CoreJava 附加题目

习题：

1. （循环）输入一个整数，计算它各位上数字的和。（注意：是任意位的整数）

2. （循环）计算圆周率

   中国古代数学家研究出了计算圆周率最简单的办法:

   PI=4/1-4/3+4/5-4/7+4/9-4/11+4/13-4/15+4/17

   这个算式的结果会无限接近于圆周率的值,我国古代数学家祖冲之计算出,圆周率在 3.1415926 和

   3.1415927 之间,请编程计算,要想得到这样的结果,他要经过多少次加减法运算?

3. （循环）已知：faibonacci（费波那契）数列的前几个数分别为 0,1,1,2,3,5……。

   从第 3 项开始，每一项都等于前两项的和。读入一个整数 n，编程求出此数列的前 n 项。

   注意：这里的数列是从 0 开始的。

4. （数组）看下面的代码，写出输出的结果：

```
public class Ex2 {
    public static void main(String[] args) {
        int[] a = {1,2,3,4,5};
        expand(a);
        changeArray(a);
        printArray(a);
    }
    public static void expand(int[] a){
```

```
        int[] newArray = new int[a.length * 2];
        System.arraycopy(a, 0, newArray, 0, a.length);
        a = newArray;
    }
    public static void changeArray(int[] a){
        a[0] = 10;
    }
    public static void printArray(int[] a){
        for(int i = 0; i<a.length; i++){
            System.out.print(a[i] + "\t");
        }
        System.out.println();
    }
}
```

请选择输出结果：

A. 10 2 3 4 5

B. 1 2 3 4 5

C. 10 2 3 4 5 0 0 0 0 0

D. 1 2 3 4 5 0 0 0 0 0

5. （数组）筛选法求质数：输入一个整数 n，求小于这个整数的所有质数。

算法：定义一个长度为 n 的 boolean 数组，true 表示是质数，false 表示不是质数。初始均为 true。

之后从 2 开始循环：

I. 找到第一个值为 true 的下标 i

II. 把所有下标为 i 的倍数的值置为 false。

III. 直到扫描完数组中的所有数值。

IIV. 最后遍历数组，如果下标 i 的值为 true，则说明 i 为质数。

6. （数组）十五个猴子围成一圈选大王，依次 1-7 循环报数，报到 7 的猴子被淘汰，直到最后一只猴子成为大王。问，哪只猴子最后能成为大王？

7. （数组）八皇后问题

在 8×8 的国际象棋盘上，放置八个皇后，使任何一个皇后都不能吃掉另一个。国 际象棋规则中，皇后可以吃到任何一个与他在同一行、同一列或者同一斜线上的敌方棋子，所以八皇后问题的所有解满足。8 个皇后都不在同一行、同一列，或者同 一斜线上。

