

课程报告

课程名称： 软件测试（外文教材）

课 号： 2212186

学 院： 计算机与信息安全学院

学 号： 2100300425

姓 名： 韦昆杰

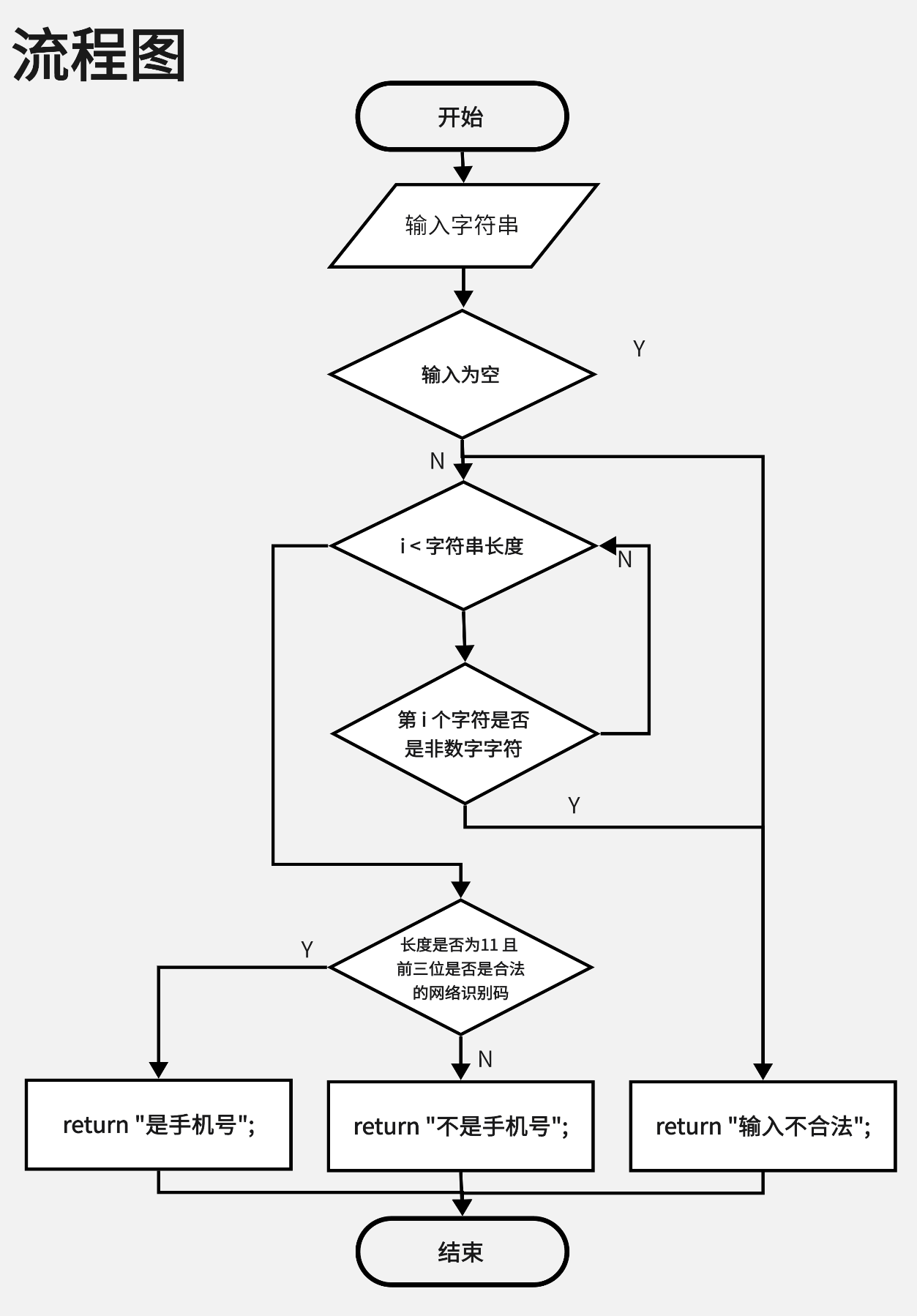
报告日期： 2022年 11 月 24 日

|  |  |  |
| --- | --- | --- |
| 项目 | 评价 | 得分 |
| 测试设计 | 黑盒法：□合理 □大致合理 □部分合理 □大部分不合理  白盒法：□合理 □大致合理 □部分合理 □大部分不合理  工具： □使用多种工具 □使用一种工具 □未使用工具 |  |
| 报告质量 | 内容：□完整 □较完整 □主体内容完整 □缺少必要内容  问题描述：□清晰详实 □较清晰 □一般 □主体描述不清  排版：□规范 □较规范 □少部分内容排版欠规范 □不规范  结论及分析：□详实 □较好 □一般 □缺少描述 |  |

1. **测试前提准备**
2. 测试模块：判断手机号函数
3. 测试环境：Windows 10、lntelliJ IDEA Community Edition 2021.2.1
4. 自动化测试工具：JUnit(4.12)
5. 测试方法：黑盒测试、白盒测试
6. 测试级别：单元测试

**2.测试的功能模块：判断手机号函数**

模块功能描述：输入11位数字的字符串，判断它是不是一个手机号码，如果是返回“是手机号”，如果不是返回“不是一个手机号”，输入有非数字字符或长度小于或大于11位时为非法输入。（***模块代码及自动化测试代码见附页***）

 流程图：

**3.黑盒测试**

**（1）等价类划分：**

|  |  |  |
| --- | --- | --- |
| **输入类型** | **有效等价类** | **无效等价类** |
| 是否为空输入 | (1)非空输入 | (2)空输入 |
| 是否全是数字 | (3)全是数字 | (4)有其它字符 |
| 长度是否为11 | (5)Length==11 | (6)Length<11  (7)Length>11 |

