# 实验2 Java类与对象（一）

**一 、实验目的与要求**

1、掌握类的封装与对象的使用。

2、掌握构造方法的使用，Java多态性的应用，如本章的方法的重载。

3、掌握方法getXXX()和setXXX()的用法。

4、**请将代码和运行结果粘贴题目下面，以QQ在线作业word版形式提交。**

**5、提交时间截止10.16日之前。**

**二、 实验内容**

**1、教材第124页编程题（2）四个类：PC、CPU、HardDisk和Test**

PC:

package Page124;

public class PC {

CPU cpu;

HardDisk HD;

public void setCPU(CPU c) {

this.cpu = c;

}

public void setHardDisk(HardDisk h) {

HD = h;

}

public void show() {

System.out.println("cpu: " +cpu.getSpeed());

System.out.println("Hd: "+HD.getAmount());

}

}

CPU:

package Page124;

public class CPU {

//int型成员变量

private int speed;

//getSpeed()返回speed值

public int getSpeed() {

return speed;

}

//setSpeed(int m)设置speed值

public void setSpeed(int m) {

this.speed = m;

}

}

HardDisk:

package Page124;

public class HardDisk {

int amount;

public int getAmount() {

return amount;

}

public void setAmount(int m) {

this.amount = m;

}

}

Test:

package Page124;

public class Test {

public static void main(String[] args) {

CPU cpu=new CPU();

cpu.setSpeed(2000);

HardDisk disk=new HardDisk();

disk.setAmount(200);

PC pc=new PC();

pc.setCPU(cpu);

pc.setHardDisk(disk);

pc.show();

}

}

结果：

cpu: 2000

Hd: 200

进程已结束，退出代码为 0

2、Tank战斗场景：根据下面表中对类的描述写出类的源码。

|  |  |  |  |
| --- | --- | --- | --- |
| 源文件名称：Tank.java | | | |
| 类名：Tank (坦克) | | | |
| 属性 | 变量名 | 含义 | 数据类型 |
| speed | 速度 | double |
| bulletAmount | 炮弹数量 | int |
| color | 坦克颜色  （所有坦克的颜色都一致，若一个坦克颜色改变则都改变） | char |
| 方法 | 方法名 | 含义 | 返回值含义 |
| speedUp(int x) | 加速到x公里/小时（最高时速不能超过200，若超过，给出提示，并自动降为200） | void |
| speedDown(int x) | 减速到x公里/小时 | void |
| setBulletAmount(int x) | 设置炮弹数量 | void |
| getSpeed() | 获取坦克当前的速度 | 当前速度 |
| getBulletAmount() | 获取炮弹当前的数量 | 当前炮弹数量 |
| fire() | 开火（每开火一次，发射一枚炮弹） | void |
| conFire(int x) | 连续开火（连续发射，共计x枚炮弹，注意连续开火也是一枚枚发射出去的，而不是一次发射x枚） | void |
| setColor(char x) | 将坦克颜色设置为x色 | void |
| brake() | 刹车（速度降为0） | void |

|  |  |
| --- | --- |
| 源文件名称：Fight.java | |
|  | |
| 类名：Fight (主类，即Java应用程序) | |
| main方法完成任务 | 1.实例化两个坦克对象 |
| 2.实现加速、设置炮弹数量、设置所有坦克颜色为绿色功能 |
| 3.开火 |
| 4.减速 |
| 5.设置坦克颜色为黄色（与2中不同，用另一种方法实现对颜色属性的改变） |
| 6.连续开火 |
| 每次执行一次动作后，都需要用System.out.println输出坦克类的相关属性值以验证是否被改变。开火后，当炮弹数量为0时，坦克就立即刹车。 |

Tank:

package TankTest;

public class Tank {

double speed;

int bulletAmount;

char color;

//加速到x公里/小时（最高时速不能超过200，若超过，给出提示，并自动降为200）

public void speedUp(int x){

this.speed=x;

speed++;

if (x>200) {

System.out.println("最高时速不能超过200公里/小时！");

speed--;

}

System.out.println("当前坦克加速到："+x+"公里/小时");

};

//减速到x公里/小时

public void speedDown(int x){

this.speed=x;

speed--;

};

//设置炮弹数量

public void setBulletAmount(int bulletAmount) {

this.bulletAmount = bulletAmount;

}

//获取坦克当前的速度

public double getSpeed() {

return speed;

}

//获取炮弹当前的数量

public int getBulletAmount() {

return bulletAmount;

}

//开火（每开火一次，发射一枚炮弹）

public void fire(){

bulletAmount--;

