

**计算机与信息 学院实验报告**

|  |  |  |
| --- | --- | --- |
| 实验课程： | 面向对象程序设计实验 | |
| 实验编号： | 实验六 | |
| 实验名称： | 类的多态性 | |
| 实验人员： | 学号 | 22111302041 |
| 姓名 | 高蝶 |
| 班级 | 22级计算机科学与技术1班 |
| 指导教师： | 陶涛 | |
| 实验室： | 学院楼2号楼202 | |
| 实验日期： | 2023.3.28 | |

# 实验目的

1. 掌握子类中方法重写，并利用方法重写实现对象的多态性；

2. 掌握super关键字，利用super解决子类重写父类成员的问题；

3. 理解通过继承实现面向对象的多态性。

# 实验要求

1. 所编写的程序应符合类的封装、继承要求，符合面向对象程序编写规范；

2. 能利用继承性、多态性进行复杂程序的设计和实现；

3. 对问题进行充分的需求分析，设计正确、全面的测试数据，设计类的定义框架及UML图后，再进行编程实现；

4. 所设计的测试数据全部通过后，再进行程序提交，提高程序的通过率。

# 实验内容

**1.验证型实验**

仔细读下面的JAVA语言源程序，自己给出程序的运行结果，再与上机运行程序所得的结果作比较，如果有错，务必弄清楚出错原因。

class Animal {

int age;

String name;

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

// 动物类里面有叫和吃两个方法

public void cry() {

System.out.println("我不知道叫什么");

}

public void eat() {

System.out.println("我不知道吃什么");

}

}

// 狗类继承于动物类

class Dog extends Animal {

// 覆盖（重写）方法

public void cry() {

System.out.println("旺旺");

}

public void eat() {

System.out.println("我是狗，我爱吃骨头");

}

}

// 猫类继承于动物类

class Cat extends Animal {

// 覆盖（重写）方法

public void cry() {

System.out.println("喵喵");

}

public void eat() {

System.out.println("我是猫，我爱吃鱼");

}

}

// 食物类

class Food {

String name;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

// 食物类里面让它有一个方法

public void showName() {

}

}

// 鱼（食物的一种）继承于食物

class Fish extends Food {

public void showName() {

System.out.println("食物：鱼");

}

}

// 骨头（食物的一种）继承于食物

class Bone extends Food {

public void showName() {

System.out.println("食物：骨头");

}

}

// 主人类 存在一种喂食方法

class Master {

// 给动物喂食物，如果没有多态，他要写给猫喂食和给狗喂食两个方法

// 有了多态，以后即使再来好多动物，用这一个函数就可以了

public void feed(Animal an, Food f) {

an.eat();

f.showName();

}

}

public class DuoTaiDemo {

public static void main(String args[]) {

Master master = new Master();

master.feed(new Dog(), new Bone());

master.feed(new Cat(), new Fish());

}

}

**2.应用型实验**

**2.1成绩计算**

编写两个系别的学生类：英语系，计算机系（要求通过继承学生类）

各系有以下成绩：

（1）英语系：演讲，期中考试，期末考试；

（2）计算机系：操作能力，英语写作，期中考试，期末考试。

各系总分评测标准：

（1）英语系：演讲50%，期末考试25%，期中考试25%；

（2）计算机系：操作能力40%，英语写作20%，期末考试20%，期中考试20%。

定义一个可容纳100个学生的学生类数组，输入n个英语系学生的信息与m个计算机系学生的信息，然后按年龄大小升序输出数组中的学生信息（学号、姓名、性别、年龄、系别、各分项成绩与综合成绩），若年龄相同则按学号升序输出，成绩保留1位小数。

输入：整数n与n个英语系学生信息（学号、姓名、性别、年龄、演讲成绩、期中考试成绩与期末考试成绩），整数m与m个计算机系学生信息（学号、姓名、性别、年龄、操作成绩、英语写作成绩、期中考试成绩与期末考试成绩）

