CHƯƠNG 8: THIẾT KẾ HƯỚNG ĐỐI TƯỢNG

Mục tiêu

- Hiểu về tính kế thừa trong lập trình hướng đối tượng
- Sử dụng được Abstract class và Interface
- Phân biệt override và overload
- Sử dụng đúng các từ khóa: extends, implements, static

.

1. TÍNH KẾ THỪA – INHERITANCE

TÍNH KẾ THỪA – INHERITANCE

- Kế thừa: Lớp con sử dụng được thuộc tính và phương thức của lớp cha
- Kế thừa giúp tránh phải tạo những thứ (thuộc tính, phương thức) đã có sẵn, khi tạo lớp mới.
- Kế thừa: tạo lớp tổng quát trước, và tạo các lớp con kế thừa lớp tổng quát và có thêm các thuộc tính hoặc phương thức riêng.
- Lớp cha (lớp cơ sở) – Lớp con (lớp dẫn xuất) từ khóa extends

Override – ghi đè phương thức

- Lớp con có phương thức cùng tên và cùng tham số với phương thức ở lớp cha. (và không sử dụng phạm vi truy xuất yếu hơn)
- Dùng từ khóa super để truy xuất đến các phương thức và thuộc tính của lớp cha

Overload – nạp chồng phương thức

- Một lớp có nhiều phương thức cùng tên và khác tham số, khác kiểu trả về.
- Trong class có thể overload các Constructor

```
class HinhChuNhat extends HinhHoc{
  double dai,rong;
  public HinhChuNhat(){}
  public HinhChuNhat(double d,double r){
     dai=d;rong=r;
  }
  public HinhChuNhat(double a) {
  dai=rong=a; }
}
```

Từ khóa Static

- Biến static: khi giá trị của biến static thay đổi thì tất cả đối tượng của lớp sẽ được cập nhật.
- Hàm static: trong hàm static các biến được sử dụng phải là static
- Có thể truy xuất biến static mà không cần tạo thực thể của lớp. Biến static được truy xuất thông qua tên lớp.

Abstract class

- Lớp chung chung trừu tượng, cần có các lớp con cụ thể.

(có liên quan tới tính kế thừa)

- Không khởi tạo thực thể của lớp trừu tượng
- Lớp abstract chứa **không** hoặc **nhiều** hàm abstract. Có thể chứa phương thức không abstract.
- Một lớp thừa kế lớp abstract phải hiện thực tất các hàm abstract hoặc lớp đó cũng là abstract

Ví dụ Abstract class

```
public abstract class HinhHoc{


        public String toString(){
                return “Hinh”;
        }
        public abstract int tinhDienTich();
}
```

Interface

- Thể hiện tính đóng gói (encapsulation) trong lập trình hướng đối tượng.
- Một lớp có thể hiện thực nhiều interface

<subclass> extends <superclass> implements <interface1>, <interface2>{}

- Lớp hiện thực interface phải hiện thực tất cả các hàm được đặc tả trong interface đó.

Cấu trúc Interface

```
interface <InterfaceName>{
        <dataType1> <var1>;
        <dataType2> <var2>;
        <returnType1> <methodName1>();
        <returnType2> <methodName2>(<parameter>);
}
```

Biến khai báo có đặc tả ẩn là **public, static** và **final**

Hàm khai báo có đặc tả ẩn là **public** và **abstract**

Thank you

FACULTY OF INFORMATION TECHNOLOGY
Thu Duc College of Technology

Phone: (+848) 22 158 642
Email: fit@tdc.edu.vn
Website: fit.tdc.edu.vn