实验五 继承、多态与接口

学生姓名：黄晨箬　　学　　号：6109119066　　专业班级：计算机193班

实验类型：■ 验证 □ 综合 □ 设计 □ 创新　实验日期：2021.5.21　实验成绩：

一、实验目的

1、掌握继承和多态的概念与实现方法

2、掌握如何从已有的类中派生子类并继承父类。

3、掌握方法的重写（覆盖）和重载。

4、掌握抽象类和接口的定义和实现；

5、掌握数组列表ArrayList类的使用。

二、实验内容

习题 1. 编写程序模拟中国人、美国人是人，北京人是中国人，完成相应类别人群的基本信息输出。除主类外，程序中还有 4 个类：People、ChinaPeople、AmericanPeople 和 BeijingPeople 类。要求如下：

 People 类有权限是 protected 的 double 型成员变量 height 和 weight，以及 public void speakHello()、public void averageHeight()和 public void averageWeight()方法。

 ChinaPeople 类是 People 的子类，新增了 public void chinaGongfu()方法。要求 ChinaPeople 重写父类的 public void speakHello()、public void averageHeight()和 public void averageWeight()方法。

 AmericanPeople 类是 People 的子类，新增 public void americanBoxing()方法。要求 AmericanPeople 重写父类的 public void speakHello()、public void averageHeight()和 public void averageWeight()方法。

 BeijingPeople 类是 ChinaPeople 的子类，新增 public void beijingOpera()方法。要求 BeijingPeople 重写父类的 public void speakHello()、public void averageHeight()和 public void averageWeight()方法。

请画出 People、ChinaPeople、AmericanPeople 和 BeijingPeople 类的 UML 图及相应关系，然后程序先依次输出 ChinaPeople、AmericanPeople 和 BeijingPeople 共有行为和属性结果，再输出各类特有行为和属性的输出结果，并得到下列参考运行实例效果：

```
您好
How do you do
您好
中国人的平均身高:168.78 厘米
American's average height:176.0 cm
北京人的平均身高:172.5 厘米
中国人的平均体重:65.0 千克
American's average weight:75.0 kg
北京人的平均体重:70.0 千克
坐如钟,站如松,睡如弓
直拳、钩拳、组合拳
花脸、青衣、花旦和老生
坐如钟,站如松,睡如弓
```

其中 People 类源文件参考如下：

```
//People.java
public class People {
   protected double weight,height;
   public void speakHello() {
      System.out.println("您好");
   }
   public void averageHeight() {
       height=173;
       System.out.println("average height:"+height);
   }
   public void averageWeight() {
      weight=70;
      System.out.println("average weight:"+weight);
   }
}
```

习题 2. 阅读以下程序，获得运行结果，并回答下列问题（各问题不串联，请独立分析）。

```
1  class SuperClass {
2      int x;
3      SuperClass(){
4          x = 4;
5      }
6      void doSomething() {
7          System.out.println("in SuperClass.doSomething:");
8          System.out.println("x="+x);
9      }
10 }
11
12 class SubClass extends SuperClass {
13  int x;
14  SubClass(){
15     super();//要放在方法中的第一句
16     x = 5;
17     System.out.println("in SubClass:");
18     System.out.println("x="+x);
19 }
20
21 void doSomething( ) {
22    super.doSomething(); //super调用父类的方法
23    System.out.println("in SubClass.doSomething:");
24    System.out.println("super.x="+super.x+" sub.x="+x);
25 }
26 }
27
28 public class Inheritance {
29  public static void main(String[] args){
30     SubClass subC=new SubClass();
31     subC.doSomething();
32   }
33 }
```

问题 1：将本程序第 13 行及第 16 行删去，程序的输出结果有何变化？为什么？

问题 2：使用多行注释，将第 21 行至第 25 行内容不执行，该程序能获得输出结果吗？若能，给出结果，并进行适当分析。若不能，请分析原因。

问题 3：将第 30 行、31 行分别改为

```
SuperClass superA = new SubClass();          //第 30 行
superA.doSomething();                        //第 31 行
```

