**《软件测试》**

**实验报告三 ——边等价类分析方法**

**姓 名： 汪成肸 学 号： 2020105002**   **院 系： 计算机与信息学院 专 业：计算机科学与技术**

**实 验 室： 实验日期： 2023/03/27**

**总评成绩： 审阅教师：**

### 一、实验目的：

1. 练习基本的编程能力；
2. 掌握边等价类分析法。
3. 巩固 git 协作式管理工具的使用；
4. 巩固 gradle 项目构建 Java 应用的基本操作

### 二、实验环境：

IntelliJ IDEA 2022.2.2

### 三、实验要求：

（1）实现电话账单收费功能

(2) 采用边等价类分析方法设计测试用例，并采用Junit5 编写电话账单收费程序测试用例，并提交到代码仓库

四、实验步骤与内容

**1.需求：电话账单计费**

对于电话账单来说，当春季和秋季标准时间与夏令时时间进行转换时会产生一个很有意思的问题: 春季，这种转换发生在(3月某个)星期日凌晨2:00点，这时要将时钟设置为凌晨3:00点；秋季， 转换通常在11月的第一个星期日，时钟要从2:59:59调回2:00:00。

请为长途电话服务函数开发计费类，使用等价类分析方法构建测试用例并使用Junit5测试。

**采用如下计费规则计算通话费：**

通话时间小于等于20分钟时，每分钟收费0.05美元，通话时间不足1分钟按1分钟计算。

通话时间大于20分钟时，收费1.00美元，外加超过20分钟的部分每分钟0.10美元；

不到1分钟按1分钟计算

**假设：**

通话计费时间从被叫方应答开始计算，到呼叫方挂机时结束；

通话时间的秒数向上进位到分钟；

没有超过30个小时的通话。

**2.测试用例：**

**等价类**是指某个输入域的子集合。在该子集合中,各个输入数据对于揭露程序中的错误都是等效的，并合理地假定：测试某等价类的代表值就等于对这一类其它值的测试。因此,可以把全部输入数据合理划分为若干等价类,在每一个等价类中取一个数据作为测试的输入条件，就可以用少量代表性的测试数据取得较好的测试结果。

等价类划分可有两种不同的情况：有效等价类和无效等价类。

1)有效等价类

是指对于程序的规格说明来说是合理的、有意义的输入数据构成的集合。利用有效等价类可检验程序是否实现了规格说明中所规定的功能和性能。

2)无效等价类

与有效等价类的定义恰巧相反。无效等价类指对程序的规格说明是不合理的或无意义的输入数据所构成的集合。对于具体的问题，无效等价类至少应有一个，也可能有多个。

**边界值分析法**

如果输入条件规定了值的范围，则应取刚达到这个范围的边界的值，以及刚刚超越这个范围边界的值作为测试输入数据

如果输入条件规定了值的个数，则用最大个数、最小个数、比最小个数少1、比最大个数多1的数作为测试数据

分析规格说明，找出其他可能的边界条件。

**测试用例设计**

有效等级类：  
M1={分钟：分钟=0}  
M2={分钟：分钟<20}  
M3={分钟：20≤分钟≤60\*20}  
S1={秒： 0≤秒<30}  
S2={秒： 秒=30}  
S3={秒：30<秒<60}  
B1={跨越：春季时间转换}  
B2={跨越：秋季时间转换}  
B3={跨越：无时间转换}  
无效等价类有：  
M4={分钟：分钟<0}  
M5={分钟：分钟>60\*20}  
S3={秒： 秒<0}  
S4={秒： 秒≥60}