输出：按年龄大小升序输出数组中的学生信息（学号、姓名、性别、年龄、系别、各分项成绩与综合成绩），若年龄相同则按学号升序输出，成绩保留1位小数。

样例输入：

8

1007 calro male 23 85 84 97

2005 black male 19 74 90 69

1002 alice female 24 75 73 91

2002 ally female 18 41 74 92

1001 adam male 24 68 78 40

1008 hali male 24 70 94 81

1010 fugui female 25 44 78 61

2007 fulin male 19 77 92 68

9

2009 calina female 18 44 75 75 71

1007 fulin male 22 70 53 82 56

1005 dam male 21 56 43 72 93

1001 flank male 22 91 64 77 74

2002 bob male 24 41 76 40 42

2007 eve male 20 47 74 79 70

2010 adam male 17 43 60 57 55

2008 black male 17 77 88 76 75

1002 geo female 24 89 68 78 60

样例输出：

2008,black,male,17,Computer,77.0,88.0,76.0,75.0,78.6

2010,adam,male,17,Computer,43.0,60.0,57.0,55.0,51.6

2002,ally,female,18,English,41.0,74.0,92.0,62.0

2009,calina,female,18,Computer,44.0,75.0,75.0,71.0,61.8

2005,black,male,19,English,74.0,90.0,69.0,76.8

2007,fulin,male,19,English,77.0,92.0,68.0,78.5

2007,eve,male,20,Computer,47.0,74.0,79.0,70.0,63.4

1005,dam,male,21,Computer,56.0,43.0,72.0,93.0,64.0

1001,flank,male,22,Computer,91.0,64.0,77.0,74.0,79.4

1007,fulin,male,22,Computer,70.0,53.0,82.0,56.0,66.2

1007,calro,male,23,English,85.0,84.0,97.0,87.8

1001,adam,male,24,English,68.0,78.0,40.0,63.5

1002,alice,female,24,English,75.0,73.0,91.0,78.5

1002,geo,female,24,Computer,89.0,68.0,78.0,60.0,76.8

1008,hali,male,24,English,70.0,94.0,81.0,78.8

2002,bob,male,24,Computer,41.0,76.0,40.0,42.0,48.0

1010,fugui,female,25,English,44.0,78.0,61.0,56.8

**2.2 优秀学生遴选**

学校准备对优秀学生进行表彰，优秀学生包括本科生与研究生。本科生评优的要求是必须过CET4,学年综合测评成绩大于85分；研究生评优的标准是过CET6,已发表文章篇数大于等于1，学年综合测评成绩大于85分。要求输出符合要求的学生信息（本科生信息：学号，姓名，是否过CET4，学年综合测评成绩；研究生信息：学号，姓名，是否过CET6，发表文章篇数，学年综合测评成绩）。

在主类中实例化n个本科生与m个硕士生（n+m<=100），输出优秀本科生信息，再输出优秀研究生信息，不同类别学生按综合测评成绩降序排序，若成绩相同则按学号升序排序。

要求用多态来设计判断学生是否优秀。

输入：整数t（t为n与m的和），n个本科生信息，m个研究生信息，注意这些信息交错输入，输入信息中首位为1代表本科生，2代表研究生。

输出：输出优秀本科生信息，再输出优秀研究生信息，不同类别学生按综合测评成绩降序排序，若成绩相同则按学号升序排序。成绩保留3位小数。若相应类别优秀生不存在，则任何信息都不输出。

样例输入：

14

2 g2008 black false 0 71.9414

2 g2010 eve true 3 73.6821

1 u1008 calina true 80.8524

1 u1006 geo false 80.3068

1 u2001 ada true 94.4016

1 u2004 elin true 84.2804

1 u1003 dam false 71.412

1 u2002 george true 64.2655

2 g1005 adam false 2 72.7523

2 g2006 fugui true 1 96.8847

1 u1007 hali false 67.7762

2 g1010 alice true 0 79.7267

2 g1009 calro true 2 69.4295

2 g2005 fulin true 3 84.0751

样例输出：

excellent undergraduate student:

u2001,ada,true,94.402

excellent graduate student:

g2006,fugui,true,1,96.885

**3. 设计型实验**

**3.1 排队进行健康体检**

队列是一种特殊的线性表，特殊之处在于它只允许在表的前端（front）进行删除操作，而在表的后端（rear）进行插入操作。进行插入操作的端称为队尾，进行删除操作的端称为队头。队列中没有元素时，称为空队列。队列的数据元素又称为队列元素。在队列中插入一个队列元素称为入队，从队列中删除一个队列元素称为出队。因为队列只允许在一端插入，在另一端删除，所以只有最早进入队列的元素才能最先从队列中删除，故队列又称为先进先出（FIFO—first in first out）线性表。