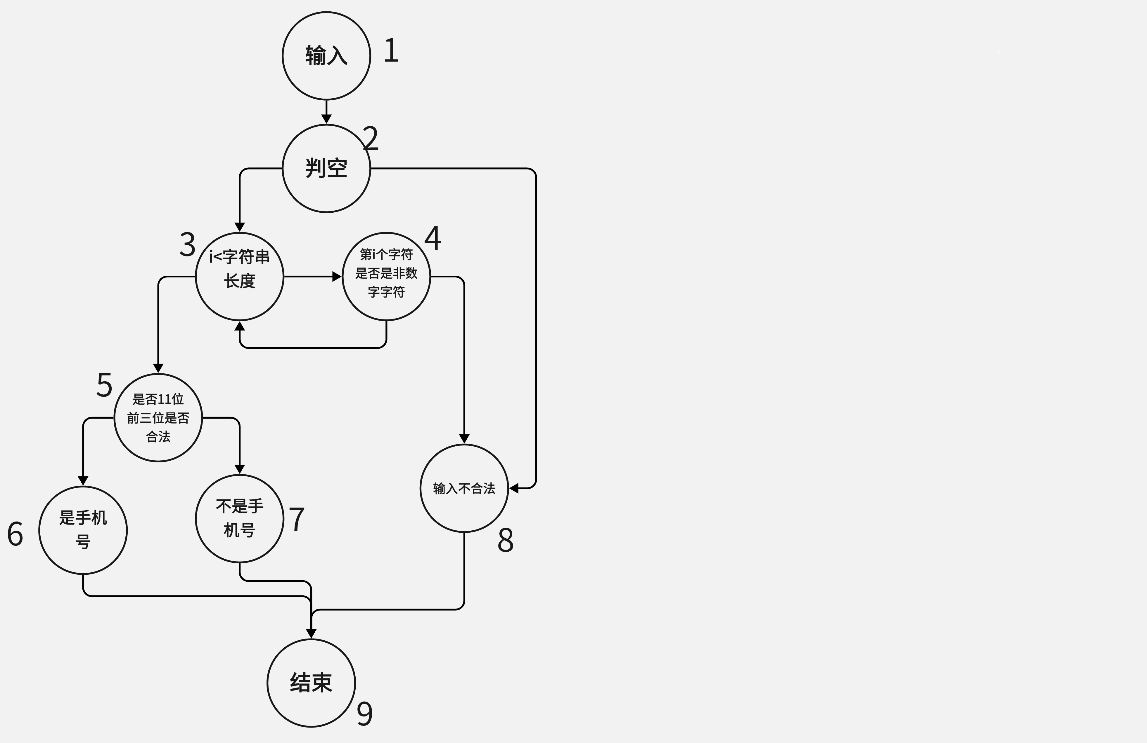
**（2）设计测试用例：**

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **测试输入(inputString)** | **覆盖等价类** | **预计输出** |
| 1 | “” | (2) | 输入不合法 |
| 2 | “191\*\*\*\*7925” | (4)(1) |
| 3 | “191” | (6)(1)(3) |
| 4 | “191666688889” | (7)(1)(3) |
| 5 | “19166668888” | (1)(3)(5) | 是手机号 |
| 6 | “12366668888” | (1)(3)(5) | 不是手机号 |

**（3）测试用例测试结果：**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **测试输入** | **预期输出** | **实际输出** | **是否符合预期** |
| 1 | “” | 输入不合法 | 输入不合法 | 符合 |
| 2 | “191\*\*\*\*7925” | 输入不合法 | 符合 |
| 3 | “191” | 不是手机号 | 不符合 |
| 4 | “191666688889” | 不是手机号 | 不符合 |
| 5 | “19166668888” | 是手机号 | 是手机号 | 符合 |
| 6 | “12366668888” | 不是手机号 | 不是手机号 | 符合 |

**4.白盒测试**

**(1)绘制程序流图：**

**(2)程序的环形复杂度**

V(G) = E–N+2 = 12–9+2 = 5

独立路径集合：

路径1：1-2-3-5-6-9

路径2：1-2-3-5-7-9

路径3：1-2-3-4-8-9

路径4：1-2-3-4-3-4-8-9

路径5：1-2-8-9

**(3)测试用例设计(路径覆盖)**

|  |  |  |
| --- | --- | --- |
| **序号** | **测试输入** | **覆盖路径** |
| 1 | “19166668888” | 1-2-3-5-6-9 |
| 2 | “12366668888” | 1-2-3-5-7-9 |
| 3 | “191\*\*\*7925” | 1-2-3-4-3-4-8-9 |
| 4 | “” | 1-2-8-9 |

**5.发现BUG及改进方案**

**(1)本次测试没出的BUG：**

对于输入长度小于或大于11的数字字符串的黑盒测试用例中，预期输出“输入不合法”，实际输出“不是手机号”。

**(2)改进方案**

对“手机号判断函数”中的“长度是否为11且前三位是否是合法的网络识别码”判断进行如下修改：

先判断长度是否为11位，

若长度是11，则进行“前三位是否是合法的网络识别码”的判断，前三位是是合法的网络识别码，则“是手机号”，否则“不是手机号”；

若长度不是11，则“输入不合法”。

改进后通过了所有的测试用例。

**6.小结**

本次测试使用了自动化测试工具JUnit，对用Java编写的“判断手机号”功能模块进行单元测试，分别用黑盒测试与白盒测试两种测试方法对功能模块进行测试，测出了功能模块的不足并给出了改进方案，改进后所有测试用例均可通过。

手机号的输入无论是在移动开发还是在Web前端开发中，都经常见到，该功能模块的可靠性由为重要。随着软件行业的不断发展，软件体积不断增加，软件测试在市场上已经成为软件开发不可缺少的重要组成部分。