输出所有的解。

提示：使用递归。

8. （封装、继承、多态）创建三个类，组成一个继承树，表示游戏中的角色。

描述如下：

I. 父类：Role（角色）。是所有职业的父类。

1). 属性：name，表示角色的名字。

2). 方法：public int attack()，该方法返回值为角色的攻击对敌人的伤害。

II. Role 的第一个子类：Magicer（法师）

1). 属性：魔法等级（范围为 1 ~ 10）。

2). 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）。

III. Role 的第二个子类：Soldier（战士）

1). 属性：攻击伤害值

2). 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。

IV. 再设计一个 Team 类，表示一个组队。具有如下方法：

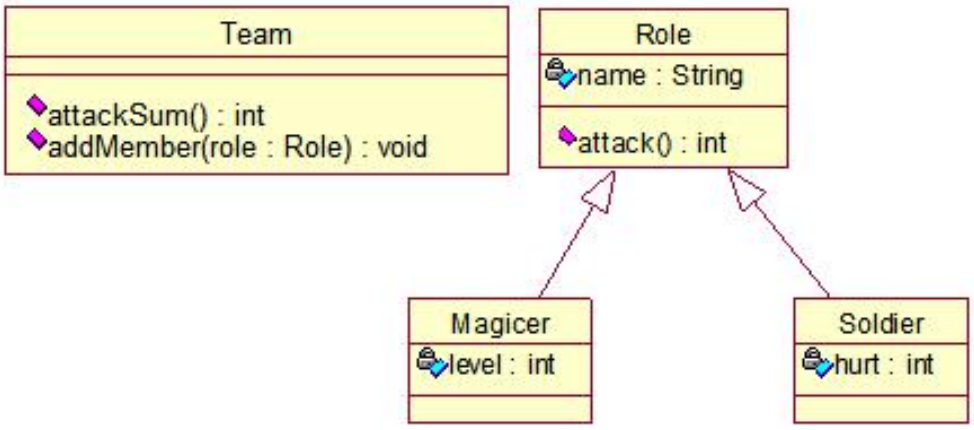
1). addMember，表示组队增加一个成员。

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：



9. （静态初始化代码块，对象创建的过程）有以下代码

class ClassA{

```
    static {
        System.out.println("In ClassA Static");
    }
    public ClassA(){
        System.out.println("ClassA()");
    }
}
class ClassB{
    static {
        System.out.println("In ClassB Static");
    }
    public ClassB(){
        System.out.println("ClassB()");
    }
}
class ClassC extends ClassB{
    static{
        System.out.println("In ClassC Static");
    }
    public ClassC(){
        System.out.println("ClassC()");
    }
}
class MyClass {
```

```
    static ClassA ca = new ClassA();
    ClassC cc = new ClassC();
    static{
        System.out.println("In MyClass Static");
    }
    public MyClass(){
        System.out.println("MyClass()");
    }
}
public class TestMain{
    public static void main(String args[]){
        MyClass mc1 = new MyClass();
        MyClass mc2 = new MyClass();
        System.out.println(mc1.cc == mc2.cc);
        System.out.println(mc1.ca == mc2.ca);
    }
}
```

写出这个程序运行的结果。

10. 在之前的游戏角色 Role 程序上进行修改。
    I. 创建 Role 接口，包含两个方法：
        1). int attack(); 表示攻击，返回值表示对敌人的伤害
        2). void practise(); 表示练习。练习之后对敌人的伤害会增加。
    II. 创建 NamedRole 类，该类为一个抽象类，实现了 Role 接口，并有两个属性：

1). name：角色的名字

2). age：角色的年龄。

III. 增加 MagicStick 接口。该接口表示法师使用的法杖。接口中包含一个方法：

1). int fire()

IV. 为 MagicStick 类增加两个实现类，分别为 GreenStick 和 BlackStick。

其中，对于这两个类的 fire 方法：

1). GreenStick 平时返回 1，夏天（6~8 月）使用时返回 2

2). BlackStic 奇数月返回 1，偶数月返回 2

V. 修改 Magicer 类

1). 为法师类增加 MagicStick 类的属性 stick，表示法师使用的法杖。

2). 让其继承自 NamedRole 类，并实现 attack 和 practise 功能。

a). int attack()返回值为法师的魔法等级(level) * 每一级的固定伤害(5)

b). void practise()方法：

i. 当法师的 stick 属性为 null 时，调用 practise 则 level++

ii. 当法师的 stick 不为 null 时，调用 practise 方法时，

法师的等级 level 满足： level = level + 1 + stick.fire()；

即：法师的等级增加为 1+stick 属性的 fire 方法的返回值.