顺序队列：建立顺序队列结构必须为其静态分配或动态申请一片连续的存储空间，并设置两个指针进行管理。一个是队头指针front，它指向队头元素；另一个是队尾指针rear，它指向下一个入队元素的存储位置。在入队、出队过程中，可能需要频繁移动数据。

循环队列：在实际使用队列时，为了使队列空间能重复使用，减少数据移动次数，对队列的使用方法稍加改进：无论插入或删除，一旦rear指针增1或front指针增1时超出了所分配的队列空间，就让它指向这片连续空间的起始位置（可以使用取余运算rear%MaxSize和front%MaxSize实现，MaxSize表示数组容量）。这实际上把队列空间想象成一个环形空间，环形空间中的存储单元循环使用，用这种方法管理的队列称为循环队列。

使用面向对象的思想实现循环队列，队列中的元素存放在数组中。其主要功能为：

（1）数据元素入队

（2）数据元素出队

（3）队列是否为空

（4）队列是否为满

（5）查询数据元素是否在队列中

（6）返回队列中元素个数

某学校安排学生与教师做健康体检活动，学生的属性包括学号、姓名等，方法包括带参构造方法，返回学生信息字符串的方法（返回学号与姓名，之间用英文逗号分隔）。教师的属性包括职工号、姓名、家庭住址等，方法包括带参构造方法，返回教师信息字符串的方法（返回职工号、姓名与家庭住址，之间用英文逗号分隔）。这两个类在设计时要求用继承的思想进行设计。

在主类中，定义队列对象存储排队中的学生与教师信息。输入一个整数表示队列能容纳的最大人数。按输入选项完成相应操作：0-结束程序，1-排队等待体检，2-体检完成出队，3-按照姓名查找某个人，4-输出队列中人员数。选项为1时（排队），输入a表示学生，b表示教师，并输入相应的完整信息，若队列未满，则该人入队，否则输出“queue is full,operation failed”；选项为2时，若队列未空，则应输出出队的人员信息，否则输出“queue is empty,operation failed”；选项为3时，输入姓名，输出队列中相应的人员信息，一个人员信息独立成行，若查询不到信息则输出“no found”。

提示：在使用Java语言编写实现循环队列时，一般使用两种方式判断队列是否已满，（1）增加一个属性size用来记录目前的元素个数。当front=rear时，size=0表示队列为空，size=数组长度表示队列已满。（2）数组中只存储“数组大小-1”个元素，保证rear转一圈之后不会和front相等，也就是队列满的时候，rear+1=front，中间刚好空一个元素。

