

主管
领导
审核
签字

哈尔滨工业大学 2020 学年春季学期

软件构造

试 题

题号	一	二	三	四	五	六	七	总分
得分								
阅卷人								

片纸鉴心 诚信不败

本试卷满分 100 分，折合 60%计入总成绩。

一（8分）某方法method有一个参数String credit，它满足的条件是“[1,10]范围内的正整数，或正整数后面带有小数点0.5”。例如，“1”、“1.5”、“2”、“3.5”、“5”、“9.5”、“10”等都是合法输入，但“0”、“0.5”、“03.5”、“6.3”、“7.0”、“10.5”等都是非法输入。请写出该方法开始部分用assert对pre-condition进行合法性检测的Java代码。

```
public void method(String credit) {

    ...//此处开始为method所完成的常规逻辑，无需填写

}
```

二（35分，每小题7分）某ADT的代码如下所示。

```
class Poem {
    public String title;
    public String author;
    private List<String> lines = new ArrayList<>();
    private Date date;

    // AF: 代表一首诗，包含四个属性：
    //     title为诗的题目，
    //     author为诗的作者，
    //     lines为诗的文本行，
    //     date为诗的发表日期

    public Poem(String t, String a, List<String> l, Date d) {
        title = t;
        author = a;
        lines = l;
        date = d;
    }

    public void addOneLine(String newLine) {
```

授课教师

姓名

学号

院系

密

封

线

```

        lines.add(newLine);
    }

    public Poem plagiarize(String newAuthor) {
        return new Poem(title, newAuthor, lines, date);
    }

    public void addPrefix(String prefix) {
        for (int i = 0; i < lines.size(); i++) {
            String newLine = prefix.concat(lines.get(i));
            lines.remove(i);
            lines.add(i, newLine);
        }
    }

    /**
     * @param start 起始行号
     * @param end 结束行号, start<=end
     * @return 从第start行开始到第end行结束的文本行
     */
    public List<String> getSomeLines(int start, int end) {
        List<String> some = new ArrayList<>();
        for (int i = start; i <= end; i++)
            some.add(lines.get(i));
        return some;
    }

    public List<String> getAllLines() {
        return lines;
    }
}

```

1 请找出上述代码中存在表示泄露的代码行，写出新代码或者给出简要修改说明，使之避免表示泄露。

答题区

2 某客户端程序如下所示，请绘制出它运行之后的snapshot diagram。

```

final List<String> lines = new ArrayList<>();
lines.add("Happy");
lines.add("Birthday");
Poem p = new Poem("100th", "HIT", lines, new Date());
p.addOneLine("Love HIT");
List<String> some = p.getSomeLines(1, 1);

```

答题区

3 针对方法getSomeLines(...), 利用你所学过的异常处理机制和防御式编程的知识, 从健壮性的角度对该方法的实现代码进行改造, 必要时可对方法的spec做出修改（但不能改变参数）。简要写出你的改造思路, 无需给出改造后的代码。

答题区

4 要对方法getSomeLines(...)进行基于等价类划分和边界值的黑盒测试。在JUnit代码中已提前构建好了一个Poem对象，内容如下。请在下表给出针对各种正常情况和不正常情况的测试用例，使之对该方法的spec有全面的覆盖。按需填写，每行一个。

Author:	HITFC
Title:	HIT's 100th Anniversary
Lines:	Harbin 2020 Love HIT Forever
Date:	2020-06-07

参数start	参数end	期望执行结果

5 针对修改后不存在表示泄露的代码，继续对其修改，使该ADT在多线程下使用时做到thread safe，且对性能影响最低。简要写出你的修改策略，无需给出修改后的全部代码。

答题区

三（13分）某个ADT的rep如下所示。

```
class Encryption {
    String ss;
    int[] cs;
}
```

该ADT的AF和RI有以下三种设计方案：

方案1

```
// AF(ss,cs) = 一个加密字符串es
//   (1) es.length() == ss.length()
//   (2) 对任意0<=j<es.length(), 如果j在cs中出现,
//       则es.charAt(j)='*'; 否则, es.charAt(j)=ss.charAt(j)
//   例如: ss="Hello", cs=[1,3], 那么es="H*1*o";
// RI:
//   cs中不包含重复的数, 且cs.length<=ss.length()
//   对任意的合法下标i, 0<=cs[i]<ss.length()
```