**3.代码实现：**

|  |
| --- |
| **实现代码：**  import java.util.Calendar;  public class TelephoneBill {   static class TelephoneBillException extends RuntimeException {  public TelephoneBillException(String msg) {  super(msg);  }  }  //通话开始时间  private Calendar startTime;  //通话结束时间  private Calendar endTime;  //通话中是否发生春节的时间转换  private boolean springConversion;  //通话中是否发生秋季的时间转换  private boolean autumnConversion;   public TelephoneBill(Calendar startTime, Calendar endTime, boolean springConversion, boolean autumnConversion) {  this.startTime = startTime;  this.endTime = endTime;  this.springConversion = springConversion;  this.autumnConversion = autumnConversion;  }  private long calculateTalkTimeBySeconds() {  if (springConversion == true && autumnConversion == true) {  throw new TelephoneBillException("输入参数错误");  }  int seconds = (endTime.get(Calendar.*DAY\_OF\_MONTH*) - startTime.get(Calendar.*DAY\_OF\_MONTH*)) \* 24 \* 60 \* 60 +  (endTime.get(Calendar.*HOUR\_OF\_DAY*) - startTime.get(Calendar.*HOUR\_OF\_DAY*)) \* 60 \* 60 +  (endTime.get(Calendar.*MINUTE*) - startTime.get(Calendar.*MINUTE*)) \* 60 +  (endTime.get(Calendar.*SECOND*) - startTime.get(Calendar.*SECOND*));  if (seconds <= 0) {  throw new TelephoneBillException("输入参数错误");  }  //春季，这种转换发生在(3月某个)星期日凌晨2:00点，这时要将时钟设置为凌晨3:00点  //春节发生转换时，seconds会多算1个小时  seconds -= springConversion ? 60 \* 60 \* 60 : 0;  //秋季， 转换通常在11月的第一个星期日，时钟要从2:59:59调回2:00:00。  //秋季发生转换时，seconds会少算1个小时  seconds += autumnConversion ? 60 \* 60 \* 60 : 0;  return seconds;  }  private long calcuateTalkTimeByMinutes() {  long seconds = calculateTalkTimeBySeconds();  long minutes = (long) Math.*ceil*(seconds \* 1.0 / 60);  return minutes;  }   */\*\*  \* 通话时间小于等于20分钟时，每分钟收费0.05美元，通话时间不足1分钟按1分钟计算。  \* 通话时间大于20分钟时，收费1.00美元，外加超过20分钟的部分每分钟0.10美元；  \* 不到1分钟按1分钟计算  \*/* public double calcuateTelephoneBill() {  long minutes = calcuateTalkTimeByMinutes();  if (minutes > 30 \* 60) {  throw new TelephoneBillException("通话时间超过30小时");  }  double telephoneBill = 0.0;  for (int i = 0; i < 20; i ++) {  minutes --;  if (minutes >= 0) {  telephoneBill += 0.05;  } else {  return telephoneBill;  }  }  if (minutes > 0) {  telephoneBill += 1.0;  }  while (minutes -- > 0) {  telephoneBill += 0.1;  }  return Double.*parseDouble*(String.*format*("%.2f", telephoneBill));  } }  测试代码：  import org.junit.jupiter.api.DisplayName; import org.junit.jupiter.params.ParameterizedTest; import org.junit.jupiter.params.provider.CsvFileSource;  import java.text.ParseException; import java.text.SimpleDateFormat; import java.util.Calendar;  import static org.junit.jupiter.api.Assertions.*assertEquals*;  public class TelephoneBillTest {   @DisplayName(value="话费边等价类测试")  @ParameterizedTest  @CsvFileSource(resources = "/话费边等价类测试.csv",numLinesToSkip =1)  public void test(String start, String end,  boolean springConversion, boolean autumnConversion, double expert) throws ParseException {   SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss");  Calendar startTime = Calendar.*getInstance*();  startTime.setTime(simpleDateFormat.parse(start));  Calendar endTime = Calendar.*getInstance*();  endTime.setTime(simpleDateFormat.parse(end));  TelephoneBill telephoneBillTest = new TelephoneBill(  startTime,  endTime,  springConversion,  autumnConversion  );  try {  *assertEquals*(expert, telephoneBillTest.calcuateTelephoneBill());  } catch (TelephoneBill.TelephoneBillException e) {  *assertEquals*("输入参数错误", e.getMessage());  }   } }  测试结果：      Checkstyle报告结果说明:一些变量应该添加final修饰符, 有一个多余的修饰符public，过多的常量，可以定义一个全局常量类（或者只用于计算电话费的常量类），便于后期修改。 |

### 五、结论分析与体会

通过这次实验，我掌握了如何使用等价类分析方法设计测试用例对程序进行测试。掌握了使用Gradle构建Java项目**，**使用checkstyle和jacoco插件生成静态代码检查报告和测试覆盖率报告。

### 六、仓库地址

https://github.com/shuaishuaixi03/software-testing/tree/main/project-0327/code