输入：队列最大容量，选项及其对应的输入信息

输出：相应信息

样例输入：

3

1

a x001 aa

1

b j001 aa 15-1-101

1

a x002 bb

4

3

aa

1

b j002 cc 12-2-201

3

cc

2

0

样例输出：

3

x001,aa

j001,aa,15-1-101

queue is full,operation failed

no found

x001,aa

# 程序清单

2.应用型实验

（1）

测试数据

第1组：8

1011 calro male 23 85 84 97

2005 black male 19 74 90 69

1002 alice female 24 75 73 91

2002 ally female 18 41 74 92

1000 adam male 24 68 78 40

1008 hali male 24 70 94 81

1010 fugui female 25 44 78 61

2007 fulin male 19 77 92 68

9

2009 calina female 18 44 75 75 71

1007 fulin male 22 70 53 82 56

1005 dam male 21 56 43 72 93

1001 flank male 22 91 64 77 74

2011 bob male 24 41 76 40 42

2006 eve male 20 47 74 79 70

2010 adam male 17 43 60 57 55

2008 black male 17 77 88 76 75

1003 geo female 24 89 68 78 60

第2组：2

1002 alice female 24 75 73 91

2002 ally female 18 41 74 92

2

2009 calina female 18 44 75 75 71

1007 fulin male 22 70 53 82 56

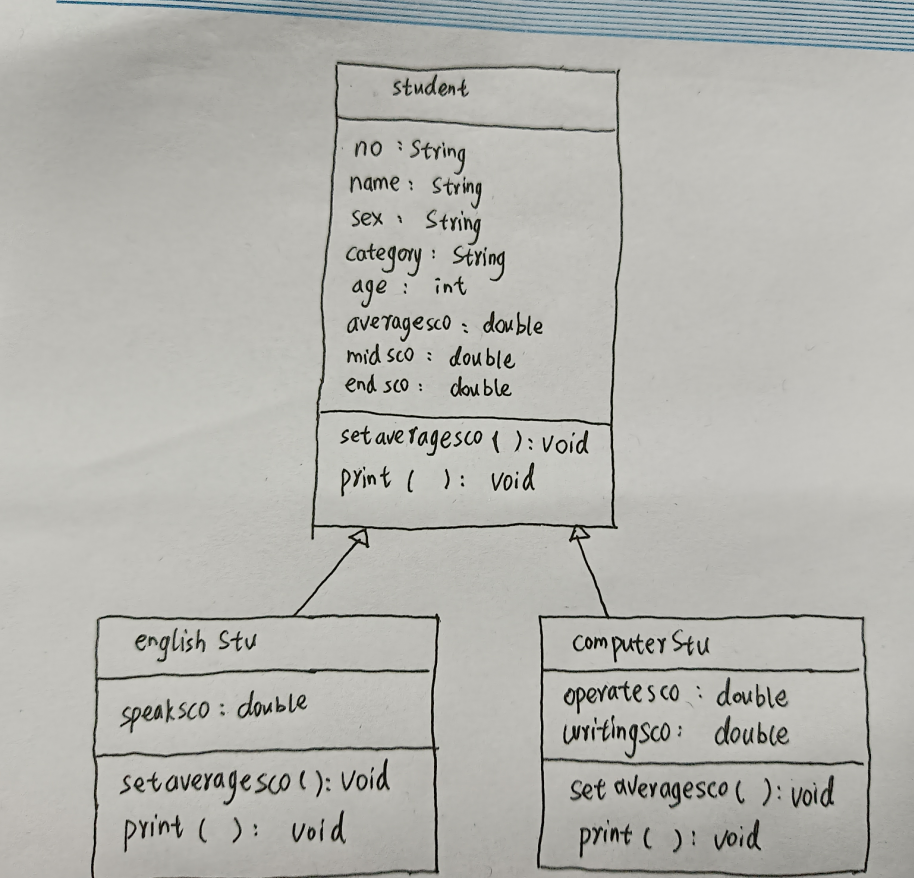
第3组：1

2005 black male 19 74 90 69

2

2009 calina female 18 44 75 75 71

1007 fulin male 22 70 53 82 56

类的UML图：

程序代码：**package** prj\_scorejisuan;

**import** java.util.Scanner;

**public** **class** Main {

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

Scanner reader=**new** Scanner(System.***in***);

**int** n,m;

String no,name,sex,category;

**int** age;

**double** midsco,endsco,speaksco,operatesco,writingsco;

Student[] stu= **new** Student[105];

n=reader.nextInt();

**for**(**int** i=0;i<n;i++) {

no=reader.next();

name=reader.next();

sex=reader.next();

age=reader.nextInt();

speaksco=reader.nextDouble();

midsco=reader.nextDouble();

endsco=reader.nextDouble();

stu[i]=**new** englishStu(no,name,sex,age,midsco,endsco,speaksco);

stu[i].setaveragesco();

}

m=reader.nextInt();

**for**(**int** i=n;i<n+m;i++) {

no=reader.next();

name=reader.next();

sex=reader.next();

age=reader.nextInt();

operatesco=reader.nextDouble();

writingsco=reader.nextDouble();

midsco=reader.nextDouble();

endsco=reader.nextDouble();

stu[i]=**new** computerStu(no,name,sex,age,midsco,endsco,operatesco,writingsco);

stu[i].setaveragesco();

