Bài 7

KẾ THỪA

Company: DEVPRO VIỆT NAM

Website: devpro.edu.vn

Design by Minh An

1

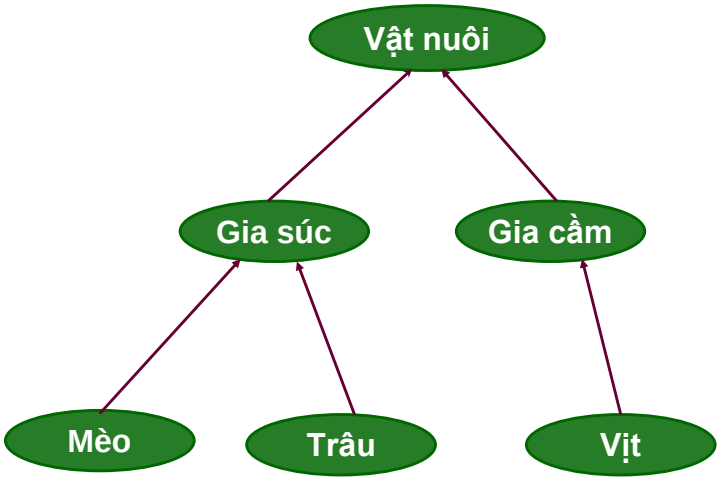
1. Một số khái niệm

- 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).
- 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.
  - Đa kế thừa không được hỗ trợ trong java thông qua lớp.

Design by Minh An

2

Một số khái niệm (tt)



Design by Minh An

3

2. 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.

Design by Minh An

4

3. 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> extends <lớp_cơ_sở>
{
   //Nội dung lớp dẫn xuất
};
```

Design by Minh An

5

Xây dựng lớp kế thừa (tt) – Ví dụ

- Xây dựng lớp cơ sở

```
public class Animal {
   protected String name;
   public Animal() {
      this.name = "";
   }
   public Animal(String name) {
      this.name = name;
   }
   public String getName() {
        return name;
   }
}
```

Design by Minh An

6

Xây dựng lớp kế thừa (tt) – Ví dụ

- **Xây dựng lớp dẫn xuất**

```
public class Cat extends Animal {
   protected int age;
   protected double height;
   public Cat() {
      super();
      this.age = 0;
      this.height = 0;
   }
   public Cat(String name,int age,int height) {
      super(name);
      this.age = age;
      this.height = height;
   }
   ...
}
```

Design by Minh An

7

4. Chỉ định truy xuất và kế thừa

- **Chỉ định truy xuất với lớp.**
  - **public: Lớp dẫn xuất có thể nằm bất kỳ đâu trong chương trình.**
  - **default: Lớp dẫn xuất phải trong cùng package.**
- **Chỉ định truy xuất với các thành phần của lớp.**
  - **Các thành phần protected hoặc public của lớp cơ sở được kế thừa ở bất kỳ lớp dẫn xuất nào.**
  - **Các thành default chỉ cho phép lớp dẫn xuất trong cùng package kế thừa và sử dụng.**
  - **Các thành phần private không được kế thừa.**

Design by Minh An

8

5. 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.

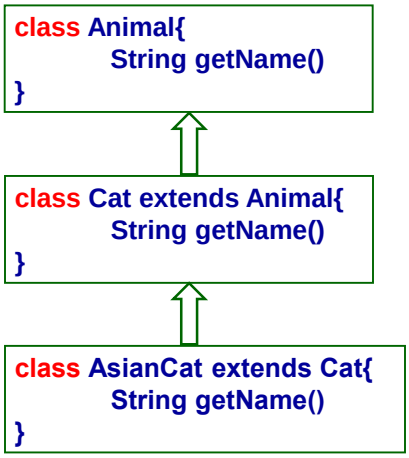
Cài đặt ứng dụng kế thừa (tt)

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.

6. Tính đa hình - polymorphism

- Tính đa hình cho phép một phương thức có tác động khác nhau trên các đối tượng khác nhau trong chuỗi kế thừa.
- Ví dụ:



Design by Minh An

11

Tính đa hình (tt) – ví dụ

```
public class Animal {
   protected String name;
   public Animal(String name) {
      this.name = name;
   }
   public void display() {
        System.out.println("Lop Animal");
   }
}
```

Design by Minh An

12

Tính đa hình (tt) – ví dụ

```
public class Cat extends Animal {
   protected int age;
   public Cat(String name,int age) {
      super(name);
      this.age = age;
   }
   @Override
   public void display() {
        System.out.println("Lop Cat");
   }
}
```


Tính đa hình (tt) – ví dụ

```
public class AsianCat extends Cat {
   public AsianCat(String name,int age) {
      super(name,age);
   }
   @Override
   public void display() {
        System.out.println("Asian Cat");
   }

