cấu trúc thì bạn phải đi xây dựng tất cả các thuộc tính và phương thức cho cả hai lớp trên. Ngược lại với lập trình hướng đối tượng thì việc giải quyết lại đơn giản hơn. Bạn chỉ cần xây dựng một lớp class luong, còn lớp class luongmoi sẽ kế thừa lại các thuộc tính và phương thức giống lớp class luong chứ không phải xây dựng lại. Nó chỉ xây dựng các thuộc tính và phương thức riêng của nó mà class luong không có.

```
//Lớp cơ sở
class luong
{
    // Thuộc tính:
    private string hoten;
    private double hesoluong;
    private static int luongcoban;
```

```
    // Lưu ý: Lương cơ bản là thuộc tính không thay đổi. Vì vậy, ta nên khai báo với từ
khóa static

    //Phương thức:
    public void nhap()
    public luong() // Phương thức khởi tạo không tham số
    public luong(int lcb, float hsl) // Phương thức khởi tạo 2 tham số
    public double tinhluong()
    public void hien()

}
//Lớp dẫn xuất
class luongmoi
{
    // Thuộc tính:
    private double hesophucap;
    // Phương thức:
    public new void nhap()
    public luongmoi() // Phương thức khởi tạo không tham số
    public luongmoi(int lcb, double hsl, double hspc) // Phương thức khởi tạo 2 tham số
    public new double tinhluong()
    public new void hien()
}
```

b. Bài giải mẫu:

```
using System;
class luong
{
    private string hoten;
    private double hesoluong;
    private static int luongcoban;
    public void nhap()
    {
        Console.Write("Nhap ho ten: ");
        hoten = Console.ReadLine();
        Console.Write("He so luong: ");
        hesoluong = double.Parse(Console.ReadLine());
    }
    public luong()
    {
        luongcoban = 450;
        hesoluong = 2.34;
    }
    public luong(int lcb, float hsl)
    {
        luongcoban = lcb;
        hesoluong = hsl;
    }
    public void hien()
    {
        Console.WriteLine("Ho ten: {0}", hoten);
        Console.WriteLine("He so luong: {0}", hesoluong);
    }
    public double tinhluong()
    {
        return luongcoban * hesoluong;
    }
```

```
}
class luongmoi : luong
{
    private double hesophucap;
    public luongmoi() : base() // Gọi phương thức thiết lập không tham số của lớp cơ sở
    {
        hesophucap = 0.4;
    }
    public luongmoi(int lcb, float hsl, double hspc) : base(lcb, hsl)
    {
        // Gọi phương thức thiết lập 2 tham số của lớp cơ sở
        hesophucap = hspc;
    }
    public new void nhap()
    {
        base.nhap();
        Console.Write("He so phu cap:");
        hesophucap = double.Parse(Console.ReadLine());
    }
    public new void hien()
    {
        base.hien();
        Console.WriteLine("He so phu cap = {0}", hesophucap);
    }
    public new double tinhluong()
    {
        return base.tinhluong() * hesophucap;
    }
}
class tester
{
    static void Main()
    {
        luongmoi A = new luongmoi();
        A.nhap();
        A.hien();
        Console.WriteLine("Luong moi: {0}", A.tinhluong());
        Console.ReadKey();
    }
}
```

Kết quả sau khi thực hiện chương trình:

Họ tên: Nguyễn Văn A
Hệ số lương: 3.6
Hệ số phụ cấp: 4.3
Lương mới: 6966

Kết luận: Mục đích xây dựng bài toán trên là giúp bạn biết cách xây dựng lớp dẫn xuất kế thừa từ một lớp cơ sở, cách gọi phương thức khởi tạo, phương thức của lớp cơ sở thông qua từ khóa **base.** Và trả lời một số câu hỏi sau: Khi nào thì xây dựng một lớp kế thừa từ lớp khác? Và mục đích của việc kế thừa?

- Một lớp được kế thừa từ lớp khác khi một lớp đó có các thuộc tính và phương thức giống với một lớp nào đó đã được xây dựng.

- Mục đích của việc kế thừa giúp chúng ta không phải đi xây dựng những gì đã có. Do vậy bạn đã tiết kiệm thời gian viết mã lệnh và sử dụng tài nguyên có hiệu quả. Hơn nữa điều đó còn giúp cho chương trình sáng sủa, và tránh được các lỗi khi lập trình.