}

Main.*sort*(stu,n+m);

}

**static** **void** sort(Student[] stu,**int** len) {

Student t=**new** Student();

**int** i,j;

**for**( i=1;i<=len-1;i++) {

**for**(j=1;j<=len-i;j++) {

**if**(stu[j-1].age>stu[j].age||stu[j-1].age==stu[j].age&&stu[j-1].no.compareTo(stu[j].no)>0) {

t=stu[j-1];

stu[j-1]=stu[j];

stu[j]=t;

}

}

}

**for**(i=0;i<len;i++) {

stu[i].print();

}

}

}

**class** Student{

String no,name,sex,category;

**int** age;

**double** averagesco,midsco,endsco;;

Student(){}

Student(String no,String name,String sex,**int** age,**double** midsco,**double** endsco){

**this**.no=no;

**this**.name=name;

**this**.sex=sex;

**this**.age=age;

**this**.midsco=midsco;

**this**.endsco=endsco;

}

**void** setaveragesco() {}

**void** print() {}

}

**class** englishStu **extends** Student{

**double** speaksco;

englishStu(String no,String name,String sex,**int** age,**double** midscore,**double** endsco,**double** speaksco){

**super**(no,name,sex,age,midscore,endsco);

**this**.speaksco=speaksco;

}

**void** setaveragesco() {

averagesco=speaksco\*0.5+midsco\*0.25+endsco\*0.25;

}

**void** print() {

System.***out***.printf("%s,%s,%s,%d,%s,%.1f,%.1f,%.1f,%.1f\n",no,name,sex,age,"English",speaksco,midsco,endsco,averagesco);

}

}

**class** computerStu **extends** Student{

**double** operatesco,writingsco;

computerStu(String no,String name,String sex,**int** age,**double** midscore,**double** endsco,**double** operatesco,**double** writingsco){

**super**(no,name,sex,age,midscore,endsco);

**this**.operatesco=operatesco;

**this**.writingsco=writingsco;

}

**void** setaveragesco() {

averagesco=operatesco\*0.4+writingsco\*0.2+midsco\*0.2+endsco\*0.2;

}

**void** print() {

System.***out***.printf("%s,%s,%s,%d,%s,%.1f,%.1f,%.1f,%.1f,%.1f\n",no,name,sex,age,"Computer",operatesco,writingsco,midsco,endsco,averagesco);

}

}

（2）

测试数据

第1组：14

2 g2008 black false 0 71.9414

2 g2010 eve true 3 73.6821

1 u1008 calina true 80.8524

1 u1006 geo false 80.3068

1 u2001 ada true 94.4016

1 u2004 elin true 84.2804

1 u1003 dam false 71.412

1 u2002 george true 64.2655

2 g1005 adam false 2 72.7523

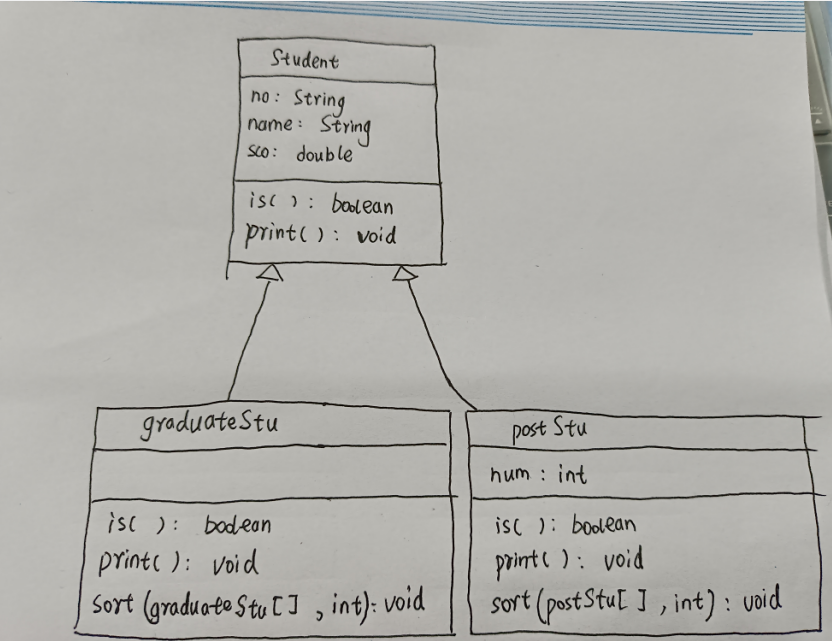
2 g2006 fugui true 1 96.8847

1 u1007 hali false 67.7762

2 g1010 alice true 0 79.7267

2 g1009 calro true 2 69.4295

2 g2005 fulin true 3 84.0751

类的UML图：

程序代码：**package** prj\_goodstudentsel;

