北京理工大学

智能时代的软件测试

3.1 等价类测试

刘辉 教授

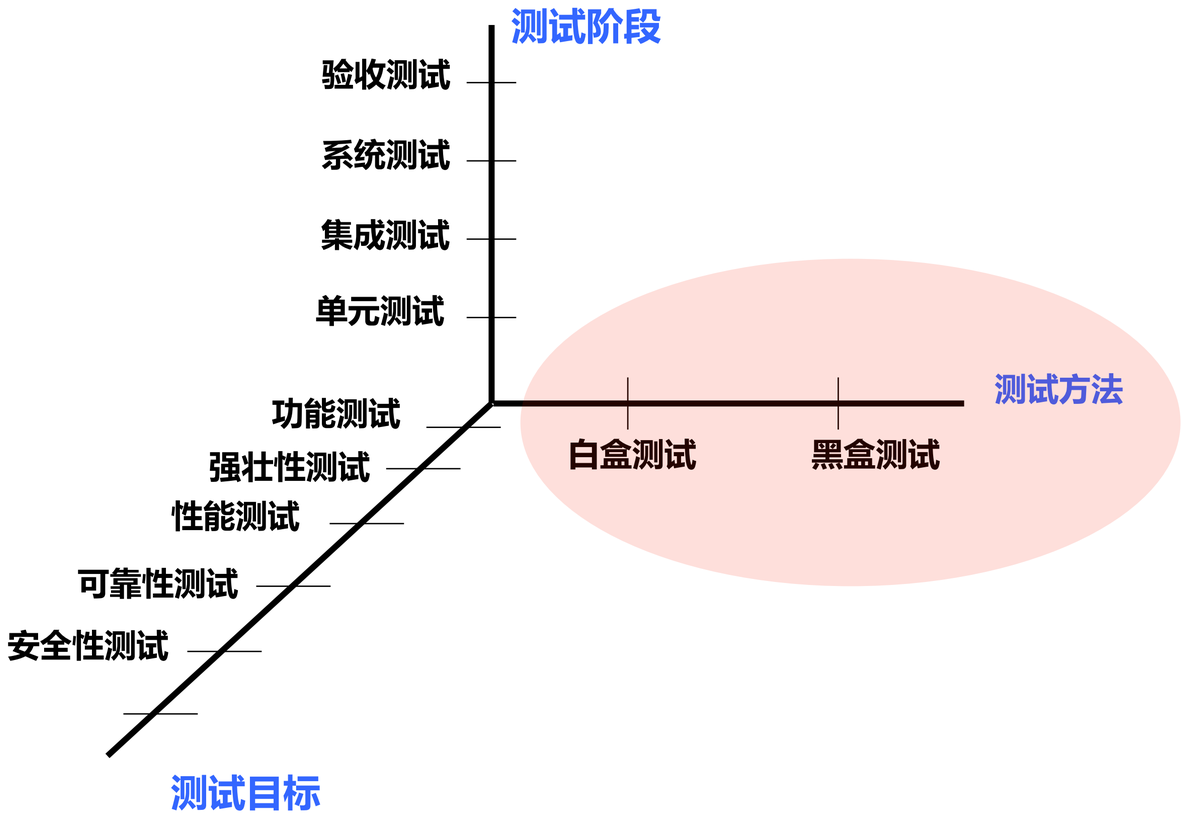
目录

CONTENTS

目录

CONTENTS

01 黑盒测试



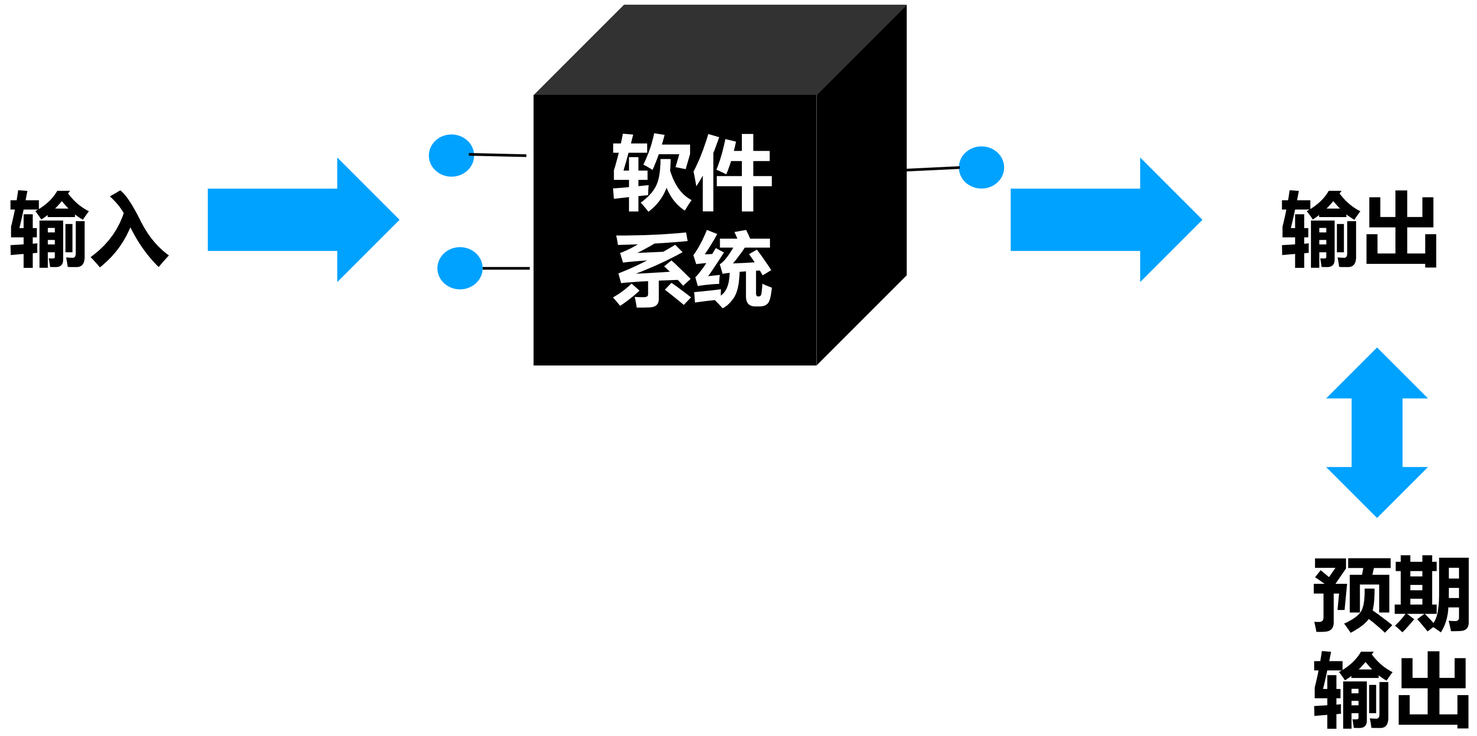
多维度分类：

测试方法

测试阶段

测试目标

01 黑盒测试



01

黑盒测试

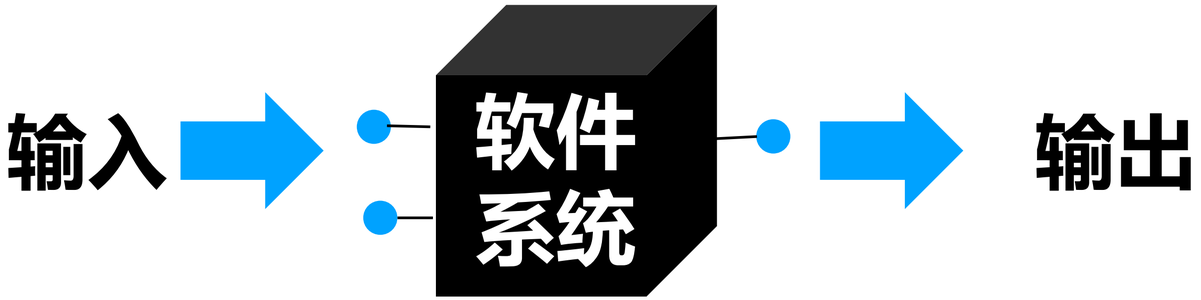
■ 黑盒测试（数据驱动测试）

- 它是把测试对象看做一个黑盒子，测试人员完全不考虑程序内部的逻辑结构和内部特性，只依据程序的需求规格说明书，检查程序的功能是否符合它的功能说明。

01 黑盒测试

■ 黑盒测试技术

- ➢ 等价类划分
- ➢ 边界值分析
- ➢ 输入组合法
- ➢ 因果图
- ➢ 基于状态测试



目录

CONTENTS

02

等价类测试

■ 等价类

- 如果软件的行为对于一组值来说是相同的，那么这组值就叫做等价类。
- 等价类是指测试相同目标或者暴露相同软件缺陷的一组测试用例。

02

等价类测试

■ 软件规约：

➢将输入的百分制成绩转换为A、B、C、D四档成绩。

- 90以上：A
- 80-90：B
- 70-80：C
- 70以下：D

```
void Grade(int score)
{
   if(score>100)
     {
       System.out.print("Out of scope");
       return;
      }
    if(score>90)
     {
       System.out.print("A");
       return;
      }
    if(score>80)
     {
       System.out.print("B");
       return;
      }
     if(score>70)
     {
       System.out.print("C");
       return;
      }
    System.out.print("D");
       return;
}
```