**附页：**

**测试的功能模块代码(未改进前)：**

|  |
| --- |
| public class PhoneNumJudge {  //存放三大运营商号码的前3位——网络识别号  private final Map<String, String> networkIdNumMap;  /\*\*  \* 构造函数，初始化 networkIdNumMap  \*/  public PhoneNumJudge() {  networkIdNumMap = new HashMap<>();  //中国电信 133、149、153、173、177、180、181、189、191、199  networkIdNumMap.put("133", "中国电信");  networkIdNumMap.put("149", "中国电信");  networkIdNumMap.put("153", "中国电信");  networkIdNumMap.put("173", "中国电信");  networkIdNumMap.put("177", "中国电信");  networkIdNumMap.put("180", "中国电信");  networkIdNumMap.put("181", "中国电信");  networkIdNumMap.put("189", "中国电信");  networkIdNumMap.put("191", "中国电信");  networkIdNumMap.put("199", "中国电信");  //中国联通 130、131、132、145、155、156、166、171、175、176、185、186  networkIdNumMap.put("130", "中国联通");  networkIdNumMap.put("131", "中国联通");  networkIdNumMap.put("132", "中国联通");  networkIdNumMap.put("145", "中国联通");  networkIdNumMap.put("155", "中国联通");  networkIdNumMap.put("156", "中国联通");  networkIdNumMap.put("166", "中国联通");  networkIdNumMap.put("171", "中国联通");  networkIdNumMap.put("175", "中国联通");  networkIdNumMap.put("176", "中国联通");  networkIdNumMap.put("185", "中国联通");  networkIdNumMap.put("186", "中国联通");  //中国移动 134、135、136、137、138、139、147、150、151、152、157、158、159、182、183、187、188、178  networkIdNumMap.put("134", "中国移动");  networkIdNumMap.put("135", "中国移动");  networkIdNumMap.put("136", "中国移动");  networkIdNumMap.put("137", "中国移动");  networkIdNumMap.put("138", "中国移动");  networkIdNumMap.put("139", "中国移动");  networkIdNumMap.put("147", "中国移动");  networkIdNumMap.put("150", "中国移动");  networkIdNumMap.put("151", "中国移动");  networkIdNumMap.put("152", "中国移动");  networkIdNumMap.put("157", "中国移动");  networkIdNumMap.put("158", "中国移动");  networkIdNumMap.put("159", "中国移动");  networkIdNumMap.put("182", "中国移动");  networkIdNumMap.put("183", "中国移动");  networkIdNumMap.put("187", "中国移动");  networkIdNumMap.put("188", "中国移动");  networkIdNumMap.put("178", "中国移动");  }  /\*\*  \* 输入一个11位数字的字符串，  \* 判断它是不是一个手机号码，  \* 如果是返回“是手机号”  \* 如果不是返回“不是一个手机号”  \* 输入有非数字字符或长度小于或大于11位时为非法输入  \*  \* @param phoneNum 输入的字符串  \* @return 判断结果  \*/  public String judgePhoneNum(String phoneNum) {  //是否为空输入  if(phoneNum.isEmpty()) return "输入不合法";  //是否有数字以外的其它字符  for (int i = 0; i < phoneNum.length(); i++) {  if (phoneNum.charAt(i) < '0' || phoneNum.charAt(i) > '9')  return "输入不合法";  }  //长度是否为11且前三位网络识别码是否合法  if (phoneNum.length() == 11 && networkIdNumMap.containsKey(phoneNum.substring(0, 3))) {  return "是手机号";  } else {  return "不是手机号";  }  }  } |

**自动化测试代码：**

|  |
| --- |
| public class PhoneNumTest {  private PhoneNumJudge phoneNumJudge;  @Before  public void init() {  phoneNumJudge = new PhoneNumJudge();  }  /\*\*  \* 空输入测试  \*/  @Test  public void nullInputTest(){  Assert.assertEquals("输入不合法",phoneNumJudge.judgePhoneNum(""));  }  /\*\*  \* 有非数字字符输入测试  \*/  @Test  public void haveOtherCharsTest(){  Assert.assertEquals("输入不合法",phoneNumJudge.judgePhoneNum("191\*\*\*\*7925"));  }  /\*\*  \* 长度小于11畋测试  \*/  @Test  public void smallLengthInputTest(){  Assert.assertEquals("输入不合法",phoneNumJudge.judgePhoneNum("191"));  }  /\*\*  \* 长度大于11输入测试  \*/  @Test  public void bigLengthInputTest(){  Assert.assertEquals("输入不合法",phoneNumJudge.judgePhoneNum("191666688889"));  }  /\*\*  \* 11位数字  \* 是手机号测试  \*/  @Test  public void positiveInputTest() {  Assert.assertEquals("是手机号",phoneNumJudge.judgePhoneNum("19166668888"));  }  /\*\*  \* 11位数字  \* 但不是手机号测试  \*/  @Test  public void negativeInputTest() {  Assert.assertEquals("不是手机号",phoneNumJudge.judgePhoneNum("12366668888"));  }  } |