**import** java.util.Scanner;

**public** **class** Main {

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

Scanner reader=**new** Scanner(System.***in***);

**int** t,x,num,i,j=0,k=0;

**double** sco;

String no,name;

**boolean** cet;

Student p;

graduateStu []grastu=**new** graduateStu[100];

postStu [] poststu=**new** postStu[100];

t=reader.nextInt();

**for**(i=0;i<t;i++) {

x=reader.nextInt();

no=reader.next();

name=reader.next();

cet=reader.nextBoolean();

**if**(x==1) {

sco=reader.nextDouble();

p=**new** graduateStu(no,name,cet,sco);

**if**(p.is()==**true**) {

grastu[j]=**new** graduateStu (no,name,cet,sco);

j++;

}

}**else** {

num=reader.nextInt();

sco=reader.nextDouble();

p=**new** postStu(no,name,cet,sco,num);

**if**(p.is()==**true**) {

poststu[k]=**new** postStu (no,name,cet,sco,num);

k++;

}

}

}

//System.out.println(j+" "+k);

graduateStu.*sort*(grastu,j);

postStu.*sort*(poststu,k);

}

}

**class** Student{

String no,name;

**boolean** cet;

**double** sco;

Student(){}

Student(String no,String name,**boolean** cet,**double** sco){

**this**.no=no;

**this**.name=name;

**this**.cet=cet;

**this**.sco=sco;

}

**boolean** is() {**return** **true**;}

**void** print() {}

}

**class** graduateStu **extends** Student{

graduateStu(){}

graduateStu(String no,String name,**boolean** cet,**double** sco){

**super**(no,name,cet,sco);

}

**boolean** is() {

**if**(cet==**true**&&sco>85) **return** **true**;

**else** **return** **false**;

}

**void** print() {

System.***out***.printf("%s,%s,%b,%.3f\n",no,name,cet,sco);

}

**static** **void** sort(graduateStu[] grastu,**int** len) {

System.***out***.println("excellent undergraduate student:");

**int** i,j;

graduateStu t=**new** graduateStu();

**for**( i=1;i<=len-1;i++) {

**for**(j=1;j<=len-i;j++) {

**if**(grastu[j-1].sco<grastu[j].sco||grastu[j-1].sco==grastu[j].sco&&grastu[j-1].no.compareTo(grastu[j].no)>0) {

t=grastu[j-1];

grastu[j-1]=grastu[j];

grastu[j]=t;

}

}

}

**for**(i=0;i<len;i++) {

grastu[i].print();

}

}

}

**class** postStu **extends** Student{

**int** num;

postStu(){}

postStu(String no,String name,**boolean** cet,**double** sco,**int** num){

**super**(no,name,cet,sco);

**this**.num=num;

}

**boolean** is() {

**if**(cet==**true**&&sco>85&&num>=1) **return** **true**;

**else** **return** **false**;

}

**void** print() {

System.***out***.printf("%s,%s,%b,%d,%.3f\n",no,name,cet,num,sco);

}

**static** **void** sort(postStu[] stu,**int** len) {

System.***out***.println("excellent graduate student:");

**int** i,j;

postStu t=**new** postStu();

**for**( i=1;i<=len-1;i++) {

**for**(j=1;j<=len-i;j++) {

**if**(stu[j-1].sco<stu[j].sco||stu[j-1].sco==stu[j].sco&&stu[j-1].no.compareTo(stu[j].no)>0) {

t=stu[j-1];

stu[j-1]=stu[j];

stu[j]=t;

}

}

}

**for**(i=0;i<len;i++) {

stu[i].print();

