上机实践 6　接口与实现

实验 1　评 价 成 绩

1．相关知识点

接口体中只有常量的声明（没有变量）和抽象方法声明。而且接口体中所有的常量的访问权限一定都是 public（允许省略 public、final 修饰符），所有的抽象方法的访问权限一定都是 public（允许省略 public、abstract 修饰符）。

接口由类去实现以便绑定接口中的方法。一个类可以实现多个接口，类通过使用关键字 implements 声明自己实现一个或多个接口。如果一个非抽象类实现了某个接口，那么这个类必须重写该接口的所有方法。

2．实验目的

本实验的目的是让学生掌握类怎样实现接口。

3．实验要求

体操比赛计算选手成绩的办法是去掉一个最高分和最低分后再计算平均分，而学校考察一个班级的某科目的考试情况时，是计算全班同学的平均成绩。Gymnastics 类和 School 类都实现了 ComputerAverage 接口，但实现的方式不同。

4．运行效果示例

程序运行效果如图 6.1 所示。

```
体操选手最后得分:9.668
班级考试平均分数:73.09
```

图 6.1　评价成绩

5．程序模板

请按模板要求，将【代码】替换为 Java 程序代码。

Estimator.java

```
interface CompurerAverage {
   public double average(double x[]);
}
class Gymnastics implements CompurerAverage {
   public double average(double x[]) {
      int count=x.length;
      double aver=0,temp=0;
      for(int i=0;i<count;i++) {
         for(int j=i;j<count;j++) {
            if(x[j]<x[i]) {
              temp=x[j];
              x[j]=x[i];
              x[i]=temp;
            }
```

```
        }
      }
      for(int i=1;i<count-1;i++) {
        aver=aver+x[i];
      }
      if(count>2)
        aver=aver/(count-2);
      else
        aver=0;
      return aver;
    }
}
class School implements CompurerAverage {
   【代码1】//重写public double average(double x[])方法,返回数组x[]的元素的算术平均
}
public class Estimator{
   public static void main(String args[]) {
      double a[] = {9.89,9.88,9.99,9.12,9.69,9.76,8.97};
      double b[] ={89,56,78,90,100,77,56,45,36,79,98};
      CompurerAverage computer;
      computer=new Gymnastics();
      double result= 【代码2】 //computer调用average(double x[])方法，将数组a
                                //传递给参数x
      System.out.printf("%n");
      System.out.printf("体操选手最后得分:%5.3f\n",result);
      computer=new School();
      result=【代码3】 //computer调用average(double x[])方法，将数组b传递给参数x
      System.out.printf("班级考试平均分数:%-5.2f",result);
   }
}
```

6. 实验指导

✧ 可以把实现某一接口的类创建的对象的引用赋给该接口声明的接口变量中，那么该接口变量就可以调用被类实现的接口方法。

✧ 接口产生的多态就是指不同类在实现同一个接口时可能具有不同的实现方式。

7. 实验后的练习

School 类如果不重写 public double average(double x[])方法，程序编译时提示怎样的错误？

8. 填写实验报告

实验编号：601 学生姓名：　　实验时间：　　教师签字：

实验效果评价	A	B	C	D	E
模板完成情况					
实验后的练习效果评价	A	B	C	D	E
练习完成情况					
总评					

实验 2　货车的装载量

1．相关知识点

接口回调是多态的另一种体现，接口回调是指：可以把使用某一接口的类创建的对象的引用赋给该接口声明的接口变量中，那么该接口变量就可以调用被类实现的接口中的方法，当接口变量调用被类实现的接口中的方法时，就是通知相应的对象调用接口的方法，这一过程称为对象功能的接口回调。不同的类在使用同一接口时，可能具有不同的功能体现，即接口的方法体不必相同，因此，接口回调可能产生不同的行为。

2．实验目的

本实验的目的是让学生掌握接口回调技术。

3．实验要求

货车要装载一批货物，货物由三种商品组成：电视、计算机和洗衣机。卡车需要计算出整批货物的重量。

要求有一个 ComputeWeight 接口，该接口中有一个方法：

```
public double computeWeight()
```

有三个实现该接口的类：Television、Computer 和 WashMachine。这三个类通过实现接口 computeTotalSales 给出自重。

有一个 Truck 类，该类用 ComputeWeight 接口类型的数组作为成员（Truck 类面向接口），那么该数组的单元就可以存放 Television 对象的引用、Computer 对象的引用或 WashMachine 对象的引用。程序能输出 Truck 对象所装载的货物的总重量。