}

//连续开火（连续发射，共计x枚炮弹，注意连续开火也是一枚枚发射出去的，而不是一次发射x枚）

public void conFire(int x){

this.bulletAmount=x;

if (x>0){

x--;

}

}

//将坦克颜色设置为x色

public void setColor(char x) {

this.color = x;

}

//刹车（速度降为0）

public void brake(){

speed=0;

}

}

Fight:

package TankTest;

public class Fight {

public static void main(String[] args) {

//1.实例化两个坦克对象

Tank t1,t2;

t1=new Tank();

t2=new Tank();

//2.实现加速、设置炮弹数量、设置所有坦克颜色为绿色功能

t1.speedUp(250);

t2.speedUp(178);

t1.setBulletAmount(10);

t2.setBulletAmount(12);

t1.setColor('绿');

t2.setColor('绿');

System.out.println("t1坦克的颜色："+t1.color);

System.out.println("t2坦克的颜色："+t2.color);

System.out.println("t1坦克当前的炮弹数量："+t1.bulletAmount+"发");

System.out.println("t2坦克当前的炮弹数量："+t2.bulletAmount+"发");

//3.开火

t1.fire();

t2.fire();

//4.减速

t1.speedDown(150);

t2.speedDown(120);

//5.设置坦克黄色（用另一种方法实现对颜色属性的改变）

t1.setColor('黄');

t2.setColor('黄');

//6.连续开火,

// 每次执行一次动作后，都需要用System.out.println输出坦克类的相关属性值以验证是否被改变。

// 开火后，当炮弹数量为0时，坦克就立即刹车。

t1.conFire(8);

t2.conFire(9);

if(t1.bulletAmount>0){

t1.brake();

}

if(t2.bulletAmount>0){

t2.brake();

}

System.out.println("t1坦克目前的速度："+t1.speed+"公里/小时");

System.out.println("t2坦克目前的速度："+t2.speed+"公里/小时");

System.out.println("t1坦克的颜色："+t1.color);

System.out.println("t2坦克的颜色："+t2.color);

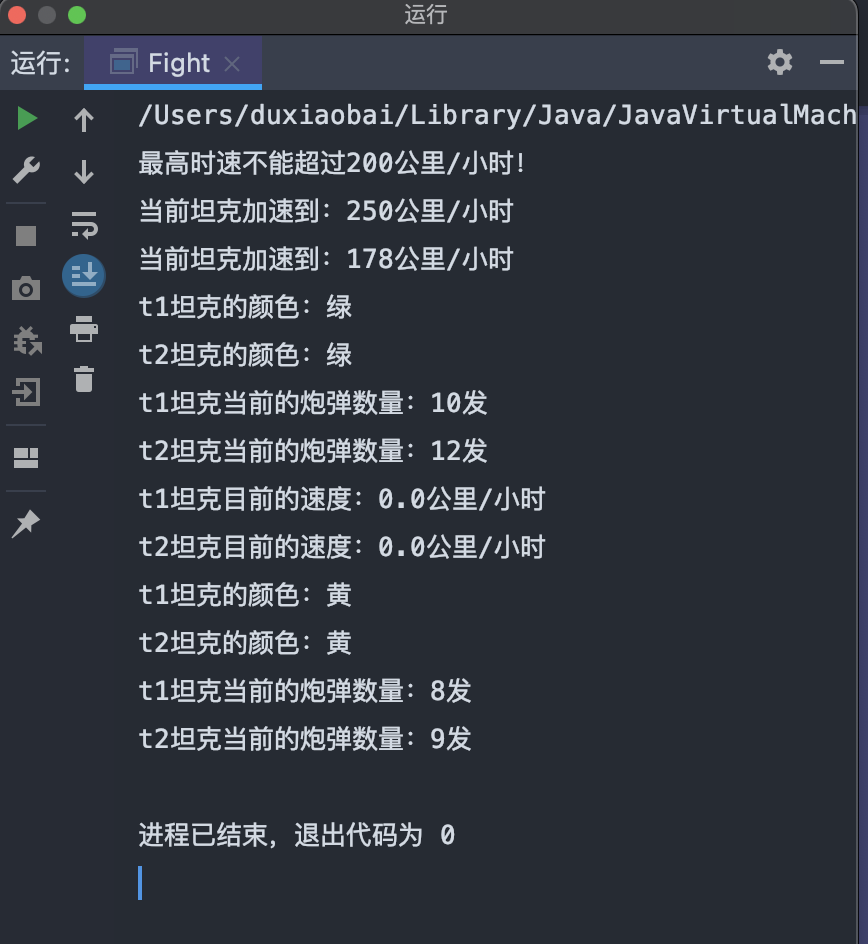
System.out.println("t1坦克当前的炮弹数量："+t1.bulletAmount+"发");