}

}

}

3.设计型实验

（1）

测试数据

第1组：3

1

a x001 aa

1

b j001 aa 15-1-101

1

a x002 bb

4

3

aa

1

b j002 cc 12-2-201

3

cc

2

0

第2组：3

1

a x001 aa

1

b j001 aa 15-1-101

3

cc

2

0

第3组：

2

1

a x001 aa

1

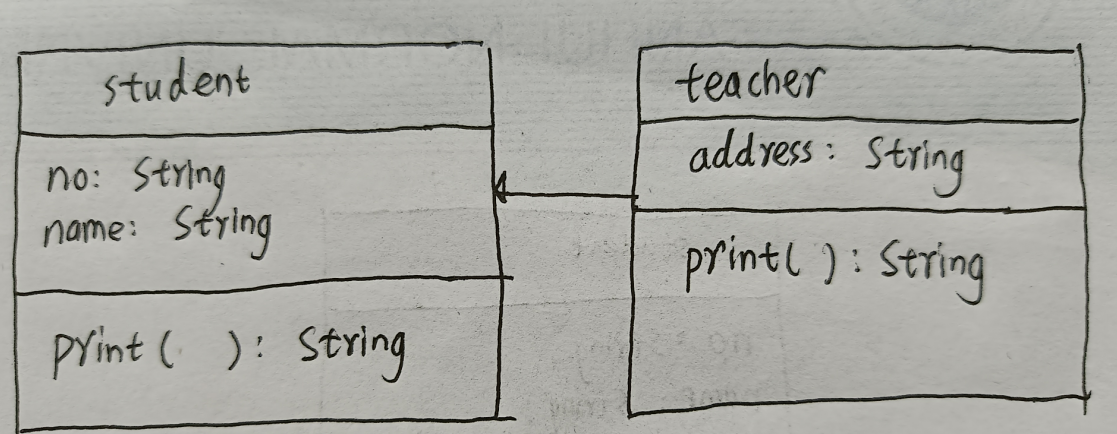
b j001 aa 15-1-101

4

3

aa

0

类的UML图：

程序代码：**package** prj\_healthexam;

**import** java.util.Scanner;

**public** **class** Main {

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

Scanner reader=**new** Scanner(System.***in***);

**int** n,x,front=0,rear=0,i=0,flag=0;

**char** c;

String name,no,address,nname;

n=reader.nextInt();

x=reader.nextInt();

student[] stu=**new** student[300];

**while**(x!=0) {

**if**(x==1) {

c=reader.next().charAt(0);

no=reader.next();

name=reader.next();

**if**(c=='b') {

address=reader.next();

**if**(rear-front<n) {stu[rear]=**new** teacher(no,name,address);rear++;}

**else** System.***out***.println("queue is full,operation failed");

}**else** {

**if**(rear-front<n) {stu[rear]=**new** student(no,name);rear++;}

**else** System.***out***.println("queue is full,operation failed");

}

}**else** **if**(x==2) {

**if**(rear>front) {

System.***out***.println(stu[front].print());

front++;

}**else** {

System.***out***.println("queue is empty,operation failed");

front=0;

rear=0;

}

}**else** **if**(x==3) {

nname=reader.next();

**for**(i=front;i<rear;i++) {

**if**(stu[i].name.equals(nname)) {

System.***out***.println(stu[i].print());

flag=1;

}

}

**if**(flag==0) System.***out***.println("no found");

flag=0;

}**else** **if**(x==4) {

System.***out***.println(rear-front);

}

x=reader.nextInt();

}

}

}

**class** student{

String no,name;

student(){}

student(String no,String name){

**this**.no=no;

**this**.name=name;

}

String print() {

**return** no+","+name;

}

}

**class** teacher **extends** student{

String address;

teacher(){}

teacher(String no,String name,String address){

**super**(no,name);

**this**.address=address;

}

String print() {

**return** no+","+name+","+address;

}

}

# 实验结果及分析

1.验证型实验

（1）分析本程序的执行过程，给出其运行结果

**执行过程： 创建Master类的对象master，调用它的feed方法，创建Dog和Bone类的对象向上转型为Animal和 Food类 作为实参传入，再调用上转型对象的eat和showName方法输出；同理，再创建Cat和Fish类重复上述操作。**

