ÔN TẬP LẠI LỚP TRỪU TƯỢNG ABSTRACT TRONG JAVA

* Là một loại lớp mà không thể được khởi tạo trực tiếp.
* Lớp trừu tượng thường chứa các phương thức trừu tượng (abstract method) , đây là các phương thức không có thân hàm (Tức là chỉ có mỗi khai báo thôi ý ).
* Các lớp con kế thừa từ lớp trừu tượng phải triển khai tất cả các phương thức trừu tượng đó để trở thành một lớp cụ thể có thể được sử dụng.
* Đặc điểm của lớp trừu tượng:

+ Không thể khởi tạo đối tượng trực tiếp mà chúng ta cần dùng các lớp con khác kế thừa lại lớp trừu tượng (cũng như triển khai các các cái lớp con của lớp trừu tượng ) để từ đó có thể khởi tạo được đối tượng.

+ Chứa phương thức trừu tượng : Lớp trừu tượng phải chứa ít nhất một hoặc nhiều phương thức trừu tượng . Phương thức trừu tượng là một phương thức mà chỉ khai báo mà không có phần thân hàm. Các lớp con kế thừa phải cung cấp thân hàm cho các phương thức này.

* Chúng ta thường sử dụng lớp trừu tượng với mục đích:

+ Tính chất kế thừa : Lớp trừu tượng cho phép tạo ra một cấu trúc kế thừa giúp tái sử dụng mã nguồn và tạo ra một mô hình chung.

* Ta có thể lấy một ví dụ như này :

// Lớp trừu tượng Shape

abstract class Shape {

// Phương thức trừu tượng để tính diện tích

public abstract double calculateArea();

}

// Lớp con kế thừa từ Shape - Circle

class Circle extends Shape {

private double radius;

public Circle(double radius) {

this.radius = radius;

}

// Triển khai phương thức trừu tượng

@Override

public double calculateArea() {

return Math.PI \* radius \* radius;

}

}

// Lớp con kế thừa từ Shape - Rectangle

class Rectangle extends Shape {

private double length;

private double width;

public Rectangle(double length, double width) {

this.length = length;

this.width = width;

}

// Triển khai phương thức trừu tượng

@Override

public double calculateArea() {

return length \* width;

}

}

// Lớp sử dụng các đối tượng hình học

public class GeometryManager {

public static void main(String[] args) {

// Tạo đối tượng hình tròn

Circle circle = new Circle(5.0);

// Tính diện tích hình tròn

double circleArea = circle.calculateArea();

System.out.println("Diện tích hình tròn: " + circleArea);

// Tạo đối tượng hình vuông

Rectangle rectangle = new Rectangle(4.0, 6.0);

// Tính diện tích hình vuông

double rectangleArea = rectangle.calculateArea();

System.out.println("Diện tích hình vuông: " + rectangleArea);

}

}

+ Trong ví dụ này, Shape là lớp trừu tượng chứa một phương thức trừu tượng “caculateArea() “.Các lớp con Circle và Rectangle kế thừa từ lớp Shape và triển khai phương thức calculateArea() theo cách tính toán của chúng(Tức là mỗi lớp kế thừa sẽ có một cách tính toán riêng ra ).

+ Khi sử dụng hàm main (Của một lớp buồi nào đó), chúng ta có thể tạo ra và sử dụng các đối tượng hình học mà không cần quan tâm đến chi tiết cụ thể của từng loại hình.Điều này giúp tạo ra một mô hình chung cho quản lí hình học và tăng khả năng tái sử dụng mã nguồn.

+ Bắt buộc triển khai: Đôi khi ta muốn định nghĩa một số phương thức mà lớp con phải triển khai theo cách của chúng ( Ví dụ lớp trừu tượng Animal có phương thức trừu tượng Kêu() thì khi lớp Chó hoặc Mèo kế thừa thì cái phương thức kêu của mỗi lớp sẽ riêng ra tức là mỗi con có một kiểu kêu ). Lớp trừu tượng giúp đảm bảo rằng lớp con sẽ cung cấp thân hàm cho các phương thức trừu tượng này.

* Khi một lớp con kế thừa từ một lớp trừu tượng , nó có thể triển khai hoặc không triển khai tất cả các phương thức trừu tượng. Tùy thuộc vào yêu cầu cụ thể của lớp con và thiết kế , lớp con có thể triển khai tất cả , một số hoặc không phải là tất cả các phương thức trừu tượng.

abstract class AbstractClass {

// 3 phương thức trừu tượng

public abstract void method1();

public abstract void method2();

public abstract void method3();

// 2 phương thức thực

public void concreteMethod1() {

// implementation

}

public void concreteMethod2() {

// implementation

}

}

class SubClass extends AbstractClass {

// Triển khai toàn bộ 3 phương thức trừu tượng

@Override

public void method1() {

// implementation

}

@Override

public void method2() {

// implementation

}

@Override

public void method3() {

// implementation

}

// Có thể triển khai hoặc không triển khai các phương thức thực

// Nếu không triển khai, nó sẽ kế thừa từ lớp trừu tượng

}

* Một lưu ý xuyên suốt quá trình làm việc với lớp trừu tượng đó là khi một lớp con kế thừa một lớp trừu tượng thì dù thích triển khai hay không triển khai các lớp trừu tượng hay không thì “ LỚP CON PHẢI KHAI BÁO TẤT CẢ CÁC PHƯƠNG THỨC TRỪU TƯỢNG ĐÓ , DÙ CÓ TRIỂN KHAI HAY KHÔNG “.
* Việc làm này là bắt buộc để đảm bảo rằng lớp con biết về tất cả các phương thức mà nó có thể triển khai trong tương lai.
* Ví dụ nếu lớp trừu tượng “AbstractClass” có phương thức trừu tượng như sau :

abstract class AbstractClass {

public abstract void method1();

public abstract void method2();

public abstract void method3();

}

class SubClass extends AbstractClass {

// Bắt buộc khai báo tất cả các phương thức trừu tượng

@Override

public void method1() {

// implementation

}

@Override

public void method2() {

// implementation

}

@Override

public void method3() {

// implementation

}

// Các phương thức thực có thể triển khai hoặc không

}

* Khai báo này giúp đảm bảo rằng lớp con hiểu rõ về phương thức trừu tượng của lớp cha và có khả năng triển khai chúng nếu cần thiết trong tương lai.

|  |
| --- |
| + Lớp trừu tượng cho phép tạo ra một cấu trúc kế thừa giúp tái sử dụng mã nguồn và tạo ra một mô hình chung  + Đối khi chúng ta một định nghĩa một số phương thức mà lớp con phải triển khai theo cách của chúng .Lớp trừu tượng đảm bảo rằng các lớp con sẽ cung cấp thân hàm  + Khi một lớp con kế thừa một lớp cha thì chúng ta cũng có thể không triển khai hết tất cả các phương thức trừu tượng(Tức là viết phần thân cho các phương thức trừu tượng) nhưng chúng ta phải viết lại tất cả các phương thức trừu tượng(hoặc là định nghĩa hết nó lại ) để nó không bị mất phương thức trừu tượng. |