**使用了改进方案后的测试的功能模块代码：**

|  |
| --- |
| public class PhoneNumJudge {  //存放三大运营商号码的前3位——网络识别号  private final Map<String, String> networkIdNumMap;  /\*\*  \* 构造函数，初始化 networkIdNumMap  \*/  public PhoneNumJudge() {  networkIdNumMap = new HashMap<>();  //中国电信 133、149、153、173、177、180、181、189、191、199  networkIdNumMap.put("133", "中国电信");  networkIdNumMap.put("149", "中国电信");  networkIdNumMap.put("153", "中国电信");  networkIdNumMap.put("173", "中国电信");  networkIdNumMap.put("177", "中国电信");  networkIdNumMap.put("180", "中国电信");  networkIdNumMap.put("181", "中国电信");  networkIdNumMap.put("189", "中国电信");  networkIdNumMap.put("191", "中国电信");  networkIdNumMap.put("199", "中国电信");  //中国联通 130、131、132、145、155、156、166、171、175、176、185、186  networkIdNumMap.put("130", "中国联通");  networkIdNumMap.put("131", "中国联通");  networkIdNumMap.put("132", "中国联通");  networkIdNumMap.put("145", "中国联通");  networkIdNumMap.put("155", "中国联通");  networkIdNumMap.put("156", "中国联通");  networkIdNumMap.put("166", "中国联通");  networkIdNumMap.put("171", "中国联通");  networkIdNumMap.put("175", "中国联通");  networkIdNumMap.put("176", "中国联通");  networkIdNumMap.put("185", "中国联通");  networkIdNumMap.put("186", "中国联通");  //中国移动 134、135、136、137、138、139、147、150、151、152、157、158、159、182、183、187、188、178  networkIdNumMap.put("134", "中国移动");  networkIdNumMap.put("135", "中国移动");  networkIdNumMap.put("136", "中国移动");  networkIdNumMap.put("137", "中国移动");  networkIdNumMap.put("138", "中国移动");  networkIdNumMap.put("139", "中国移动");  networkIdNumMap.put("147", "中国移动");  networkIdNumMap.put("150", "中国移动");  networkIdNumMap.put("151", "中国移动");  networkIdNumMap.put("152", "中国移动");  networkIdNumMap.put("157", "中国移动");  networkIdNumMap.put("158", "中国移动");  networkIdNumMap.put("159", "中国移动");  networkIdNumMap.put("182", "中国移动");  networkIdNumMap.put("183", "中国移动");  networkIdNumMap.put("187", "中国移动");  networkIdNumMap.put("188", "中国移动");  networkIdNumMap.put("178", "中国移动");  }  /\*\*  \* 输入一个11位数字的字符串，  \* 判断它是不是一个手机号码，  \* 如果是返回“是手机号”  \* 如果不是返回“不是一个手机号”  \* 输入有非数字字符或长度小于或大于11位时为非法输入  \*  \* @param phoneNum 输入的字符串  \* @return 判断结果  \*/  public String judgePhoneNum(String phoneNum) {  //是否为空输入  if(phoneNum.isEmpty()) return "输入不合法";  //是否有数字以外的其它字符  for (int i = 0; i < phoneNum.length(); i++) {  if (phoneNum.charAt(i) < '0' || phoneNum.charAt(i) > '9')  return "输入不合法";  }  //长度是否为11(这里是用了改进方案的地方)  if(phoneNum.length() != 11) return "输入不合法";  //前三位网络识别码是否合法(这里是用了改进方案的地方)  if (networkIdNumMap.containsKey(phoneNum.substring(0, 3))) {  return "是手机号";  } else {  return "不是手机号";  }  }  } |