

Ch1 软件测试基础

一、判断是非并改正错误

1. 某用户发现电商平台 A 对其同一个订单进行了多次扣款，这是软件缺陷。(×)
……这是软件失效
2. 缺陷是存在于软件中的不希望或不可接受的偏差，一定会导致软件失效。(×)
……，如果在运行中缺陷被触发，产生了故障并且故障被传播到可见的外部输出，则一定会导致软件失效
3. 调试的主要目的是排错，测试的主要目的是揭错。(√)
4. 各类测试活动需要等到代码编写完毕才能够展开。(×)
各类测试活动的测试需求分析，测试设计等活动可以先于代码实现展开
- 故障 5. 缺陷是软件运行中出现的不希望或不可接受的偏差。(×)
故障是……存在
缺陷是软件中的不希望或不可接受的偏差（说明：缺陷是静态的，存在于软件中的）
6. 达到期望的覆盖度要求是软件测试的主要目标。(×)
揭示缺陷实现缺陷预防是软件测试的最终目标。
达到期望的覆盖度要求是软件测试的主要目标之一
7. 如果测试输入能够执行到包含缺陷的代码就可以发现该缺陷。(×)
如果测试输入能够执行到包含缺陷的代码且满足感染性和可传播性就可以发现该缺陷
8. 如果可以穷举被测系统的输入空间，那么软件测试可以说明软件的正确性。(√)
9. 失效是不希望或不可接受的外部行为，说明被测系统中一定存在缺陷。(√)
10. 软件测试的目的只是发现缺陷。(×)
软件测试的目的揭示缺陷，实现缺陷预防
11. 缺陷在软件运行期间才能出现。(×)
缺陷是存在于软件中的不希望或不可接受的偏差。
12. 软件测试过程按照被测对象规模由小到大的规则推进。(√)
13. 测试不能说明被测系统是正确。(√)
14. 缺陷是指系统表现出的与期望不符的外部行为。(×)
失效是指……
15. 软件测试是发现缺陷的研发活动，可以说明软件的正确性。(√)
软件测试是发现缺陷的研发活动，由于受到资源限制和不能穷尽软件的输入空间，软件

测试只能说明软件中存在缺陷。

16. 能够发现缺陷或者至少可能发现缺陷是指测试用例的有效性。(✓)

17. 软件测试是指通过执行被测对象发现其中缺陷的一种研发活动。(✓)

18. 软件测试的主要目的是通过缺陷发现保证软件质量达到既定的目标,同时通过对缺陷的分析和利用实现软件开发中的缺陷预防。(✓)

19. 死机是一种缺陷。(×)

死机是一种失效。

20. 只有代码才能使用静态测试方法。(×)

软件包括文档、代码等都可以使用静态测试方法。(说明: 静态测试方法是指不运行被测对象发现其中缺陷的方法,例如人工评审,人工代码走读,代码静态分析等都是静态测试方法。人工评审是常用的发现文档中缺陷的方法,人工代码走读和代码静态分析是常用的发现代码中缺陷的静态测试方法)

21. 测试用例的有效性是指测试用例随被测对象的变化容易被重复利用的程度。(×)

测试用例的可维护性是指……

二、单项选择(每题有且仅有一个答案)

1. 下面关于测试用例的说法,不正确的是(B)

- A. 测试用例是测试设计活动的输出结果,✓
- B. 测试用例是对被测系统的输入数据,预期
- C. 揭错能力是衡量测试用例质量的维度之一;✓
- D. 测试用例的可视化形式可以是文档化也可以是代码化✓

2. 下列说法正确的是(A)

- A. ✓系统产生故障,说明测试输入满足可达性;✓
- B. 系统产生故障,说明测试输入不满足可达性;
- C. ✓系统没有产生故障,说明测试输入满足可达性;
- D. ✓系统没有产生故障,说明测试输入不满足可达性;

3. 下面4个评价测试用例质量的属性维度中,(A)是指测试用例的揭错能力

- A. ✓有效性; B. 仿效性; C. 经济性; D. 可维护性;

4. 下面(A)不是缺陷

- A. ✓系统分析人员对用户需求的错误理解;
- B. \开发人员将“!=”写成“=”;