程序的输出结果有变化吗？为什么？

习题 3. 有来自 4 类（鸟类、昆虫类、爬行类和鱼类）的 100 个动物聚在一起开会，商议和另一个动物部落进行运动会事宜，会议要求每个动物都要报告自己所属的动物类别和自己的天赋，以便选拔人才、组织队伍参赛。

设计：

1. 用 Animal 作为基类，类 Animal 中要有属性成员编号、类别，以及 showMessage()方法将所有属性成员信息输出。
2. 设计 Talent 接口，接口中有抽象方法 showTalent()。
3. 鸟类、昆虫类、爬行类和鱼类均继承 Animal 并实现 Talent 接口，同时对每个子类至少要添加一个描述的动物特有的行为和一个特有的属性，以更准确地描述子类对象。

使用：用循环随机生成这 100 个动物装入动物数组，用循环让每个参会的动物报告自己的类别和天赋。 设计时运用继承、重写并设计多态机制。

习题 4-5. 完成课本下列章节的编程题内容。

习题 4：第 11 章 课本 389 页 11.13（去掉重复元素）

习题 5：第 11 章 课本 390 页 11.17（代数：完全平方）

三、实验要求

1、要求学生在实验前一定要非常清楚并灵活运用涉及章所讲过的内容；

2、在上机实验前编写好实验内容要求的程序，以便上机实验时调试、修改等。

3、上机实验后，每位学生必须对该次实验内容写一份实验报告，包括程序内容、调试过程、运行情况和结果等。

四、实验环境

1、PC微机；

2、DOS操作系统或 Windows 操作系统；

3、Eclipse程序集成环境。

五、实验步骤

1、根据题目要求，画出UML类图，并标明各类间关系；

2、给出本程序的java数据结构；

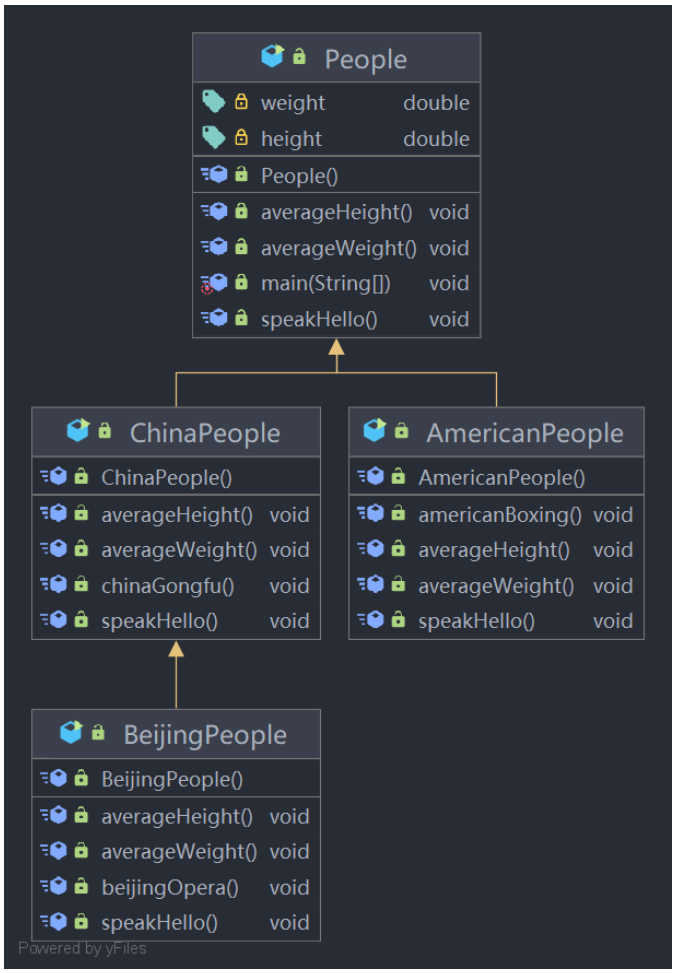
3、编写出程序；

4、调试程序：给出一些测试数据，检查输出结果。

六、实验数据及处理结果

习题一：

[程序流程图]



