Day07

Key Point

- 面向对象的特征
- 封装
- 继承
- 多态
  - 编译时多态
  - 运行时多态
- 权限修饰符
- super关键字
- 方法的重写

一、 选择题

1. 有如下代码

```
class Animal{}
class Dog extends Animal{}
class Cat extends Animal{}
public class TestAnimal{
public static void main(String args[]){
//主方法代码省略
}
public static Animal getAnimal(){
//1
}
}
```

问：下列几个选项中，有哪几个放在//1 位置能够编译通过？
A. return null;
B. return new Animal();
C. return new Dog();
D. return new Cat();

2. 有如下代码

```
1) //MyClass.java
2) package corejava.chp6;
3) public class MyClass{4) int value;
5) }
```

```
6)
7) //MySubClass.java
8) package corejava.temp;
9) import corejava.chp6.MyClass;
10) public class MySubClass extends MyClass{
11) public MySubClass(int value){
12) this.value = value;
13) }
14) }
```

选择正确答案：
A. 编译通过
B. 编译不通过，应把第12 行改成super.value = value;
C. 编译不通过，应把第12 行改成super(value);
D. 编译不通过，可以为MySubClass 增加一个value 属性
E. 编译不通过，把 第 4行改为 protected int value; 把第 12 行改为super.value = value;

3. 下列各项的说法正确的是：
A. 继承可以降低代码的冗余度
B. 内部类是封装的体现形式之一
C. 封装可以提高代码的复用性
D. 运行时多态的前提是有继承或者实现关系
E. Java中类与类之间是单继承，接口与接口之间是多继承

二、 简答题

1. 有以下代码

```
class Super{
public Super(){
    System.out.println("Super()");
}
public Super(String str){
    System.out.println("Super(String)");
}
}
class Sub extends Super{
public Sub(){
    System.out.println("Sub()");
}
public Sub(int i){
    this();
    System.out.println("Sub(int)");
}
public Sub(String str){
    super(str);
    System.out.println("Sub(String)");
```

```
}
}
public class TestSuperSub{
public static void main(String args[]){
    Sub s1 = new Sub();
    Sub s2 = new Sub(10);
    Sub s3 = new Sub("hello");
}
}
```

写出该程序运行的结果

2. 看下面代码，写出程序运行的结果

```
class Super{
public void m1(){
    System.out.println("m1() in Super" );
}
}
public void m2(){
    System.out.println("m2() in Super" );
}
}
class Sub extends Super{
public void m1(){
    System.out.println("m1() in Sub");
    super.m1();
}
}
public class TestSuperSub{
public static void main(String args[]){
    Sub s = new Sub();
    s.m1();
    s.m2();
}
}
```

3. 有如下代码

```
class Super{
public void method(){
    System.out.println("method() in Super");
}
public void method(int i){
    System.out.println("method(int) in Super");
}
}
class Sub extends Super{
public void method(){
```

```
    System.out.println("method() in Sub");
}
public void method(String str){
    System.out.println("method(String) in Sub");
}
}
public class TestSuperSub{
public static void main(String args[]){
    Super s = new Sub();
    s.method(10);s.method();
    s.method("hello");
}
}
```

问：该程序是否能编译通过？如果可以，输出结果是什么？如果不可以，应该如何修改？

4. 有以下代码

```
class ClassA{
public ClassA(){
    System.out.println("ClassA()");
}
}
class ClassB{
public ClassB(){
    System.out.println("ClassB()");
}
}
class ClassC extends ClassA{
public ClassC(){
    System.out.println("ClassC()");
}
}
class ClassD extends ClassB{
private ClassA ca = new ClassA();
private ClassC cc;
public ClassD(){
    System.out.println("ClassD()");
}
public ClassD(int i){
    cc = new ClassC();
    System.out.println("ClassD(int)");
}}
public class TestConstructors{
    public static void main(String args[]){
    ClassD cd1 = new ClassD();
    ClassD cd2 = new ClassD(10);
}
}
```