- C. 代码中遗漏语句结束符;
- D. 代码没有对调用返回值可能为 null 进行是否为 null 的判断;
5. 下面 (C) 必须包含在测试用例里
- A、测试用例标识; B、优先级; C、预期结果; D、测试说明;
6. 以破坏测试对象的正常行为为目的测试技术被称为 (B)
- A. 正向测试; B. 反向测试; C. 代码测试; D. 系统测试
7. 代码如图 1 所示, 其中 (C) 存在缺陷 (使用指向局部变量的指针, aux 是局部变量在栈中分配空间, 只有在函数内部有意义, 出了函数 aux 的地址为空)

```
A、 void func_ARG(int **pp, unsigned n)
{
    int aux;
    if (n == 1) {
        *pp = &aux;
    } else {
        *pp = (int *)malloc(n * sizeof(int));
    }
}
```

图 1

8. 代码如图 2 所示, 输入 (B) 能产生故障 (fault), 但不产生失效 (failure); 输入(A)既产生故障, 也产生失效

```
public boolean findInteger(int[] x, int y){
    if (x==null) return false;
    //正确代码为for (int i=0; i<=x.length;i++)
    for (int i=1; i<=x.length;i++){
        if (x[i]==y)
            return true;
    }
    return false;
}
```

图 2

- A. x={2,3,4,5}, y=2 B. x={2,3,4,5}, y=3 C. x= null, y=4> D. 不存在
9. 用于衡量测试用例发现缺陷可能性的属性是 (A)
- A、有效性; B、仿效性; C、经济性; D、可维护性;
10. 不考虑系统内部实现结构验证系统是否到达用户需求的测试方法是 (A)
- A、黑盒测试; B、白盒测试; C、功能测试; D、非功能测试

三、多项选择（每题有 2 个或 2 个以上答案）

1. 下面描述中 (BEGHI) 是缺陷
- A. 系统长时间没有响应; ~~失效~~
 - B. 代码中遗漏了相关系统特性的实现;
 - C. 系统的响应时间不满足用户需求; ~~X~~
 - D. 系统分析人员对用户需求的理解错误; ~~X~~
 - E. 工程师没有及时修订用户操作手册导致实际操作流程和手册中的指导步骤不符;
 - F. 移动应用程序闪退;
 - G. 对于 Java 代码而言, 在长生命周期的对象中引用短生命周期对象;
 - H. 代码中的死循环;
 - I. 用例规约 (use case specification) 中缺少对异常处理流程的说明 需求是不完整
 - J. 系统输出不正确的结果; ~~失效~~

2. 代码如图 3 所示, 对于输入 ids=""(空字符串)而言, 满足 (AB)

```
public boolean mc1(String ids){  
    if ( ids.length() == 0) // 这里有缺陷, 正确的代码为 : if (ids == null)  
        return false;  
    for (String id : ids.split( regex: ";" )){  
        if (id.trim().startsWith("5101"))  
            return true;  
    }  
    return false;  
}
```

图 3

- A. 可达性; B. 感染性; C. 传播性;
3. 下列说法正确的是 (ABC)
- A. 测试用例必须包含测试输入和预期结果两个部分;
 - B. 若测试用例不具备感染性, 则一定不能发现缺陷;
 - C. 需求定义不完整是缺陷;
 - D. 内存泄漏是缺陷; ~~X~~
4. 代码如图 4 所示, for 循环中 $i > 0$ 存在缺陷, 应为 $i \geq 0$, 则输入 (BCD) 不能发现该缺陷

```

public int findLast (int[] x, int y)
{
    //Effects: If x==null throw NullPointerException
    // else return the index of the last element
    // in x that equals y.
    // If no such element exists, return -1
    for (int i=x.length-1; i > 0; i--)
    {
        if (x[i] == y)
        {
            return i;
        }
    }
    return -1;
}

```

图 4

可发现

A. x={2,3,5}, y=2; B. x=null, y=4; C. x = {1,2,4}, y=4; D. x={1}, y=4;

四、问答题

1、代码如图 5 所示，请根据代码回答如下问题

- 1) Identify the defect (识别代码中的缺陷)
- 2) If possible, identify an input that does not execute the defect. (如果存在，给出一个不能执行到该缺陷的输入)
- 3) If possible, identify an input that executes the defect, but does not result in a fault. (如果存在，给出一个可以执行到该缺陷但不能产生故障的输入)
- 4) If possible, identify an input that results in a fault, but not a failure. (如果存在，给出一个可以产生故障但不产生失效的输入)

```

4      /**
5      * if arr is null throw NullPointerException
6      * else return the number of occurrences of 0 in arr
7      */
8      @
9      public int numZero(int[] arr) {
10         int count = 0;
11         for (int i = 1; i < arr.length; i++)
12             if (arr[i] == 0)
13                 count++;
14         return count;
15     }