[数据结构设计]

```
/*
    按照实验内容中提供的信息，依次创建不同的类，并编写新增的方法和重构父类的方法，最后依次调用运行即可。
*/
```

[程序源代码]

```java
public class People {
    protected double weight,height;
    public void speakHello() {
        System.out.println("您好");
    }
    public void averageHeight() {
        height=173;
        System.out.println("average height:"+height);
    }
    public void averageWeight() {
        weight=70;
        System.out.println("average weight:"+weight);
    }

    public static void main(String[] args) {
        ChinaPeople c = new ChinaPeople();
        AmericanPeople a = new AmericanPeople();
        BeijingPeople b = new BeijingPeople();

        c.speakHello();
        a.speakHello();
        b.speakHello();

        c.averageHeight();
        a.averageHeight();
        b.averageHeight();

        c.averageWeight();
        a.averageWeight();
        b.averageWeight();

        c.chinaGongfu();
```

```java
        a.americanBoxing();
        b.beijingOpera();

    }
}

class ChinaPeople extends People {
    public ChinaPeople () {

    }

    public void chinaGongfu() {
        System.out.println("坐如钟、站如松、睡如弓");
    }

    public void speakHello() {
        System.out.println("您好");
    }

    public void averageHeight () {
        height = 168.78;
        System.out.println("中国人的平均身高: " + height + " 厘米");
    }

    public void averageWeight() {
        weight = 65.0;
        System.out.println("中国人的平均体重: " + weight + " 千克");
    }
}
```

```java
class AmericanPeople extends People {
    public AmericanPeople() {}

    public void americanBoxing() {
        System.out.println("直拳、钩拳、组合拳");
    }

    public void speakHello() {
        System.out.println("how do you do");
    }

    public void averageHeight() {
        height = 176.0;
        System.out.println("American's average height: " + height + " cm");
    }

    public void averageWeight() {
        weight = 75.0;
        System.out.println("American's average weight: " + weight + " kg");
    }
}
```

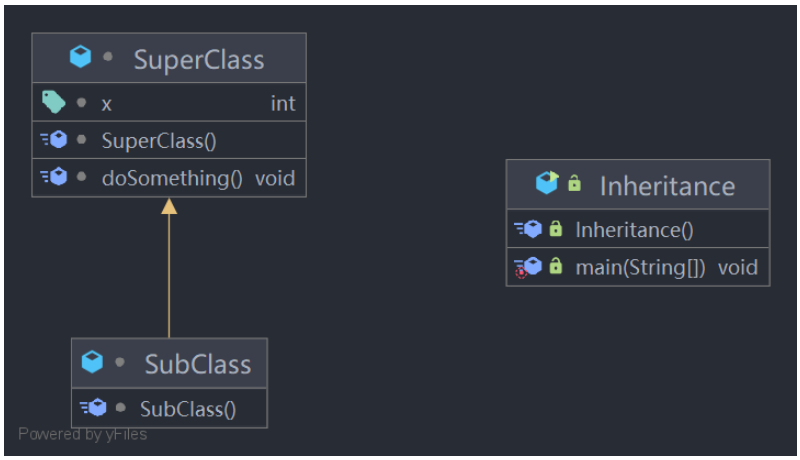
```java
class BeijingPeople extends People {
    public BeijingPeople() {}

    public void beijingOpera() {
        System.out.println("花脸、青衣、花旦和老生");
    }

    public void speakHello() {
        System.out.println("您好");
    }

    public void averageHeight() {
        height = 172.5;
        System.out.println("北京人的平均身高: " + height + " 厘米");
    }

    public void averageWeight() {
        weight = 70.0;
        System.out.println("北京人的平均体重: " + weight + "千克");
    }
}
```

