使用注意事项

1) 修饰符可以用来修饰类中的属性，成员方法以及类

2) 只有默认和public才能修饰类！，并且遵循以上访问权限的特点

3) 成员方法的访问规则和属性完全一样.

P281 - P285 Java_封装

介绍

封装(encapsulation)就是把抽象出的数据[属性]和对数据的操作[方法]封装在一起，数据被保护在内部，程序的其他部分只有通过被授权的操作[方法]，才能对数据进行操作

好处

1) 隐藏实现细节

2) 可以对数据进行验证，保证安全合理.

实现步骤

1) 将属性进行私有化 private [不能直接修改属性]

2) 提供一个公共的(public) set方法，用于对属性判断并赋值.

```
public void setXxx(类型 参数名){ // Xxx表示某个属性
    //加入数据验证的业务逻辑
    属性=参数名;
}
```

3) 提供一个公共的(public)get方法，用于获取属性的值

```
public 数据类型 getXxx(){ //权限判断, Xxx某个属性
    return xx;
}
```

P286 - P Java_继承

提出代码复用性问题.

当多个类存在相同的属性(变量)和方法时，可以从这些类中抽象出父类，在父类定义这些相同的属性和方法，所有的子类不需要重新定义这些属性和方法，只需要通过extends来声明继承父类即可.

[子类(派生类) ; 父类(基类，超类)]

关系是相对的

使用细节

1) 子类继承了所有的属性和方法，但私有的属性和方法不能在子类中直接访问，要通过父类的公共的方法去访问(即间接访问).

2) 子类必须调用父类的构造器，完成父类的初始化 super();

3) 当创建子类对象时，不管使用子类的哪个构造器，默认情况下总会去调用父类的无参构造器，如果父类没有提供无参构造器，则必须在子类的构造器中用super去指定使用父类的哪个构造器完成对父类的初始化工作，否则，编译不会通过.

4) 如果希望指定去调用父类的某个构造器，则显式的调用一下

5) super在使用时，需要放在构造器的第一行

6) super() 和 this() 都只能放在构造器的第一行，因此这两个方法

7) java所有类都是Object的子类，Object类是所有类的基类

8) 父类构造器的调用不限于直接父类，将一直向上追溯到Object类(顶级父类).

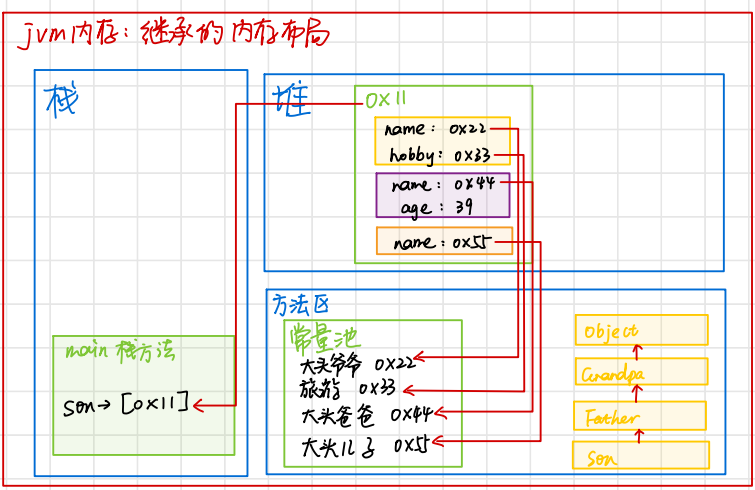
9) 子类最多只能继承一个父类(直接继承)，即java中是单继承机制

如何使A类继承B类和C类？ A继承B，B继承C

10) 不能滥用继承，子类和父类之间必须满足 is-a 逻辑关系

Cat is an Animal ? Yes → Cat extends Animal

```java
public class ExtendTheory {
    public static void main(String[] args) {
        son son = new son();
    }
}
class Grandpa {
    String name = "大头爷爷";
    String hobby = "旅游";
}
class Father extends Grandpa {
    String name = "大头爸爸";
    int age = 39;
}
class son extends Father {
    String name = "大头儿子";
}
```



!!!

1) 若在Father类中将age属性设为prevate，在内存中Father类的age属性仍存在，只是son无法直接访问，但可以增加public方法间接访问.

2) 若访问son.age时，Father类中age为private私有，Grandpa类中也有public age，但系统仍会报错，因为son.age无法向上访问Grandpa中的age，即使Grandpa.age为public.

Exercise

```java
public class ExtendExercise {
    public static void main(String[] args) {
        B b = new B();
    }
}
class A {
    A(){
        System.out.println("a");
    }
    A(String name){
        System.out.println("a name");
    }
}
class B extends A{
    B(){
        this( name: "abc");
        System.out.println("b");
    }
    B(String name){
        super();
        System.out.println("b name");
    }
}
```

输出：1> a
2> b name
3> b

① 默认调用父类无参构造器
② this 与 super 二者不能同时存在.