}
```

Tính đa hình (tt)

- Lớp con có thể định nghĩa lại phương thức được kế thừa từ lớp cha.
  - Ghi đè (overriding).
  - Nạp chồng (overloading).
- Ghi đè:
  - Giữ nguyên tên phương thức và danh sách đối số. Khi đó phương thức của lớp cha sẽ bị che đi.
- Nạp chồng:
  - Giữ nguyên tên phương thức và thay đổi danh sách đối số trong cùng lớp. Phương thức lớp cha được gọi bình thường.
- Lưu ý:
  - Không được thay đổi kiểu dữ liệu trả về của phương thức.

Tính đa hình (tt) - Ứng dụng

- Cho các lớp:
  - Lớp HangHoa gồm các thuộc tính: Mã hàng, tên hàng, số lượng, giá tiền và các phương thức: khởi tạo, xuat().
  - Lớp HangDienTu kế thừa lớp HangHoa và có thêm các thuộc tính: Hãng sản xuất, công suất và các phương thức: khởi tạo, xuat().
  - Lớp TiVi kế thừa lớp HangDienTu và có thêm các thuộc tính: Nhãn hiệu, kích thước màn hình (inchs) và các phương thức: khởi tạo, xuat().
  - Lớp MayTinh kế thừa lớp HangDienTu và có thêm các thuộc tính: Dung lượng HDD, dung lượng RAM, tốc độ CPU và các phương thức: khởi tạo, xuat().

Tính đa hình (tt) - Ứng dụng

- **Cài đặt chương trình:**
  - **Cài đặt các lớp theo mô tả như trên.**
  - **Tạo một danh sách hỗn hợp gồm 3 tivi và 4 máy tính.**
  - **Hiển thị danh sách hàng hóa hỗn hợp.**
  - **Hiển thị danh sách tivi.**
  - **Hiển thị danh sách máy tính.**

Design by Minh An

17

7. Lớp trừu tượng

- **Là lớp được khái quát hóa.**
- **Lớp chỉ được dùng để kế thừa (lớp cha).**
- **Lớp trừu tượng không có thể hiện (không tạo đối tượng).**
- **Các phương thức của lớp trừu tượng có thể là phương thức trừu tượng, chỉ được khai báo, không được định nghĩa.**
- **Các phương thức trừu tượng được định nghĩa cụ thể ở các lớp con (overriding).**
- **Lớp có phương thức trừu tượng thì phải là lớp trừu tượng.**

Design by Minh An

18

Lớp trừu tượng (tt) – Khai báo

```
public abstract class ClassName{
   //Data
   //Methods
}
```


Lớp trừu tượng (tt) – Ví dụ

```
public abstract class Person{
   String name;
   public abstract void show();
}
```

Lớp trừu tượng (tt) – Ví dụ

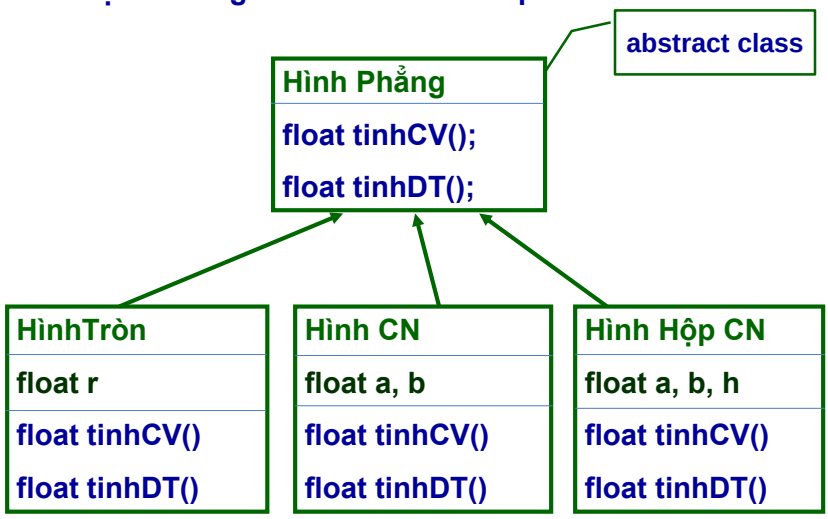
```
public class Student extends Person{
   String nameClass;
   double mark;
   @Override
   public void show(){
      System.out.println(name);
      System.out.println(nameClass);
      System.out.println(mark);
   }
}
```


Lớp trừu tượng (tt) – Ví dụ