System.out.println("t2坦克当前的炮弹数量："+t2.bulletAmount+"发");

}

}

结果：



3、this的用法

class Student {

public Student() {

System.out.println("无参的构造方法");

}

public Student(String name) {

this();

System.out.println("一个参数的构造方法");

}

public Student(String name, int age) {

this(name);

System.out.println("两个参数的构造方法");

}

}

* 定义Example04类，代码如下所示：

public class Example04 {

public static void main(String[] args) {

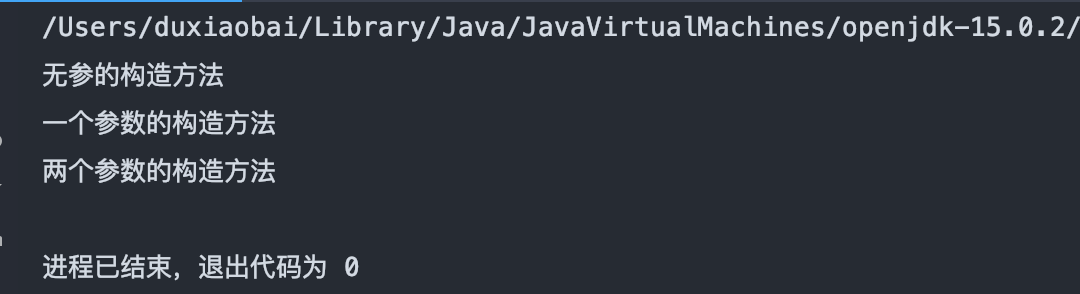
Student s1 = new Student("Jack", 22);

}

}

运行结果是什么？为什么是这样的结果？

调用本类的其他构造。不管是构造函数还是一般函数，因为都要使用对象，所以只要通过对象调用，那么在这个函数中都有一个this引用变量存在，记录当前调用这个函数的对象堆内存地址值。



4、家族姓氏（static的用法）

|  |  |  |  |
| --- | --- | --- | --- |
| 源文件名称：FamilyPerson.java | | | |
| 类名：FamilyPerson | | | |
| 属性 | 变量名 | 含义 | 数据类型 |
| surname | 姓 | String |
| name | 名 | String |
| 方法 | 方法名 | 含义 | 返回值类型含义 |
| setSurname | 设置姓 | void |
| setName | 设置名 | void |

|  |  |
| --- | --- |
| 源文件名称：MainClass.java | |
| 类名：MainClass (主类，即应用程序) | |
| main方法 | 完成任务 |
| 1.实例化三个对象，一个父亲father，一个大儿子sonone，一个二儿子sontwo |
| 2.调用setSurname为姓赋值（例如：张、李、王） |
| 3.调用setName为三个对象的名赋值 |
| 4.显示三个对象的姓名 |
| 5. 调用setSurname给父亲、大儿子、二儿子改姓 |
| 6.再次显示三个对象的姓名 |
| 备注 | 第2步和第5步的代码不完全相同，思考一下。 |

Family：

package Family;

public class FamilyPerson {

private String surname;

private String name;

public void setSurname(String surname) {

this.surname=surname;

}

public void setName(String name) {

this.name=name;

}

//

@Override

public String toString() {

return surname + name;

}

}

MainClass:

package Family;

public class MainClass {

static String surname;

public static void main(String[] args) {

//1.实例化三个对象，一个父亲father，一个大儿子sonone，一个二儿子sontwo

FamilyPerson father=new FamilyPerson();

FamilyPerson sonone=new FamilyPerson();

FamilyPerson sontwo=new FamilyPerson();

//2.调用setSurname为姓赋值（例如：张、李、王）

father.setSurname("张");

sonone.setSurname("张");

sontwo.setSurname("张");

//3.调用setName为三个对象的名赋值

father.setName("国庆");

sonone.setName("建国");

sontwo.setName("建业");

//4.显示三个对象的姓名

System.out.println(father);

System.out.println(sonone);

System.out.println(sontwo);

//5. 调用setSurname给父亲、大儿子、二儿子改姓

surname="赵";

father.setSurname(surname);

sonone.setSurname(surname);

sontwo.setSurname(surname);

//6.再次显示三个对象的姓名

System.out.println(father);

System.out.println(sonone);

System.out.println(sontwo);

}

}



**三、实验内容中遇到的问题、解决方法和体会**