CHƯƠNG 3

KẾ THỪA

Nội dung

- Các khái niệm cơ bản
- Xây dựng lớp kế thừa
- Các thành phần được/không được kế thừa
- Cài đặt ứng dụng kế thừa
- Phương thức khởi tạo và sự kế thừa

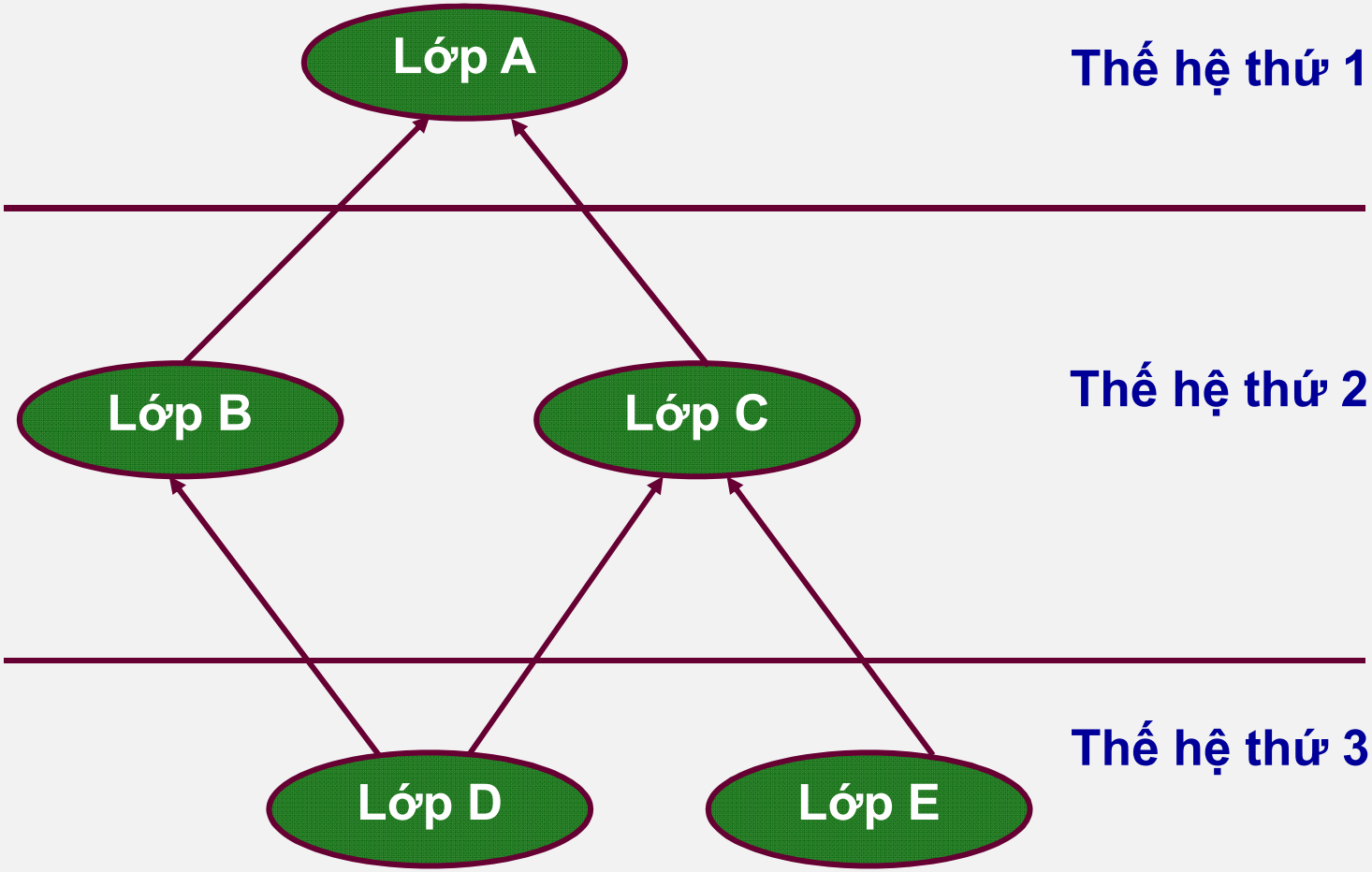
4.1. Các khái niệm

- Lớp mới B, nhận được các thuộc tính, phương thức từ một lớp A đã được định nghĩa.
- Nghĩa là lớp B được thừa kế từ lớp A
- Khi đó:
  - Lớp A được gọi là lớp cơ sở (lớp cha)
  - Lớp B được gọi là lớp dẫn xuất (lớp con)
- Lớp B dẫn xuất trực tiếp từ lớp A gọi là kế thừa trực tiếp
- Lớp B dẫn xuất trực tiếp từ lớp A, lớp C dẫn xuất trực tiếp từ lớp B, khi đó lớp C cũng được kế thừa các thành phần của lớp A.
- Việc lớp C kế thừa lớp A gọi là kế thừa gián tiếp

Các khái niệm (tt)

- **Kế thừa đơn: Một lớp dẫn xuất chỉ kế thừa trực tiếp các thành phần từ duy nhất một lớp cơ sở.**
- **Kế thừa bội (đa kế thừa): Một lớp dẫn xuất có thể kế thừa trực tiếp các thành phần từ nhiều lớp cơ sở.**
- **Phạm vi kế thừa:**
  - ***Kế thừa public:*** **Tất cả các thành phần public/protected trong lớp cơ sở trở thành các thành phần public/protected trong lớp dẫn xuất.**
  - ***Kế thừa private:*** **Tất cả các thành phần public/protected trong lớp cơ sở sẽ trở thành các thành phần private trong lớp dẫn xuất.**
- **Các thành phần private trong lớp cơ sở không thể được kế thừa.**
- **Không phải tất cả các thành phần có thể kế thừa từ lớp "cha" xuống lớp "con" lại có thể được lớp "cháu" kế thừa.**