编译运行以上代码，请写出运行时输出的结果

5. 有以下代码

```
class Super{
}
class Sub extends Super{
public Sub(){}
public Sub(String str){
    super(str);
}
}
```

问：该程序应该如何修改才能编译通过？

三、 编程题

1. 有如下代码

```
class Animal{
private String name;
// 1
}
class Dog extends Animal{
//2
}
class Cat extends Animal{
//3
}
public class TestAnimal{
public static void main(String args[]){
    Animal[] as = new Animal[]{
    new Dog("Pluto"),
    new Cat("Tom");
    new Dog("Snoopy");
    new Cat("Garfield");
    };
    Dog[] dogs = getAllDog(as);
    for(int i = 0; i<=dogs.length; i++){
    System.out.println(dogs[i].getName());}
    }
    public static Dog[] getAllDog(Animal[] as){
    //4
}
}
```

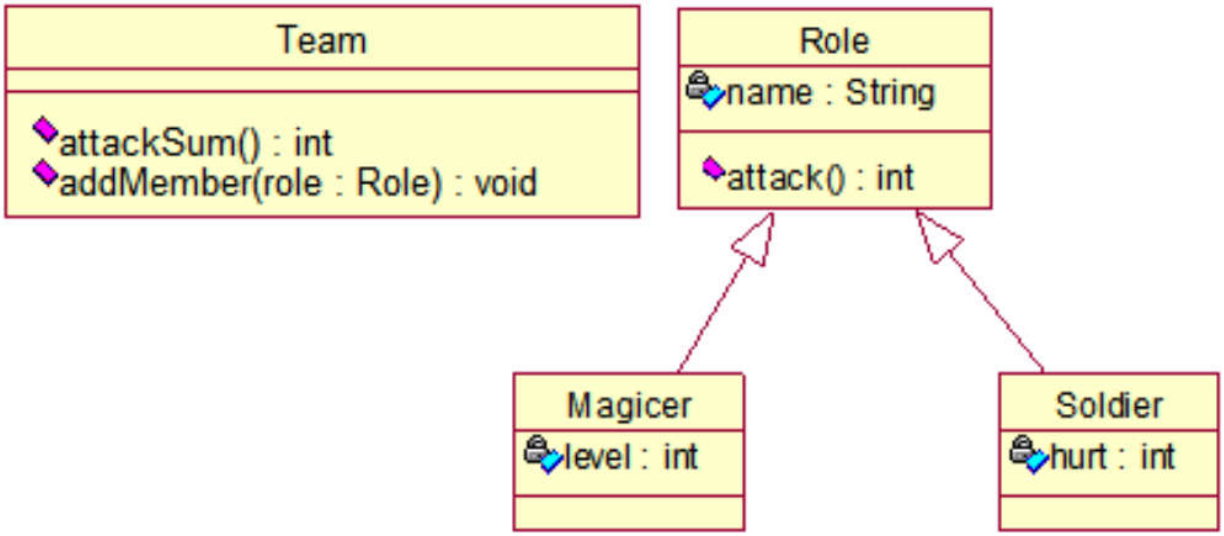
程序填空：
a) 在 //1, //2, //3 处填上适当的get/set 方法和构造方法
b) 完成//4 处的填空。 getAllDog 方法从一个Animal 数组中挑选出所有的Dog 对象，并把这些对象放在一个Dog 数组中返回。

2. 已知一个类Student 代码如下：

```
class Student{
String name;
int age;
String address;
String zipCode;
String mobile;
}
```