```
public class Teacher extends Person{
   String faculty;
   double salary;
   @Override
   public void show(){
      System.out.println(name);
      System.out.println(faculty);
      System.out.println(salary);
   }
}
```

Lớp trừu tượng (tt) – Ví dụ

- Cài đặt chương trình theo sơ đồ lớp



Design by Minh An

23

Lớp trừu tượng (tt) – Ví dụ

- **Một công ty sản xuất có 2 loại nhân viên:**
  - Nhân viên sản xuất: lương = sản phẩm * 10.000.
  - Nhân viên văn phòng: lương = mức lương – ngày nghỉ * 10.000.
  - Thông tin chung của nhân viên công ty gồm: Mã nhân viên, họ và tên; năm vào làm.
  - Ngoài lương được tính như trên, mỗi nhân viên còn được phụ cấp một khoản tiền là 100000. Và khoản tiền này cứ tăng thêm 20000 cho mỗi năm công tác ở công ty.
- **Viết chương trình:**
  - Nhập vào danh sách nhân viên công ty.
  - Tính tổng số tiền công ty phải trả cho nhân viên mỗi tháng.

Design by Minh An

24

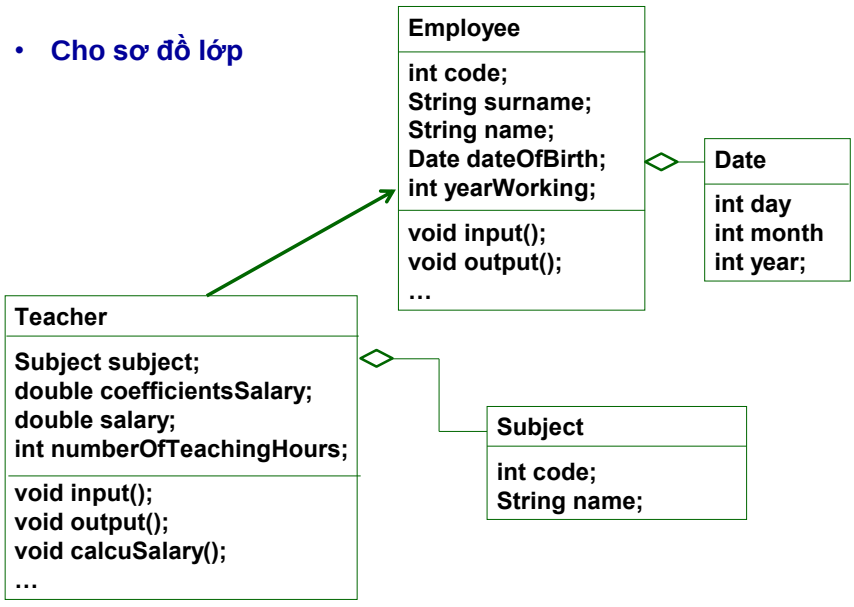
BÀI TẬP

Design by Minh An

25

Bài tập 1

- Cho sơ đồ lớp



Design by Minh An

26

Bài tập 1

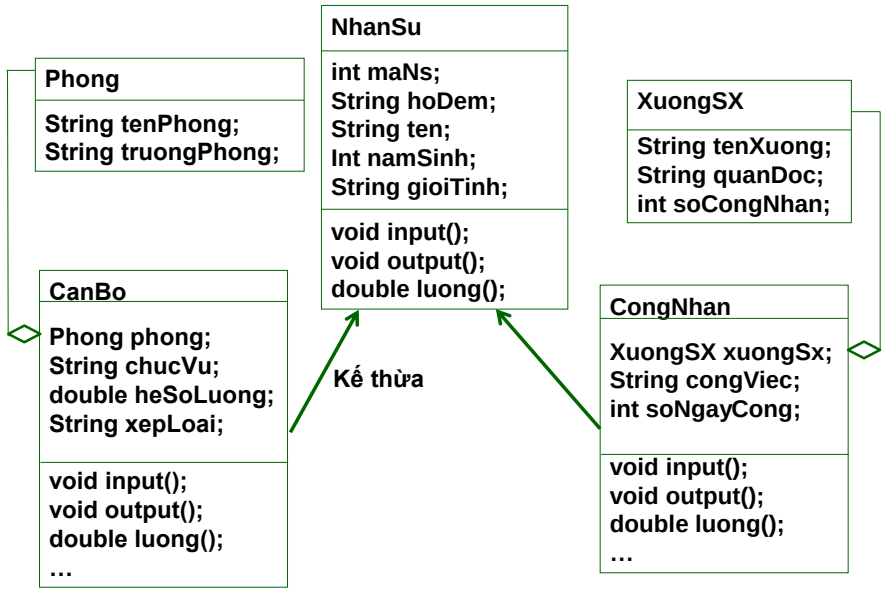
- Cài đặt chương trình theo sơ đồ lớp trên và thực hiện các chức năng.
  - Tạo danh sách n teacher và sử dụng phương thức calcuSalary() để tính salary cho teacher theo công thức.

  salary = coefficientsSalary * 1150000 +
  numberOfTeachingHours * 60000.
  - Hiển thị danh sách teacher.
  - Hiển thị danh sách các teacher có lương thấp hơn 5000000.
  - Thêm một teacher mới vào vị trí k trong danh sách, hiển thị danh sách sau khi thêm.
  - Tính tổng salary của tất cả các teacher và hiển thị kết quả.
  - Chuyển các teacher không có giờ dạy xuống cuối danh sách, hiển thị danh sách.
  - Sắp xếp danh sách theo name với thứ tự từ điển, hiển thị danh sách sau khi sắp.

Design by Minh An

27

Bài tập 2: Cho sơ đồ lớp



Design by Minh An

28

Bài tập 2

- Cài đặt chương trình theo sơ đồ lớp trên và thực hiện các chức năng.
  - Tạo danh sách n nhân sự gồm cả cán bộ và công nhân (chọn 1 để nhập cán bộ, chọn 2 để nhập công nhân) và sử dụng phương thức tinhLuong() để tính lương cho nhân sự theo công thức.

  Lương công nhân = (nếu số ngày công >= 23: soNgayCong * 250000 + 500000; nếu số ngày công >=20: soNgayCong*250000 + 300000; nếu số ngày công >= 18: soNgayCong*250000; còn lại: soNgayCong*250000 – 200000).

  Lương cán bộ = (nếu xếp loại A: heSoLuong*1150000 + 1000000; nếu xếp loại B: heSoLuong*1150000; xếp loại C: heSoLuong*1150000 – 400000; còn lại: heSoLuong*1150000 – 1000000).

Design by Minh An

29

Bài tập 2

- Hiển thị danh sách nhân sự.
- Hiển thị danh sách công nhân nam.
- Hiển thị danh sách cán bộ nữ.
- Hiển thị danh sách cán bộ có lương cao nhất.
- Hiển thị danh sách công nhân có lương thấp nhất.
- Tính tổng salary của tất cả các cán bộ và hiển thị kết quả.
- Tính tổng lương của tất cả công nhân, hiển thị kết quả.
- Chuyển các công nhân xuống cuối danh sách, hiển thị danh sách.
- Sắp xếp danh sách nhân sự theo tên với thứ tự từ điển, hiển thị danh sách sau khi sắp.

Design by Minh An

30

8. Interface – Đa kế thừa

- Một Interface trong Java là một bản thiết kế của một lớp, nhưng không phải là một lớp.
- Không có constructor, chỉ có các phương thức trừu tượng.
- Interface là một kỹ thuật để thu được tính trừu tượng hoàn toàn và đa kế thừa trong Java.
- Interface không thể được khởi tạo, nó giống như lớp trừu tượng.
- Các trường của Interface là public, static và final theo mặc định và các phương thức là public và abstract.
- Một Interface trong Java là một tập hợp các phương thức trừu tượng (abstract).
- Một class triển khai một interface, do đó kế thừa các phương thức abstract của interface.

8.1. Khai báo Interface

- Inerface được khai báo trong file.java, biên dịch thành file.class.
- **Cú pháp:**

