## 1.复习快速排序

## 2.继承Inheritance的概念

·继承是java面向对象编程技术的一块基石，因为它允许创建分等级层次的类。

·继承就是子类继承父类的特征和行为，使得子类对象（实例）具有父类的实例域和方法，或子类从父类继承方法，使得子类具有父类相同的行为。

·什么 是 什么：Is-a 的关系

·人类，子类可以是教师，学生，男生，女生。通过继承，子类（subclass）自动拥有父类（superclass）的所有成员（成员变量和方法）。Java只能拥有单继承，一个子类只能继承一个父类，一个父类可以拥有多个子类。但是现实情况下会需要多重继承的概念，这时使用接口。

创建一个包，取名为InheritTest

创建一个主类，名称为InheritTest

**public** **class** InheritTest {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Student s1 = **new** Student();

s1.setName("Peter");

s1.setAge(18);

s1.setSchool("JDFZ");

System.***out***.println(s1.getName());

System.***out***.println(s1.getAge());

System.***out***.println(s1.getSchool());

}

创建一个java类，名为Person

**public class** Person{

**private** **int** age;

int height;前面未加访问修饰符，成员变量为default类型

**protected** String name;

public int weight;

**public** **void** setName(String n) {

name = n;

}

**public** **void** setAge(**int** i) {

age = i;

}

**public** String getName() {

**return** **this**.name;

}

**public** **int** getAge() {

**return** **this**.age;

}

}

**class** Student **extends** Person{

**private** String school;

**public** String getSchool() {

**return** school;

}

**public** **void** setSchool(String s) {

school = s;

}

}

1） 内存分析：

子类对象内部有一个父类对象。子类对象比父类对象要大，因为父类有的它都有，它还自己多了东西。

2）访问控制



类中加入成员变量：

**int** height;//前面未加访问修饰符，成员变量为default类型

**public** **int** weight;

在主方法中逐行加入：

System.***out***.println(s1.weight);

System.***out***.println(s1.height);

System.***out***.println(s1.name);

System.***out***.println(s1.age);

在Student类中加入：

**public** **int** changeAge(**int** i) {

**return** age+i;

}

因此，重点注意private。继承会继承此成员变量，但是不能访问。

为什么getAge方法可以？

Public class和default class

在相同src下创建一个新包，取名为InheritTest2

import Person类：

**import** ExtendTest.Person;;

**public** **class** Teacher **extends** Person{

**public** **int** changeHeight() {

**return** height+3;

}

}

因为Person类中的height是default，因此changeHeight方法不能访问。

讲Person 类中的public去掉，发现不能访问。

1. 继承中的方法重写，super关键字以及构造方法

方法重写:

1. 在子类中可以根据需要从父类中继承来的方法进行重写
2. 重写方法必须和被重写方法具有相同方法名称，参数列表和返回类型
3. 重写方法不能使用比被重写方法更严格的访问控制

在Person类中加入：

**public** String getInfo() {

**return** "name:"+""+name+"\n"+"age:"+""+age;

}

在Student类中加入：

**public** String getInfo() {

**return** "name:"+""+getName()+"\n"+"age:"+""+getAge()+"\nschool:"+""+getSchool();

}

Super关键字

使用super来引用父类的成分

在Person类中加入：

**public** **void** setWeight(**int** i) {

weight = i;

}

}

在Student类中加入：

**public** **int** weight;

**public** **void** setWeight(**int** i) {

**super**.setWeight(i);

weight = 100;

System.***out***.println(weight);

System.***out***.println(**super**.weight);

}

在主方法中调用

内存分析:

1. 继承中的构造方法，由于创建子类对象时是也创建了一个父类对象，因此，构造方法也需要必须定义。
2. 子类的构造的过程中必须调用其父类的构造方法
3. 子类可以在自己的构造方法中使用super（）调用父类的构造方法。
4. 使用this（）调用本类的另外的构造方法
5. 如果调用super，必须写在子类构造方法的第一行
6. 如果子类的构造方法中没有显示地调用父类构造方法，则系统默认父类无参数的构造方法。
7. 如果子类构造方法中即没有显示调用父类构造方法，而父类中又没有无参的构造方法，则编译出错。（在eclipse中出错，直接编译是可以的，因为直接编译时会默认有一个构造方法，在eclipse中不默认）

**public** Person() {

name = "Frank";

}

**public** Person(**int** i,**int** h) {

age = i;

height = h;

}

**public** Student(String s) {

school = s;

}

**public** Student(**int** w, **int** h,String s) {

**super**(w,h);

name = s;

}