4．运行效果示例

程序运行效果如图 6.2 所示。

```
货车装载的货物重量:4319.69000 kg
货车装载的货物重量:588.20000 kg
```

图 6.2　货车的装载量

5．程序模板

请按模板要求，将【代码】替换为 Java 程序代码。

CheckCarWeight.java

```
interface ComputerWeight {
   public double computeWeight();
}
class Television implements ComputerWeight {
   【代码1】 //重写computeWeight()方法
}
class Computer implements ComputerWeight {
   【代码2】 //重写computeWeight()方法
}
class WashMachine implements ComputerWeight {
   【代码3】 //重写computeWeight()方法
}
class Truck {
```

```
    ComputerWeight [] goods;
    double totalWeights=0;
    Truck(ComputerWeight[] goods) {
        this.goods=goods;
    }
    public void setGoods(ComputerWeight[] goods) {
        this.goods=goods;
    }
    public double getTotalWeights() {
       totalWeights=0;
       【代码4】 //计算totalWeights
       return totalWeights;
    }
}
public class CheckCarWeight {
    public static void main(String args[]) {
       ComputerWeight[] goods=new ComputerWeight[650]; //650件货物
       for(int i=0;i<goods.length;i++) { //简单分成三类
            if(i%3==0)
              goods[i]=new Television();
            else if(i%3==1)
              goods[i]=new Computer();
            else if(i%3==2)
              goods[i]=new WashMachine();
       }
       Truck truck=new Truck(goods);
       System.out.printf("\n货车装载的货物重量:%-8.5f kg\n",truck.getTotal-
       Weights());
       goods=new ComputerWeight[68]; //68件货物
       for(int i=0;i<goods.length;i++) { //简单分成两类
            if(i%2==0)
              goods[i]=new Television();
            else
              goods[i]=new WashMachine();
       }
       truck.setGoods(goods);
       System.out.printf("货车装载的货物重量:%-8.5f kg\n",truck.getTotalWeights());
    }
}
```

6. 实验指导

✧ 对于【代码 1】，可以简单返回一个值表示货物的重量，比如：

```
public double computeWeight() {
    return 3.5;
}
```

✧ 由于数组 goods 的每个单元存放的是实现 ComputerWeight 接口的对象的引用，实验中的【代码 4】可以通过循环语句让数组 goods 的每个单元调用 computeWeight()方法，并将该方法返回的值累加到 totalWeights，如下所示：

```
for(int i=0;i<goods.length;i++) {
   totalWeights += goods[i].computeWeight();
}
```

7．实验后的练习

请在实验的基础上再编写一个实现 ComputerWeight 接口的类，比如 Refrigerrator。这样一来，货车装载的货物中就可以有 Refrigerrator 类型的对象。

当系统增加一个实现 ComputerWeight 接口的类后，Truck 类需要进行修改吗？

8．填写实验报告

实验编号：602 学生姓名：　　实验时间：　　教师签字：

实验效果评价	A	B	C	D	E
模板完成情况					
实验后的练习效果评价	A	B	C	D	E
练习完成情况					
总评					

实验 3　小狗的状态

1．相关知识点

在设计程序时，经常会使用接口，其原因是，接口只关心操作，但不关心这些操作具体实现的细节，可以使程序的设计者把主要精力放在程序的设计上，而不必拘泥于细节的实现（细节留给接口的实现者），即避免设计者把大量的时间和精力花费于具体的算法上。

使用接口进行程序设计的核心技术之一是使用接口回调，即将实现接口的类的对象的引用放到接口变量中，那么这个接口变量就可以调用类实现的接口方法。

所谓面向接口编程，是指当设计某种重要的类时，不让该类面向具体的类，而是面向接口，即所设计类中的重要数据是接口声明的变量，而不是具体类声明的对象。

2．实验目的

本实验的目的是让学生掌握面向接口编程思想。

3．实验要求

小狗在不同环境条件下可能呈现不同的状态表现，要求用接口封装小狗的状态。具体要求如下。

（1）编写一个接口 DogState，该接口有一个名字为 void showState()的方法。

（2）编写 Dog 类，该类中有一个 DogState 接口声明的变量 state。另外，该类有一个 show()方法，在该方法中让接口 state 回调 showState()方法。