VI. 增加 Weapon 接口，表示战士使用的武器。Weapon 接口中定义了两个方法：

1). void setSoldier(Soldier s); 该方法表示设置武器的使用者

2). int fire(); 该方法的返回值表示战士使用该武器时，对敌人的伤害值

VII. 为 Weapon 增加两个实现了，Bolo，表示大刀，Pike，表示长矛。

对这两个实现类的描述如下：

1). Bolo：

a). 当 soldier 的年龄大于等于 18 岁时，fire 方法返回 100

b). 当 soldier 年龄小于 18 岁时，fire 方法返回 50

2). Pike：Pike 类有一个属性：name，表示长矛的名字。

a). 当长矛的名字和战士的名字一致时，fire 方法返回 1000

b). 当长矛的名字和战士的名字不一致时，fire 方法返回 25

VIII. 修改 Soldier 类

1). 为 Soldier 类增加一个 Weapon 属性，表示战士的武器

2). 让其继承自 NamedRole 类，并实现 attack 和 practise 功能。其中

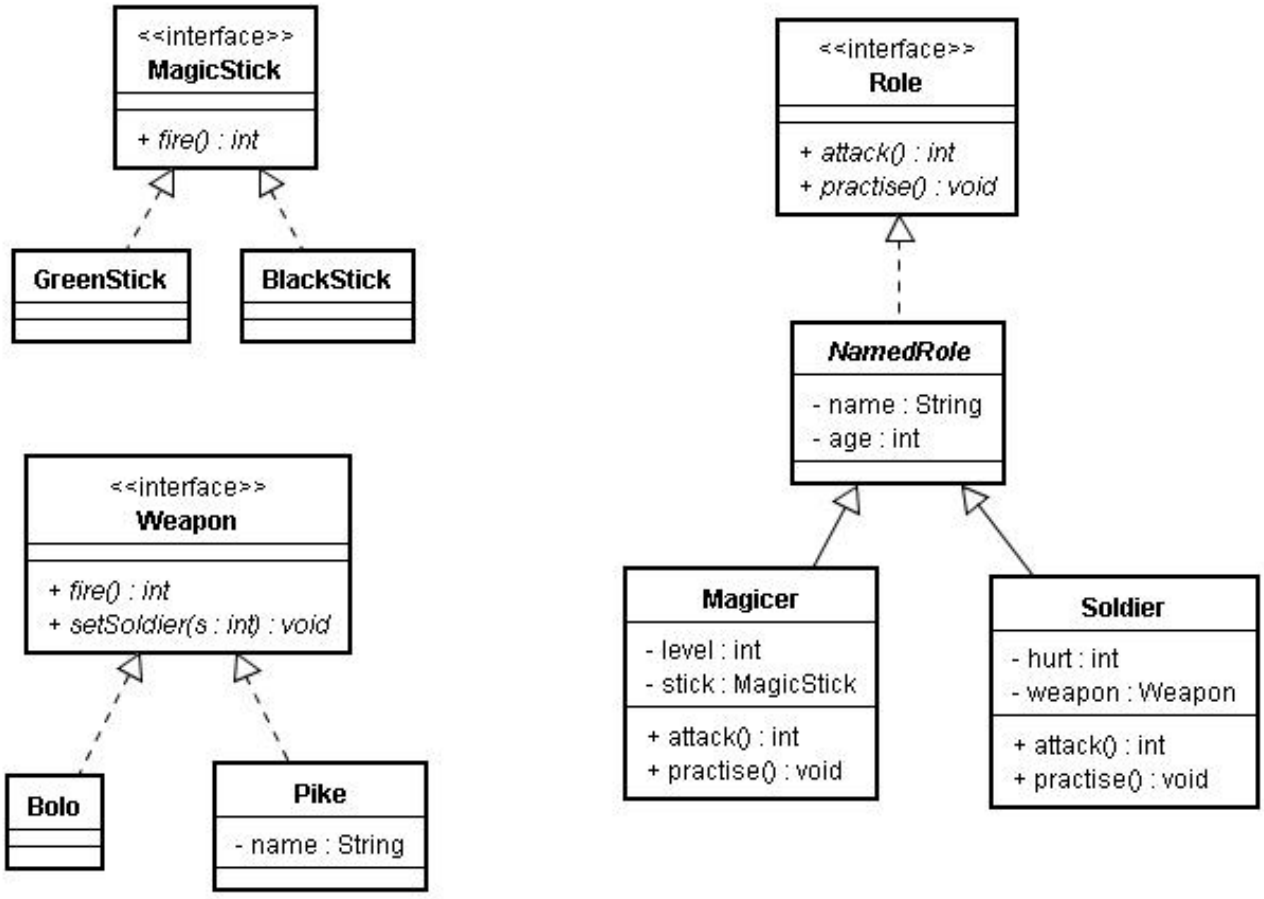
a). Soldier 的 attack() 返回值为战士的 hurt 值与武器的 fire()返回值的和，

即：总攻击输出 = 战士的徒手伤害值 + 武器的伤害值

b). practise()方法：每调用一次则战士的 hurt 值+10

IX. 编写相应的测试代码。

相关类图如下：



11. （Object 类）有下面代码

```
interface IA{
    void ma();
}
class MyClass implements IA{
    public void ma(){}
    public String toString(){
        return “MyClass toString()”;
    }
}
public class TestMyClass{
    public static void main(String args[]){
        IA ia = new MyClass();
        System.out.println(ia);
    }
}
```