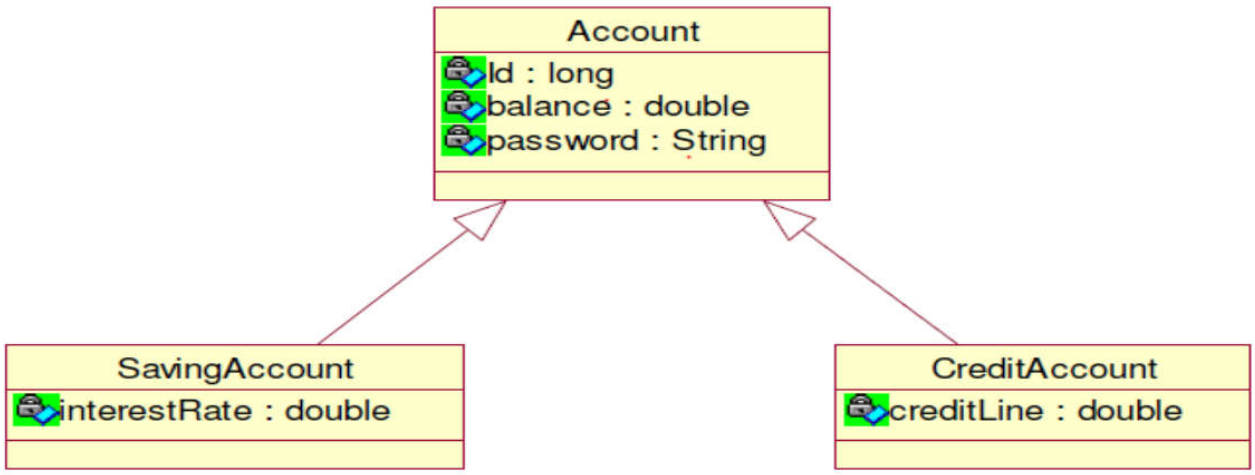
a) 把Student 的属性都作为私有，并提供get/set 方法以及适当的构造方法。
b) 为Student 类添加一个getPostAddress 方法，要求返回Student 对象的地址和邮编。

3. 创建三个类，组成一个继承树，表示游戏中的角色。
描述如下：
父类： Role。是所有职业的父类。
属性： name，表示角色的名字。
方法： public int attack()，该方法返回值为角色的攻击对敌人的伤害。
Role 有两个子类：
1 ）法师Magicer
属性：魔法等级（范围为1 ~ 10 ）
方法：
public int attack()，该方法返回法师的攻击对敌人造成的伤害值。法师攻击伤害值为：魔法等级*魔法基本伤害值（固定为5 ）
2 ）战士Soldier
属性：攻击伤害值
方法：
public int attack()，该方法返回战士的攻击对敌人造成的伤害值。战士的攻击伤害值为：其攻击伤害属性值
注意：上述的三个类所有属性都应当作为私有，并提供相应的get/set 方法。
再设计一个Team 类，表示一个组队。具有如下方法:
1 ）addMember，表示组队增加一个成员。注意：组队成员最多为6 人.提示：应当利用一个数组属性，保存所有成员
2 ）attackSum，表示组队所有成员进行攻击时，对敌人造成的总伤害值
省略 get/set 方法后的类图如下：