02

等价类测试

■ **软件规约：**

➢ 将输入的百分制成绩转换为A、B、C、D四档成绩。

- 90以上：A
- 80-90：B
- 70-80：C
- 70以下：D

■ **95，97，98**

■ **85，81，89**

■ **72，73，79**

■ **59，60，0**

02

等价类测试

- 95，97，98
- 85，81，89
- 72，73，79
- 59，60，0

```
void Grade(int score)
{
    if(score>100)
      {
        System.out.print("Out of scope");
        return;
       }
     if(score>90)
      {
        System.out.print("A");
        return;
       }
     if(score>80)
      {
        System.out.print("B");
        return;
       }
      if(score>70)
      {
        System.out.print("C");
        return;
       }
    System.out.print("D");
        return;
}
```

02

等价类测试

■ 软件规约：

➢依据成绩score计算奖励。

- 90以上：$4 \times score$
- 80-90：$2 \times score$
- 70-80：$score$
- 70以下：$0.5 \times score$

```
double Grade(int score)
{
  if(score>90)
      {
         return 4*score;
      }
     if(score>80)
      {
         return 2*score;
      }
    if(score>70)
     {
         return score;
      }
    return 0.5*score;
}
```

02

等价类测试

■ **软件规约：**

➢ 依据成绩score计算奖励。

- 90以上：$4 \times score$
- 80-90：$2 \times score$
- 70-80：$score$
- 70以下：$0.5 \times score$

■ **95，97，98**

■ **85，81，89**

■ **72，73，79**

■ **59，60，0**

02

等价类测试

- 95，97，98
- 85，81，89
- 72，73，79
- 59，60，0

```
double Grade(int score)
{
  if(score>90)
      {
          return 4*score;
      }
     if(score>80)
      {
          return 2*score;
      }
    if(score>70)
     {
          return score;
      }
    return 0.5*score;
}
```

02

等价类测试

■ 等价类划分方法

- 把所有可能的输入数据划分成若干部分（等价类），然后从每一个等价类中选取少数有代表性的数据作为测试用例。

02

等价类测试

■ 测试用例选择

- 设计一个测试用例，尽可能多地覆盖尚未覆盖的有效等价类。重复这个步骤直到覆盖所有有效等价类为止；
- 设计一个测试用例，尽可能少地覆盖尚未被覆盖的无效等价类（大于等于一）。重复这个步骤，直到所有无效等价类都被覆盖为止。

02

等价类测试

■ 软件规约：

➢ 将输入的百分制成绩转换为A、B、C、D四档成绩。

- 90以上：A
- 80-90：B
- 70-80：C
- 70以下：D

02

等价类测试

■ 有效等价类

- [90,100]
- [80,90)
- [70,80)
- [0,70)

■ 软件规约：

- 将输入的百分制成绩转换为A、B、C、D四档成绩。
  - 90以上：A
  - 80-90：B
  - 70-80：C
  - 70以下：D

02

等价类测试

■ 无效等价类

- (100,+∝]
- [-∝, 0)

■ 软件规约：

- 将输入的百分制成绩转换为A、B、C、D四档成绩。
  - 90以上：A
  - 80-90：B
  - 70-80：C
  - 70以下：D

02

等价类测试

■ 有效等价类

- [90,100]
- [80,90)
- [70,80)
- [0,70)

■ 无效等价类

- (100,+∝]
- [-∝, 0)

02

等价类测试

■ 测试用例选择

➢设计一个测试用例，尽可能多地覆盖尚未覆盖的有效等价类。重复这个步骤直到覆盖所有有效等价类为止；

■ **有效等价类**

- [90,100]
- [80,90)
- [70,80)
- [0,70)

■ **无效等价类**

- (100,+∝]
- [-∝, 0)

02

等价类测试

■ 测试用例

- A: 95
- B: 85
- C: 75
- D: 65

■ 有效等价类

- A: [90,100]
- B: [80,90)
- C: [70,80)
- D: [0,70)

■ 无效等价类

- E: (100,+∝]
- F: [-∝, 0)