方案2

```
// AF(ss,cs) = 一个加密字符串es
//   es是对ss中的第j位连续重复cs[j]次得到的。如果cs[j]=0,
//   那么ss中的第j位不在es中出现。0<=j<ss.length()。
//   例如: ss="HIT", cs=[1,0,3], 则es="HTTT"
//   即第0位的'H'重复1次, 第1位的'I'不出现, 第2位的'T'重复3次
// RI:
//   ss.length() == cs.length
//   对任意的合法下标i, cs[i]>=0
```

方案3

```
// AF(ss,cs)= 一个加密字符串es
//     es是对ss中的第j位连续重复cs[j]次得到的。如果cs[j]=0,
//     那么ss中的第j位不在es中出现。0<=j<ss.length()。
// RI:
//     ss.length() == cs.length
//     对任意的合法下标i, cs[i]==0或1
```

1 (4分) 下表第1列给出了该ADT的ss和cs (R空间中的取值)，请给出其在不同的AF/RI方案下映射得到的A空间的值。若ss和cs取值不合法，可填写“非法rep”。

方案1:

AF("HITFC",[2,4])	
AF("HITFC",[1,0,1,0,1])	
AF("HITFC",[3,0,5,9,1])	

方案2:

AF("",[])	
AF("HITFC",[1,0,1,0,1])	
AF("HITFC",[3,0,5,1,1])	

方案3:

AF("HITFC",[1,3,1,1,0])	
AF("HITFC",[1,0,1,1,0])	

2 (4分) 该ADT有以下方法的实现，该方法返回加密后的字符串的长度。

```
public int getLength() {
    int length = 0;
    for (int i = 0; i < cs.length; i++)
        length += cs[i];
    return length;
}
```

那么该getLength()方法符合以上1/2/3三个AF/RI方案中的哪个(些)? _____

3 (5分) 考虑该ADT的构造函数如下所示:

```
public Encryption(String ss, int[] cs) {
    this.ss = ss;
    this.cs = cs;
}
```

在上述AF/RI的方案1、2、3下，该构造函数的spec会有很大差异。假设你已有了三种方案下该构造函数的spec，且各spec中均要求输入参数ss和cs要满足各自的RI。请比较这三个spec的pre-condition之间的强弱关系，并简要说明理由。

答题区

四(12分)某ADT的功能是随机选择诗中的若干行,以日志形式输出文本,其代码如下所示。

```
class Recorder {
    private static final NUMBER = 5; //随机选择的次数
    private static final Logger log = Logger.getLogger("Recorder");
    private Poem poem; //该类型参见第三题中的Poem

    public void recording() {
        poem = new Poem(...); //此处忽略了参数, 新创建一个poem对象
        log.addHandler(new ConsoleHandler());
        log.setLevel(Level.INFO);

        Random r = new Random();
        List<String> lines = poem.getAllLines();
        for (int i = 0; i < NUMBER; i++) {
            String line = lines.get(r.nextInt(lines.size()));
            if(line.length() >= 10) //长度超过10才可被日志输出
                log.info(line);
        }
    }
}
```

后续要扩展新功能: 在不同场景下, 诗歌对象(poem)可从不同的源头读入(例如磁盘文件、用户键盘输入、某网络地址等)而不只是在方法中新创建; 输出日志的渠道可从控制台输出扩展到其他输出(例如写入磁盘文件、写入某个网络地址等); 随机选择诗歌文本行的时候, 文本行的长度限制也可自由配置(而不是目前局限于10)。