[程序运行结果]

```
您好
how do you do
您好
中国人的平均身高: 168.78 厘米
American's average height: 176.0 cm
北京人的平均身高: 172.5 厘米
中国人的平均体重: 65.0 千克
American's average weight: 75.0 kg
北京人的平均体重: 70.0千克
坐如钟、站如松、睡如弓
直拳、钩拳、组合拳
花脸、青衣、花旦和老生
```

习题二：

[程序流程图]



[数据结构设计]

省略

[程序源代码]

```java
class SuperClass {
    int x;
    SuperClass() {
        x = 4;
    }
    void doSomething() {
        System.out.println("in SuperClass.doSomething:");
        System.out.println("x=" + x);
    }
}

class SubClass extends SuperClass {
    int x;
    SubClass() {
        super();
        x = 5;
        System.out.println("in SubClass:");
        System.out.println("x=" + x);
    }

    void doSomething() {
        super.doSomething();
        System.out.println("in SubClass.doSomething:");
        System.out.println("super.x=" + super.x + " sub.x=" + x);
    }
}

public class Inheritance {
    public static void main(String[] args) {
        SuperClass superA = new SubClass();
        superA.doSomething();
    }
```

[程序运行结果]

```
in SubClass:
x=5
in SuperClass.doSomething:
x=4
in SubClass.doSomething:
super.x=4 sub.x=5
```

如果删除13行和16行

```
in SubClass:
x=4
in SuperClass.doSomething:
x=4
in SubClass.doSomething:
super.x=4 sub.x=4
```

如果删除这两行，在SubClass中的x的值会继承SuperClass里的值，所以全部显示为4。

如果将21-25行注释掉，依然可以运行

```
in SubClass:
x=5
in SuperClass.doSomething:
x=4
```

当注释掉SubClass中的doSomething方法后，subC.doSomething(); 指令就不会执行，但是其他的指令仍然会正常运行，所以最后也能成功编译。

当修改了30-31行的内容之后，输出结果没有发生改变

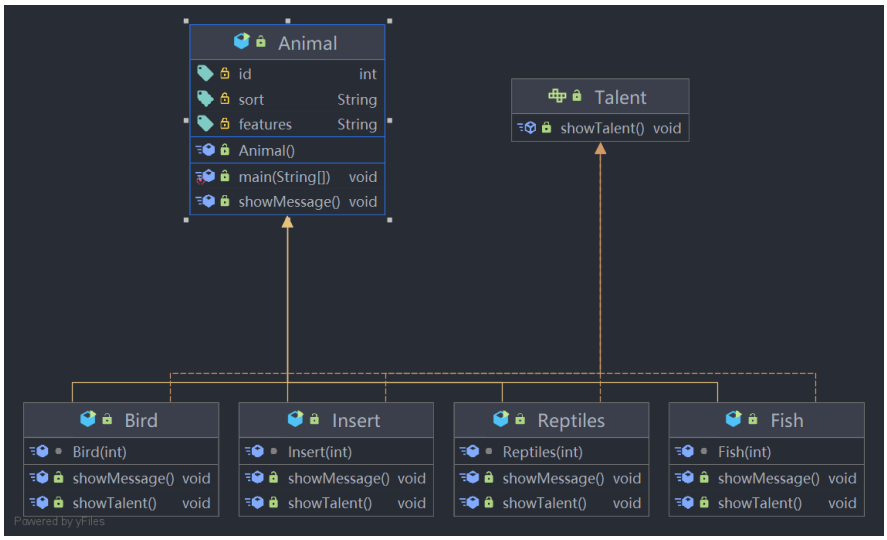
```
in SubClass:
x=5
in SuperClass.doSomething:
x=4
in SubClass.doSomething:
super.x=4 sub.x=5
```

经过观察我们知道，实际上修改的内容只是将subClass修改为了superClass，但实际上，此处是用于声明对象类型，而他们的对象类型是一样的，所以最后的输出结果不会受

到影响。

习题三：

[程序流程图]