**运行结果：我是狗，我爱吃骨头**

**食物：骨头**

**我是猫，我爱吃鱼**

**食物：鱼**

2.应用型实验

（1）

各个类、方法的作用：主类完成输入创建上转型对象再调用方法输出以及实现学生排序；Student父类提供相应模板；englishStu和computerStu子类主要计算总分。

运行结果：1:2008,black,male,17,Computer,77.0,88.0,76.0,75.0,78.6

2010,adam,male,17,Computer,43.0,60.0,57.0,55.0,51.6

2002,ally,female,18,English,41.0,74.0,92.0,62.0

2009,calina,female,18,Computer,44.0,75.0,75.0,71.0,61.8

2005,black,male,19,English,74.0,90.0,69.0,76.8

2007,fulin,male,19,English,77.0,92.0,68.0,78.5

2006,eve,male,20,Computer,47.0,74.0,79.0,70.0,63.4

1005,dam,male,21,Computer,56.0,43.0,72.0,93.0,64.0

1001,flank,male,22,Computer,91.0,64.0,77.0,74.0,79.4

1007,fulin,male,22,Computer,70.0,53.0,82.0,56.0,66.2

1011,calro,male,23,English,85.0,84.0,97.0,87.8

1000,adam,male,24,English,68.0,78.0,40.0,63.5

1002,alice,female,24,English,75.0,73.0,91.0,78.5

1003,geo,female,24,Computer,89.0,68.0,78.0,60.0,76.8

1008,hali,male,24,English,70.0,94.0,81.0,78.8

2011,bob,male,24,Computer,41.0,76.0,40.0,42.0,48.0

1010,fugui,female,25,English,44.0,78.0,61.0,56.8

2: 2002,ally,female,18,English,41.0,74.0,92.0,62.0

2009,calina,female,18,Computer,44.0,75.0,75.0,71.0,61.8

1007,fulin,male,22,Computer,70.0,53.0,82.0,56.0,66.2

1002,alice,female,24,English,75.0,73.0,91.0,78.5

3: 2009,calina,female,18,Computer,44.0,75.0,75.0,71.0,61.8

2005,black,male,19,English,74.0,90.0,69.0,76.8

1007,fulin,male,22,Computer,70.0,53.0,82.0,56.0,66.2

（2）

各个类、方法的作用：主类输入创建上转型对象并判断是否是优秀学生，是则放入相应的类数组中，最后调用排序方法再输出；Student父类；graduateStu和postStu为子类，主要实现优秀学生的排序

运行结果：excellent undergraduate student:

u2001,ada,true,94.402

excellent graduate student:

g2006,fugui,true,1,96.885

3.设计型实验

（1）

各个类、方法的作用：主类进行输入判断执行1234哪个操作进行具体调用；student父类和teacher子类定义输出方法。

运行结果：1:3

x001,aa

j001,aa,15-1-101

queue is full,operation failed

no found

x001,aa

2:no found

x001,aa

3:2

x001,aa

j001,aa,15-1-101

# 实验小结

[本次实验的反思、收获和体会]

反思：1：思考问题思维不严谨，如第三题进行1234相关操作时我的front与rear没有进行相应的变化。

2：题目有点长时就没有完全看出理解题意。也可能是注意力在其他细节上，其中一点就忽略了。如题目中的“若相应类别优秀生不存在，则任何信息都不输出。”我一开始就没注意到。

3：课上新学的内容如上转型，多态等的一些使用细节不太清楚，导致代码输出总是不在意料之中。新上的内容还不太会，需要去巩固。

收获：真正写代码使用了子类创建对象上转型给父类对象，加深了相关知识点的理解，体会实现多态需继承，重写及上转型三点，也体会到这样写对一些复杂问题还挺好用的。

体会：意识到不同的题目要求类的写法差别还挺大的，如第一第二题，第一题是一起排序，而第二是分开排的，这个转化如何实现我还想了好一会呢。有些题目用java来写确实是有优势的。有些方法的实现还有些吃力，该抓紧去学学相关知识点了。