（3）编写若干个实现 DogState 接口的类，负责刻画小狗的各种状态。

（4）编写主类，在主类中测试小狗的各种状态。

4. 运行效果示例

程序运行效果如图 6.3 所示。

```
狗在主人面前:听主人的命令
狗遇到敌人:狂叫，并冲向去很咬敌人
狗遇到朋友:晃动尾巴,表示欢迎
狗遇到同伴:嬉戏
```

图 6.3　小狗的状态

5. 程序模板

请按模板要求，将【代码】替换为 Java 程序代码。

CheckDogState.java

```
interface DogState {
    public void showState();
}
class SoftlyState implements DogState {
    public void showState() {
       System.out.println("听主人的命令");
    }
}
class MeetEnemyState implements DogState {
   【代码1】 //重写public void showState()
}
class MeetFriendState implements DogState {
   【代码2】 //重写public void showState()方法
}
class MeetAnotherDog implements DogState {
   【代码3】 //重写public void showState()方法
}
class Dog {
   DogState  state;
   public void show() {
      state.showState();
   }
   public void setState(DogState s) {
      state = s;
   }
}
public class CheckDogState {
   public static void main(String args[]) {
      Dog yellowDog =new Dog();
      System.out.print("狗在主人面前:");
      yellowDog.setState(new SoftlyState());
      yellowDog.show();
      System.out.print("狗遇到敌人:");
      yellowDog.setState(new MeetEnemyState());
      yellowDog.show();
      System.out.print("狗遇到朋友:");
      yellowDog.setState(new MeetFriendState());
      yellowDog.show();
      System.out.print("狗遇到同伴:");
```

```
        yellowDog.setState(new MeetAnotherDog());
        yellowDog.show();
    }
}
```

6. 实验指导

当增加一个实现 DogState 接口的类后，Dog 类不需要进行修改。

7. 实验后的练习

用面向接口的思想编写程序，模拟水杯中的水在不同温度下可能出现的状态。

8. 填写实验报告

实验编号：603 学生姓名：　　实验时间：　　教师签字：

实验效果评价	A	B	C	D	E
模板完成情况					
实验后的练习效果评价	A	B	C	D	E
练习完成情况					
总评					

实 验 答 案

实验 1：

【代码 1】

```
public double average(double x[]) {
     int count=x.length;
     double aver=0;
     for(int i=0;i<count;i++) {
        aver=aver+x[i];
     }
     aver=aver/count;
     return aver;
}
```

【代码 2】computer.average(a);

【代码 3】computer.average(b);

实验 2：

【代码 1】

```
public double computeWeight() {
     return 3.5;
}
```

【代码 2】

```
public double computeWeight() {
```

```
    return 2.67;
}
```

【代码 3】

```
public double computeWeight() {
    return 13.8;
}
```

【代码 4】

```
for(int i=0;i<goods.length;i++) {
    totalWeights += goods[i].computeWeight();
}
```

实验 3：

【代码 1】

```
public void showState() {
    System.out.println("狂叫，并冲向去很咬敌人");
}
```

【代码 2】

```
public void showState() {
    System.out.println("晃动尾巴,表示欢迎");
}
```

【代码 3】

```
public void showState() {
    System.out.println("嬉戏");
}
```

自 测 题

1. 下列哪个代码替换程序中的【代码】不会导致编译错误？

（A）protected int getNumber() { return 100; }

（B）int getNumber() { return 100; }

（C）public int getNumber() { return 100; }

（D）int getNumber() { return 'a'+'b'; }

```
interface class AAA {
   abstract int getNumber();
}
class BBB implements AAA {
   【代码】
}
```

2. 请说出 E 类中 System.out.printf 的输出结果。

```
interface Computer {
    int computer(int x,int y);
}
abstract class AA {
    int computer(int x,int y) {
        return x-y;
    }
}
class B extends AA implements Computer{
   public int computer(int x,int y) {
      return x+y;
   }
}
public class E {
   public static void main(String args[]) {
     Computer com =new B();
     AA a=new B();
     System.out.printf("%d\n",com.computer(12,10));
     System.out.printf("%d\n",a.computer(15,5));
   }
}
```

答案：

1. C
2. 22
   20