■ 测试用例选择

- 设计一个测试用例，尽可能多地覆盖尚未覆盖的有效等价类。重复这个步骤直到覆盖所有有效等价类为止。

02

等价类测试

■ 测试用例

- A: 95
- B: 85
- C: 75
- D: 65

■ 测试用例

- E: 101
- F: -2

■ 有效等价类

- A: [90,100]
- B: [80,90)
- C: [70,80)
- D: [0,70)

■ 无效等价类

- E: (100,+∝]
- F: [-∝, 0)

■ 测试用例选择

- 设计一个测试用例，尽可能少地覆盖尚未被覆盖的无效等价类（大于等于一）。重复这个步骤，直到所有无效等价类都被覆盖为止。

目录

CONTENTS

03

等价类测试案例

■ **软件规约：**

➢ 根据年龄、性别、婚姻状况数等计算绩点。

- 根据绩点进行退税

年龄	20～39岁	10点
	40～59岁	4点
	60岁以上20岁以下	2点
性别	男	6点
	女	7点
婚姻	已婚	8点
	未婚	3点

03 等价类测试案例

1.年龄	数字范围	1～150
	有效等价类	20～39岁
		40～59岁
		60岁以上，150岁以下
		20岁以下，1岁以上
	无效等价类	1岁以下
		150岁以上

2.性别	有效等价类	男
		女
	无效等价类	非「男」或「女」
3.婚姻	有效等价类	已婚
		未婚
	无效等价类	非「已婚」或「未婚」

03 等价类测试案例

■ 测试用例

- 年龄=20,性别=男，婚姻=已婚
- 年龄=40,性别=女，婚姻=未婚
- 年龄=60,性别=女，婚姻=未婚
- 年龄=2，性别=女，婚姻=未婚

1.年龄	数字范围	1～150
	有效等价类	20～39岁 ✓
		40～59岁 ✓
		60岁以上，150岁以下 ✓
		20岁以下，1岁以上 ✓
	无效等价类	1岁以下
		150岁以上

2.性别	有效等价类	男 ✓
		女 ✓
	无效等价类	非「男」或「女」
3.婚姻	有效等价类	已婚 ✓
		未婚 ✓
	无效等价类	非「已婚」或「未婚」

■ 测试用例选择

- 设计一个测试用例，尽可能多地覆盖尚未覆盖的有效等价类。重复这个步骤直到覆盖所有有效等价类为止。

03 等价类测试案例

■ 测试用例

- 年龄=0,性别=男，婚姻=未婚
- 年龄=160,性别=女，婚姻=未婚
- 年龄=60,性别=人妖，婚姻=未婚
- 年龄=2，性别=女，婚姻=单身

1.年龄	数字范围	1～150
	有效等价类	20～39岁
		40～59岁
		60岁以上，150岁以下
		20岁以下，1岁以上
	无效等价类	1岁以下
		150岁以上

2.性别	有效等价类	男
		女
	无效等价类	非「男」或「女」
3.婚姻	有效等价类	已婚
		未婚
	无效等价类	非「已婚」或「未婚」

■ 测试用例选择

- 设计一个测试用例，尽可能少地覆盖尚未被覆盖的无效等价类（大于等于一）。重复这个步骤，直到所有无效等价类都被覆盖为止。

03 等价类测试案例

■ **测试用例 （覆盖有效等价类）**

- 年龄=20,性别=男，婚姻=已婚
- 年龄=40,性别=女，婚姻=未婚
- 年龄=60,性别=女，婚姻=未婚
- 年龄=2，性别=女，婚姻=未婚

■ **测试用例 （覆盖无效等价类）**

- 年龄=0,性别=男，婚姻=未婚
- 年龄=160,性别=女，婚姻=未婚
- 年龄=60,性别=人妖，婚姻=未婚
- 年龄=2，性别=女，婚姻=单身

1.年龄	数字范围	1～150
	有效等价类	20～39岁
		40～59岁
		60岁以上，150岁以下
		20岁以下，1岁以上
	无效等价类	1岁以下
		150岁以上

2.性别	有效等价类	男
	无效等价类	非「男」或「女」
3.婚姻	有效等价类	已婚
		未婚
	无效等价类	非「已婚」或「未婚」

目录

CONTENTS

04 小结

等价类测试是黑盒测试的重要手段

软件的行为对于一组值来说是相同的，
那么这组值就叫做等价类。

等价类测试方法就是把所有输入划分了若干等价类，
从每个等价类中选择若干典型输入作为测试用例。

谢谢

https://liuhuigmail.github.io/