选择正确答案：

A. 编译不通过，因为 IA 接口中没有定义 toString 方法

B. 编译通过，输出：“IA@地址”

C. 编译通过，输出：“MyClass toString()”

12. （StringBuilder）给定一个长度，随机产生一个该长度的字符串，由大写,小写字母以及数字组成。

Java 中随机数的生成:

java.util.Random r = new java.util.Random();

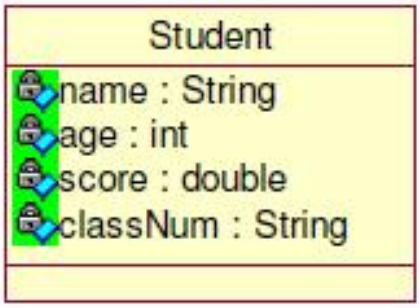
int a = r.nextInt(100): a 0-99 的随机数

提示：生成 int 值，并将 int 值转换成 ASCII 编码对应一个字符。

13. （String）给定一个字符串,判断该字符串中是否包含某个子串.如果包含,求出子串的所有出现位置。

如: "abcd23abc34bcd"中, "bc"子串的出现位置为: 1,7,11.字符串和子串均由用户输入。

14. （综合）有如下 Student 对象



其中，classNum 表示学生的班号，例如“class05”。

有如下 List：

List<Student> list = new ArrayList<Student>();

list.add(new Student(“Tom”, 18, 100, “class05”));

list.add(new Student(“Jerry”, 22, 70, “class04”));

list.add(new Student(“Owen”, 25, 90, “class05”));

list.add(new Student(“Jim”, 30,80 , “class05”));

list.add(new Student(“Steve”, 28, 66, “class06”));

list.add(new Student(“Kevin”, 24, 100, “class04”));

在这个 list 的基础上，完成下列要求：

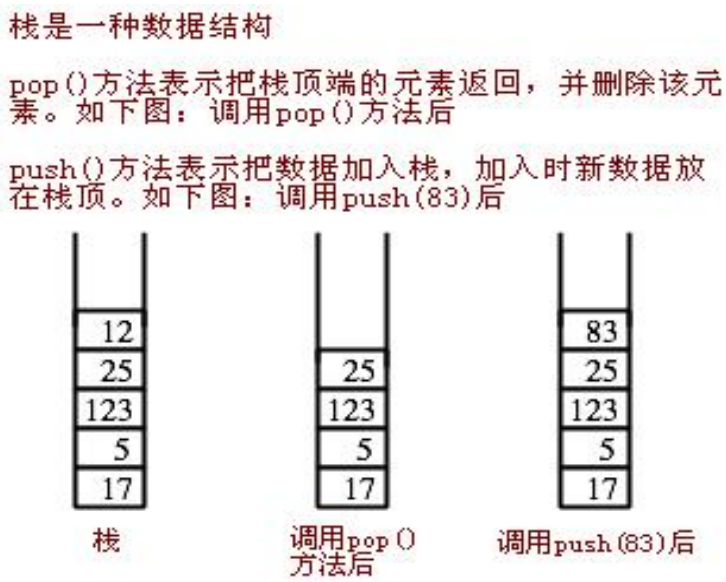
I. 计算所有学生的平均年龄

II. 计算各个班级的平均分

15. （综合）写一个 MyStack 类，表示“栈”这种数据结构。

栈在表示上，就如同一个单向开口的盒子，每当有新数据进入时，都是进入栈顶。 其基本操作为 push 和 pop。push 表示把一个元素加入栈顶，pop 表示把栈顶元素弹出。

示意图如下：



栈的特点：先进后出。

栈的基本操作：

I. push(Object o)：表示把元素放入栈

II. Object pop()：返回栈顶元素，并把该元素从栈中删除。如果栈为空，则返回 null 值。

III. Object peek()：返回栈顶元素，但不把该元素删除。如果栈为空，则返回null 值。

IV. boolean isEmpty()：判断该栈是否为空

V. int size()：返回该栈中元素的数量

要求：利用 List，实现栈。