[数据结构设计]

```
/*
    首先编写各个类，然后编写随机生成动物类型的方法和随机编号的方法。赋予不同动物有独有的属性
    然后再根据随机生成动物类型所得的字符串创建相对应的对象，然后调用相对应的展示信息的方法输出并重复一百次上述步骤即可。
 */
```

[程序源代码]

```
import java.lang.reflect.InvocationTargetException;
import java.util.Random;
public class Test {
    public static void main(String[] args) throws InstantiationException, IllegalAccessException, ClassNotFoundException
        String []id = new String [100];
        randomId(id, idLen: 5);
        Random r = new Random();
        String species = "";
        for(int i = 0;i<100;i++) {
            species = randomSpecies(r.nextInt( bound: 4));
            Class<Animal> animal1 = (Class<Animal>) Class.forName(species); //ERROR!
            Animal animal = animal1.getDeclaredConstructor().newInstance();
            animal.setAttr(id[i]);
            animal.showMessage();
            animal.showTalent();
        }
    }
```

```java
    public static String randomSpecies(int i) {  //随机动物类型
        switch(i) {
            case 0:
                return "Bird";
            case 1:
                return "Insect";
            case 2:
                return "Reptile";
            case 3:
                return "Fish";
            default:
                return "";
        }
    }

    public static void randomId(String[] str,int idLen) {
            char []temp = new char[10];
            int index = 0;

            for(int i = 48;i<58;i++) {
                temp[index++] = (char)i;
            }

            Random r = new Random();  //随便取数字
```

```java
        @SuppressWarnings("unused")
            String id = "";
        for(int i = 0;i < str.length;i++) {
            for(int j = 0;j<idLen;j++) {
                index = r.nextInt( bound: 10);
                id += temp[index];
            }
            str[i] = id;
            id = "";
        }
    }
}

abstract class Animal {
    protected String id;
    public String type;
    public abstract void showTalent();

    public void showMessage() {
        System.out.println("我的编号是 " + id + " ,类别是 " + type);
    }

    public void setAttr(String id) {
        this.id = id;
    }
}
```

```java
class Bird extends Animal {
    public Bird() {
        type = "Bird";
    }

    public Bird(String id) {
        type = "Bird";
        this.id = id;
    }

    public void showTalent(){
        System.out.println("我能飞");
    }
}

class Insect extends Animal {
    public Insect() {
        type = "Insect";
    }

    public Insect(String id) {
        type = "Insect";
        this.id = id;
    }

    public void showTalent() {
        System.out.println("我很小");
    }
}
```

```java
class Reptile extends Animal {
    public Reptile() {
        type = "Reptile";
    }

    public Reptile(String id) {
        type = "Reptile";
        this.id = id;
    }

    public void showTalent() {
        System.out.println("我很强壮");
    }
}

class Fish extends Animal {
    public Fish() {
        type = "Fish";
    }

    public Fish(String id) {
        type = "Fish";
        this.id = id;
    }

    public void showTalent() {
        System.out.println("我能游泳");
    }
}
```

[程序运行结果]

```
Exception in thread "main" java.lang.ClassNotFoundException Create breakpoint : Insect <2 internal calls>
    at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:521)
    at java.base/java.lang.Class.forName0(Native Method)
    at java.base/java.lang.Class.forName(Class.java:315)
    at exp_5.Test.main(Test.java:13)
```

这里应该是步骤中的“根据字符串内容生成对应的对象”这一步的代码错误，一开始查询发现使用forName和newInstance方法可以达到目标，但是实际上手后发现newInstance方法已经被弃用，于是改用getDeclaredConstructor().newInstance(),运行后仍然报错，多次查询无果。

习题四：

[程序流程图]

省略

[数据结构设计]

```java
/*
    首先将输入的字符串按照空格分割并转化为数字类型后依次存入数组，之后对数组调用removeDuplicate方法
    将每个数都分别和之后的数比较，如果相同，就删除后面的那个相同的元素。
 */
```