```

图 5

- 1) 第 10 行 $i = 1$ 处为缺陷

- 2) 不存在, 不论 arr 输入什么, 都会执行 $i = 1$, 即便 arr 为 null, 注意空指针异常是在 $i < arr.length$ 处抛出的
- 3) 不存在, 不论 arr 输入什么, 都会执行 $i = 1$, 而执行完成 $i = 1$ 之后, i 的值为 1, 但是期望 i 的值为 0
- 4) 只要 arr 的第一个元素不是 0 的都可以, 例如 $arr = \{2, 0, 1, 7\}$

2. 代码如图 6 所示, 请根据代码回答如下问题

- 1) Identify the defect
- 2) If possible, identify an input that does not execute the defect.
- 3) If possible, identify an input that executes the defect, but does not result in a fault.
- 4) If possible, identify an input that results in a fault, but not a failure .
- 5) For the given test ($arr = \{0, 1, 0\}$, $expected = 2$), identify the first faulty state
(对于给定的测试用例($arr = \{0, 1, 0\}$, $expected = 2$), 给出代码运行时产生的第一个故障状态)

```
17  /**
18   * if arr is null throw NullPointerException
19   * else return the index of the last 0 in arr
20   * return -1 if 0 does not occur in arr
21   */
22  @ public int lastZero(int[] arr){
23      for( int i = 0; i < arr.length; i++)
24          if(arr[i] == 0)
25              return i;
26      return -1;
27  }
```

图 6

缺陷不唯一, 这个代码错得过分了, 与期望功能背道而驰!! 在开发中应严格避免!! 认真理解和实现代码需求非常重要!! 正常合格的程序员实现的代码功能虽包含缺陷但只是与期望的功能稍有差别。

- 1) for 语句构造不正确, 应该为 $\text{for} (int i = arr.length - 1; i \geq 0; i--)$
- 2) 不存在
- 3) 不存在, 因为执行的指令内容不一样! 期望执行的 $i \geq 0$ 而实际执行的是 $i < arr.length$;
- 4) Null, 空数组, 不含 0 或只含有一个 0 的数组都可以

- 5) 第一个错误状态为 ($i=0$, $arr=\{0,1,0\}$, $PC\ i<arr.length$), 期望的状态应该为($i=2$, $arr=\{0,1,0\}$, $i\geq 0$) (注意该程序有多处包含缺陷的语句, 因为要求第一个错误状态, 这里只是写了执行第一个包含缺陷语句之后 $int\ i=0$ 的程序内部状态)

3. Answer questions (a) - (d) for the mutant (line 8) in the *findVal()* method.

(Mutant: 变异体, 即故意置入代码中的缺陷)

```
1 //Effects: If numbers null throw NullPointerException
2 // else return LAST occurrence of val in numbers[]
3 // If val not in numbers[] return -1
4 public static int findVal (int numbers[], int val)
5 {
6     int findVal = -1;
7     for (int i=0; i<numbers.length; i++)
8     // for (int i=(0+1); i<numbers.length; i++) (mutant)
9     if (numbers [i] == val)
10         findVal = i;
11     return (findVal);
12 }
```