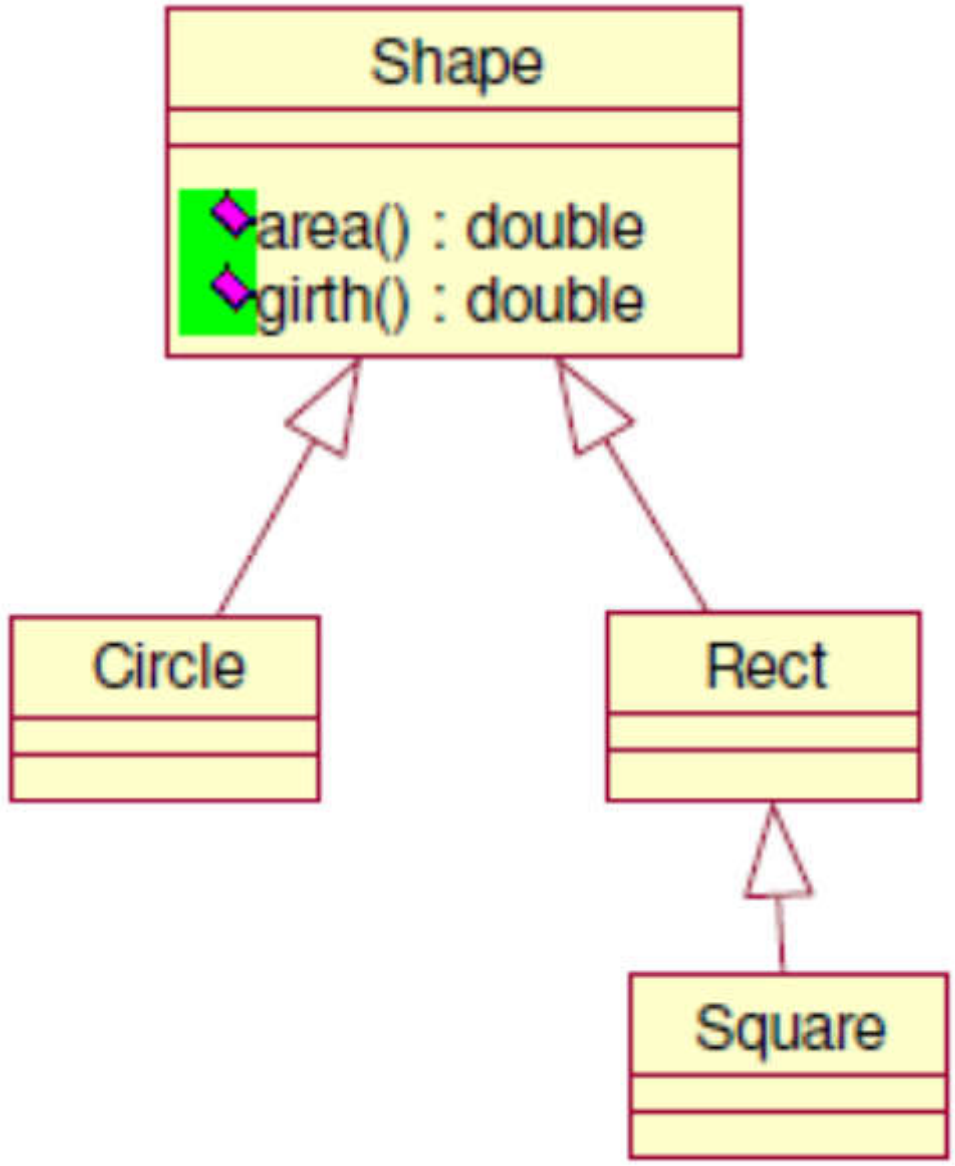
根据类图和描述，创建相应的类。并编写相应的测试代码。

4. 设计如下的继承树：



Accout 表示银行账户，id 属性表示账户id，balance 表示账户余额，password 表示账户密码；
SavingAccount 表示储蓄账户， interestRate 表示存款利率；
CreditAccount 表示信用账户， creditLine 表示信用额度。
完成下列任务：
1）所有属性都应设为私有，根据需要增加构造方法和get/set 方法。
2）修改getPassword 方法，要求每次都返回null 值。
3）修改 interestRate 的 set 方法，要求利率大于 0 并小于 10%。

5. 有以下几个类，根据下面的继承关系，用 Java 代码实现。



a) Circle 类（圆形），属性：半径；方法：求周长、求面积
b) Rect 类（矩形），属性：长、宽；方法：求周长、求面积
c) Square 类（正方形），属性：边长；方法：求周长、求面积
提示：
1）这三个类均具有求周长和面积的方法；
2）正方形是特殊的矩形

6. 在上一题的基础上，创建一个长度为3 的数组，里面有三个不同类型的对象，分别打印这三个对象的周长和面积。