[程序源代码]

```java
public class repeat {
    public static void removeDuplicate(ArrayList<Integer> list) {
        for (int i = 0; i < list.size(); i++) {
            int a = list.get(i);
            for (int j = i+1; j < list.size(); j++) {
                if (a == list.get(j)) {
                    list.remove(j);
                }
            }
        }

        System.out.print("The distinct integers are ");
        for (Integer i : list) {
            System.out.print(i + " ");
        }
    }

    public static void main(String[] args) {
        ArrayList<Integer> nums = new ArrayList<Integer>();

        System.out.print("Enter 10 integers: ");

        Scanner input = new Scanner(System.in);
        while (input.hasNextLine()) {
            String[] str = input.nextLine().split( regex: " ");
            for (String a : str) {
                nums.add(Integer.valueOf(a));
            }
            break;
        }

        removeDuplicate(nums);
    }
```

[程序运行结果]

```
Enter 10 integers: 34 5 3 5 6 4 33 2 2 4
The distinct integers are 34 5 3 6 4 33 2
```

习题五：

[程序流程图]

省略

[数据结构设计]

```
/*
    将从键盘上读取的数存入a中，然后判断a是否为正确答案（14），如果是，则跳出循环，输出got it ，
    如果不是，则判断数组列表中是否含有该值，若是，则输出提示
    如果不是，则将该数加入数组列表中，并输出错误信息重新调用answer方法
 */
```

[程序源代码]

```java
public static void alreadyAdd(ArrayList<Integer> list, int num) {
    for (Integer integer : list) {
        if (integer == num) {
            System.out.println("You already entered " + num);
            break;
        }
    }
}

public static void answer(ArrayList<Integer> list) {
    Scanner input = new Scanner(System.in);

    int a = input.nextInt();
    while (a != 14) {
        alreadyAdd(list, a);
        list.add(a);
        System.out.print("Wrong answer. Try again. What is 5 + 9? ");
        answer(list);
        a = 14;
    }
}

public static void main(String[] args) {
    ArrayList<Integer> list = new ArrayList<Integer>();

    System.out.print("what is 5 + 9? ");
    answer(list);
    System.out.println("You got it!");
}
```

[程序运行结果]

```
what is 5 + 9? 12
Wrong answer. Try again. What is 5 + 9? 34
Wrong answer. Try again. What is 5 + 9? 12
You already entered 12
Wrong answer. Try again. What is 5 + 9? 14
You got it!
```

七、思考题

1. 试说明super关键字和this关键字的含义及主要用途。

   在Java中，this通常指当前对象，super则指父类。

   this可以用于获取当前对象的方法或某个成员，而super则用于引用父类的某种东西。

2. 试描述Java程序使用抽象类和接口实现类间继承机制的异同。

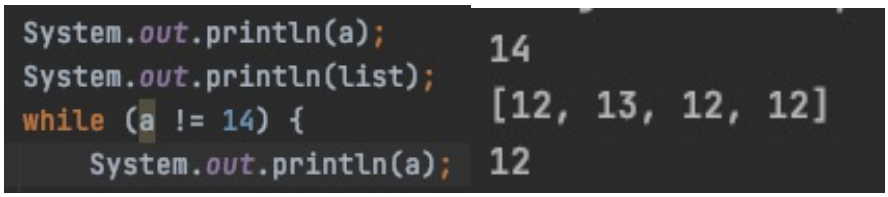
   同：接口和抽象类都不能被实例化，只能被其他类实现和继承

   异：接口允许多继承

八、实验总结与体会

经过这次实验，学习了继承和多态的概念与实现方法，掌握了抽象类和接口的相关知识，尝试了 ArrayList 类的使用。

在实验过程中，第五个实验调试时，发现了问题：



在经过反复测试之后，发现是由于 a 的值在循环中没有更新，导致即使下一次循环从键盘接收的值为 14，也照样会进行循环。解决办法是在 while 循环的最后加上 a = 14。调试后正常运行。

经过本次错误，我学到了之后的实验过程中，需要花费更多的心思，细细推敲错误的发生，并且能做到在问题出现后沉着冷静的发现问题并对应解决。