File Note

Buổi 4: Tính kế thừa

# 1, Định nghĩa, ý nghĩa của tính chất kế thừa

* **Kế thừa** : Là sự thừa hưởng các phương thức, thuộc tính của đối tượng con từ một đối tượng cha mà không cần phải khai báo
* Java không có **đa kế thừa** giữa các class. Mỗi đối tượng chỉ được kế thừa trực tiếp từ một đối tượng khác
* Sử dụng từ khóa **extends** để biểu thị mối quan hệ kế thừa

|  |
| --- |
| **public** class SuViecDaQua extends GhiChu {  // Khai bao thuoc tinh  **String** tenNguoiLienQuan;  **String** ketQuaSuViec;  **String** danhGia;  } |

=> Tác dụng : - Cấu trúc code mạch lạc

- Không cần khai báo nhiều

# 2, Cách xác định các đối tượng có mối quan hệ kế thừa

Nếu đối tượng A và B có cùng bản chất, cùng chung một số đặc điểm và hành vi, thì hình thành nên một đối tượng C gồm các đặc điểm chung của 2 đối tượng A và đối tượng

=> Khi đó: đối tượng A và B sẽ kế thừa đối tượng C

# 3, Cách xây dựng các đối tượng trong mối quan hệ kế thừa:

* Tìm đặc điểm chung giữa các đối tượng
* Hình thành nên các đối tượng cha có đặc điểm và phương thức chung của các đối tượng con
* Tối ưu lại các đối tượng con, loại bỏ các đặc điểm chung, thuộc tính chung của đối tượng con

Ví dụ

|  |  |  |
| --- | --- | --- |
| Đối tượng | Thuộc tính chung | Phương thức chung |
| Học viên | * + Họ tên   + Tuổi   + Giới tính   + Địa chỉ | * + Nhập thông tin   + Ăn uống   + Di chuyển   + In thông tin |
| Nhân viên |

Phương thức và **sự truyền tầng**

Ví dụ:

Ở đối tượng **cha** A có phương thức:

|  |
| --- |
| **void** datBaoThuc (**String** maGhiChu, **int** soLan, **String** thoiGian, **String** amThanh) {  // NDPT  } |

Ở đối tượng **con** B cũng tương ứng có phương thức:

|  |
| --- |
| **void** datBaoThuc(**int** soLan, **String** thoiGian, **String** amThanh) {  // Noi dung phuong thuc  } |

# 4, Lớp trừu tượng

* Một lớp trở thành trừu tượng khi:

+ Bản thân lớp đó chứa các phương thức trừu tượng nên bắt buộc lớp đó phải trở thành trừu tượng

+ Khi đối tượng này không trực tiếp tham gia vào bài toán, nhưng được tạo ra để xây dựng một kiến trúc code tường minh

+ Không có phương thức trừu tượng nhưng không tham gia bài toán => Vẫn là lớp trừu tượng

+ Lớp trừu tượng không trực tiếp sinh ra thực thể

|  |
| --- |
| **public abstract class** GhiChu {  // Khai bao thuoc tinh  **String** maGhiChu;  **String** tieuDe        ;  **String** noiDung;  **String** thoiGian;  **String** diaDiem;  **boolean** suKienBoLo;  } |

+ Phương thức bắt đầu bằng abstract và không có nội dung phương thức

+ Phương thức được viết trong đối tượng cha và **bắt buộc** đối tượng con phải ghi đè

|  |
| --- |
| **abstract void** tenPhuongThuc (giaTri\_tham\_so); |

# 5, Ghi đè phương thức

* Ghi đè chỉ xuất hiện trong kế thừa
* Định nghĩa lại phương thức trừu tượng đã được khai báo trong class cha được gọi là ghi đè
* Có 2 loại ghi đè:

+ Ghi đè hoàn toàn: **Định nghĩa lại hoàn toàn** tất cả phương thức được kế thừa từ class cha

+ Ghi đè không hoàn toàn: Sử dụng **kết quả phương thức** của class cha và định nghĩa nốt phần còn lại

|  |  |
| --- | --- |
| **public abstract class** TuGiac {  // Khai bao thuoc tinh  **int** tinhDienTich();  **void** tinhChuVi() {  // Nội dung phương thức  // Tinh mot canh  }  } | **public class** HinhVuong **externs** TuGiac {  @Override  **int** tinhDienTich();    @Override  **void** tinhChuVi() {  // Nội dung phương thức  **super**.tinhChuVi();  // Tính chu vi tiếp  }  } |