```
[modifier] interface InterfaceName {
   [modifier] <final data>;
   [modifier] DataType method(args);
}
```

- Chọn modifier là public để sử dụng mọi nơi.
- default: chỉ dùng cho gói.
- Modifier trong interface nên là public để dễ dùng.

Khai báo Interface – Ví dụ

```
public interface Shape {
   public final double pi = 3.141593;
   public double erea(); //dien tich
   public double perimeter(); //chu vi
}
```

```
public interface IO {
   public void input();
   public void output();
}
```


8.2. Triển khai Interface trong một lớp

- Lớp có mã nguồn để cụ thể hóa các hành vi của interface gọi là lớp triển khai (implementation).
- Một lớp có thể triển khai nhiều interface.

```
[modifier] class className [extends
  classParent] implements Interface1,
  Interface2, ... {
   //Data
   //Methods
   //Implementation interface1
   //Implementation interface2
   ...
}
```

Thực thi Interface trong một lớp – Ví dụ

```
public class Circle implements Shape, IO{
   int r;
   public double erea(){
      return pi*r*r;
   }
   public double perimeter()    {
      return 2*pi*r;
   }
   public void input(){
      Scanner in = new Scanner(System.in);
      System.out.print("Nhap r = ");
      r = in.nextInt();
   }
```


Thực thi Interface trong một lớp – Ví dụ

```
   public void output(){
      System.out.println("Hinh tron r = " + r);
      System.out.println("Chu vi = " +
        calcuArea());
      System.out.println("Dien tich = " +
        calcuPerimeter());
    }
}
public class InterfaceDemo {
    public static void main(String[] args) {
        Circle = new Circle();
        c.input();
        c.output();
    }
}
```