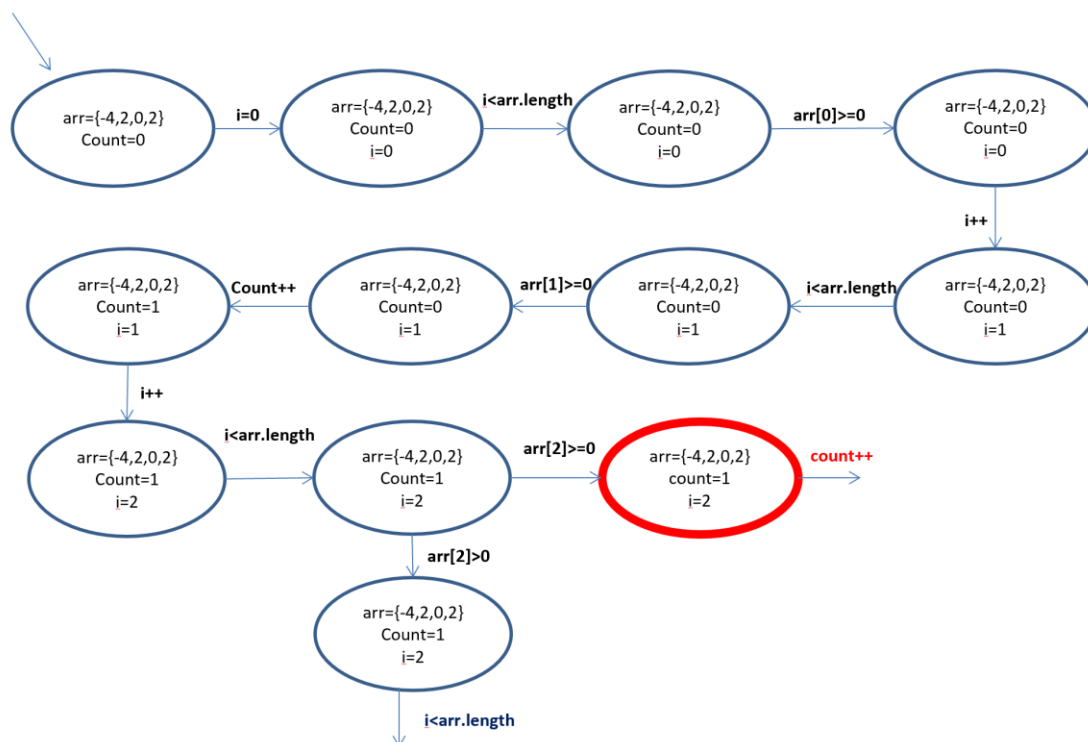
- a) If possible, find a test input that satisfies reachability but not infection for the Mutant. 不存在, 原因与第1题的第3小问是一样的
- b) If possible, find a test input that satisfies infection, but not propagation for the Mutant . 任意均可, 只要numbers不是null
- c) If possible, find a test input that strongly kills mutant m . numbers的第一个元素与val相同
- d) If possible, find a test input that weakly kills mutant m . 任意均可, 只要numbers不是null

4. Answer questions a)-e) for the following code segment

```
30  /**
31   *  if arr is null throw NullPointerException
32   *  else return number of positive elements in arr
33   */
34  @ public int countPositive(int[] arr){
35      int count = 0;
36      for( int i = 0; i < arr.length; i++)
37          if(arr[i] >= 0)
38              count++;
39      return count;
40  }
41
```

- a) Identify the defect.
- b) If possible, identify a test case that does not execute the defect.
- c) If possible, identify a test case that executes the defect, but does not result in a fault.
- d) If possible, identify a test case that results in a fault, but not a failure.
- e) For the given test case (`arr = {-4, 2, 0, 2}`, `expected = 2`), identify the first error state.
Be sure to describe the complete state.

- a) `arr[i] >= 0`
- b) `arr=null` 或者 `x={}`
- c) `arr` 的元素不含 0, 例如 `arr={1,-2}`
- d) 不存在
- e) 第一个错误状态: `({-4, 2, 0, 2}, count=1, i=2, PC at count++)` (因为正确的状态 PC 应该指向 `i<arr.length`), 分析过程如下



5. 指出下面 C/C++代码中的缺陷（提示：不止一个缺陷）

```

int main(int argc, char* argv[])
{
    int i, length;
    char *string1 = "Hello, Everyone!";
    char *string2 = (char *)malloc(10); 申请空间不足

    length = strlen(string2); 没有判断是否申请到空间，也没有对申请到的空间初始化
    printf("The initial length of string2 is:%d\n, The initial content is :%s\n", length, string2);

    for (i = 0; string1[i] != '\0'; i++) 越界读写
        string2[i] = string1[i];
}

length = strlen(string2);
printf("The end length of string2 is:%d\n, The initial content is :%s\n", length, string2);
return 0; 使用完空间没有释放
}

```