请使用template method设计模式, 对上述代码进行改造, 以支持上述功能扩展。写出你进行代码改造的基本思路描述, 无需写出具体代码。

答题区

五(12分) 以下给出了名为Student的ADT及其代码。

```
class Student {
    private String name;
    private Map<String, Integer> scores = new HashMap<>();
    //AF: name为学生名字; scores的key为学习内容, value为所获成绩
    //RI: scores中的value值范围为[0,100]

    public Student(String name) {
        this.name = name;
    }

    //根据学生所获得的成绩, 计算总成绩
    public int getFinalScore() {
        int total = 0;
        for (String content : scores.keySet())
            total += scores.get(content);
        return total/scores.size();
    }
}
```

```
//具体的学习动作，并获得分数
public void getScore(String content) {
    System.out.println("我在通过听课学习" + content);
    int score = ...; //这里实现了具体听课学习的过程，返回所得到的成绩
    scores.put(content, score);
}

public static void main(String[] args) {
    Student s = new Student("张三");
    s.getScore("计算机系统");
    s.getScore("软件构造");
    System.out.println("总成绩: " + s.getFinalScore());
}
}
```

目前代码只支持听课学习（见getScore()方法第1-2行）。现在学校鼓励学生通过网上MOOC自学来获得分数，而MOOC自学过程与听课学习过程不同，目前的getScore()代码无法支持它，将来也无法扩展到新的学习方式。

为此，拟修改Student类的设计，通过delegation机制将getScore()中的学习动作委派给接口Study（代码见下方）的子类型，学生可自由选择听课学习或MOOC自学来获得成绩。

```
interface Study {
    /**
     * @param content 学习内容
     * @return 学习之后获得的分数，[0,100]范围内的整数
     */
    int study(String content);
}
```

请简要写出最恰当的delegation机制，对已有代码做出修改，使之可支持这两种学习方式，并可灵活扩展其他学习方式（例如通过竞赛获得成绩、通过与导师做科研获得成绩等）。

答题区

六（12分）分析以下ADT的代码，判断两个类是否符合LSP。若违反，请逐条说明理由。

```
class Graph<L> {
    protected Set<L> nodes = new HashSet<>();
    protected Map<L, Set<L>> edges = new HashMap<>();
    // AF: 表示一个有向图，nodes是图的节点集合；edges中的key表示节点，
    //      value为从该key节点出发的边所指向的节点集合；L为节点类型的泛型参数
    // RI: edges的每个key，均在nodes中出现；
    //      edges的每个value中所包含的所有节点，均在nodes中出现
    //      对edges中的一个<key, value>，key不能在value中出现
    /**
     * @param nodes 一组节点
     * @param filter 过滤条件
     * @return nodes中满足filter条件的节点的数目
     */
}
```

```
    */
    public Number statistics (List<L> nodes, String filter) {
        ...
    }
}
class LoopWeightedGraph<L> extends Graph<L> {
    private Map<L, Integer> weights = new HashMap<>();
    // AF: weights中的key表示节点, value表示该key节点的权重
    // RI: edges的每个key, 均在nodes中出现;
    //     edges的每个value中所包含的所有节点, 均在nodes中出现
    //     weights包含的元素数量 == nodes包含的元素数量

    /**
     * @param nodes 一组节点
     * @param filter 过滤条件, 长度不超过20
     * @return nodes中满足filter条件的节点的数目
     * @throws 若nodes中存在重复节点
     */
    @Override
    public Integer statistics(ArrayList<L> nodes, String filter)
        throws Exception {
        ...
    }

    public int getTotalWeight() {...}
}
```

答题区

七（8分，每小题2分）填空题

- 1 请写出软件构造中的至少三个外部质量属性_____
- 2 在版本1中，有两个文件f1和f2；在版本2中，f1相对于版本1发生了变化，f2没变化。请简要阐述Git如何存储版本1和版本2？_____
- 3 在当前commit上建立新分支314change并切换到该分支上工作，使用的Git指令是_____（可以用1个指令，也可用多个指令）
- 4 某Java程序需要用户输入密码进行登录，程序员在代码中使用一个char[]类型的变量pw存储密码而不使用String类型。从安全性的角度，使用String类型会比使用char[]的风险要高，请简要给出至少一个理由_____