Các khái niệm (tt)



4.2. Xây dựng lớp kế thừa

- Xây dựng các lớp cơ sở: Khi đó các thành phần được kế thừa phải có phạm vi truy xuất là public hoặc protected.
- Xây dựng các lớp dẫn xuất theo mẫu

```
class <lớp_dẫn_xuất>:<phạm vi kế thừa> <lớp_cơ_sở>
{
   //Nội dung lớp dẫn xuất
};
```

- <Phạm vi kế thừa>: có thể là public hoặc private.
- Nếu là kế thừa bội thì các lớp cơ sở được đặt cách nhau bởi dấu phảy.
- Lớp cơ sở phải được định nghĩa trước đó.

Xây dựng lớp kế thừa (tt) – Ví dụ

- Xây dựng các lớp cơ sở

```
class A{
   protected:
         int a;
   public:
         void nhap();
         …
};
```

```
class B{
   protected:
         long b;
   public:
         void nhap();
         …
};
```

Xây dựng lớp kế thừa (tt) – Ví dụ

- Xây dựng các lớp dẫn xuất

```
class C : public A, private B {
   private:
         float c;
   public:
         void nhap();
         …
};
```

```
class D : public C {
   protected:
         double d;
   public:
         void nhap();
         …
};
```

4.3. Các thành phần được/không được kế thừa

- Các thành phần private ở lớp cơ sở không được kế thừa ở các lớp dẫn xuất.
- Các phương thức khởi tạo cũng không được kế thừa.
- Các thành phần protected hoặc public của lớp cơ sở được kế thừa.
- Phạm vi kế thừa
  - Nếu phạm vi kế thừa là private thì các thành phần được kế thừa ở lớp dẫn xuất có phạm vi truy xuất là private.
  - Nếu phạm vi kế thừa là public thì các thành phần được kế thừa ở lớp dẫn xuất có phạm vi truy xuất giống như lớp cơ sở.

4.4. Cài đặt ứng dụng kế thừa

1) Cài đặt chương trình gồm các yêu cầu:

- Cài đặt lớp Diem (điểm) gồm các thuộc tính là hoành độ và tung độ trên mặt phẳng và các phương thức cần thiết.
- Cài đặt lớp DoanThang (đoạn thẳng) gồm các thuộc tính là hai điểm mút và các phương thức cần thiết.
- Cài đặt lớp TamGiac (tam giác) kế thừa lớp đoạn thẳng và có thêm một điểm và các phương thức cần thiết.
- Cài đặt hàm main() thực hiện nhập tọa độ 3 đỉnh của một tam giác, tính và in ra màn hình diện tích tam giác.

Cài đặt ứng dụng kế thừa (tt)

2) Cài đặt chương trình thực hiện các yêu cầu:

- Cài đặt lớp SanPham (sản phẩm) gồm các thuộc tính: Mã sản phẩm, tên sản phẩm, ngày sản xuất, trọng lượng, màu sắc và các phương thức cần thiết.
- Cài đặt lớp HangDienTu (hàng điện tử) kế thừa lớp sản phẩm và có thêm các thuộc tính: Công suất, Dòng điện sử dụng (1 hay 2 chiều) và các phương thức:
  - nhap(): nhập các thông tin của hàng điện tử.
  - xuat(): xuất các thông tin lên màn hình.
- Cài đặt các chức năng:
  - Nhập vào một danh sách n hàng điện tử.
  - In danh sách của các hàng điện tử lên màn hình.
  - In ra màn hình các Mặt hàng có trọng lượng thấp nhất.

4.5. Phương thức khởi tạo và sự kế thừa

- Các phương thức khởi tạo không được kế thừa.
- Thứ tự gọi phương thức khởi tạo trong hệ thống kế thừa:
  - Giả sử lớp B kế thừa lớp A, lớp C kế thừa lớp B và mỗi lớp A, B, C đều có các phương thức khởi tạo là A(), B(), C().
  - Khi khai báo một đối tượng lớp C thì chương trình sẽ tạo ra một đối tượng lớp B để đối tượng lớp C kế thừa, một đối tượng lớp A để đối tượng lớp B kế thừa.
  - Khi đó thứ tự sẽ là A() được gọi để tạo đối tượng lớp A, B() được gọi để tạo đối tượng lớp B, và C() được gọi để tạo đối tượng lớp C.

Phương thức khởi tạo và sự kế thừa (tt)

- Giả sử có lớp A như sau:

```
class A {
        protected:
                int a;
        public:
                A(int x){
                        a = x;
                }
};
```

- Giả sử lớp B kế thừa trực tiếp từ lớp A và lớp B cũng có phương thức khởi tạo có đối.

Phương thức khởi tạo và sự kế thừa (tt)

- **Vấn đề nảy sinh:**
  - Khi khai báo ĐT lớp B thì ĐT lớp A cũng được tạo ra để ĐT lớp B kế thừa.
  - Cần phải truyền tham số cho đối tượng lớp A -> truyền như thế nào?

```
class B : public A {
  protected:
      int b;
  public:
      B(int x, int y): A(x)
      {     b = y;
      }
  };
```

```
int  main()
{
   B dtb(2, 5);
   …
}
```