




# SOFTWARE TESTING

苏临之

[sulinzhi029@nwu.edu.cn](mailto:sulinzhi029@nwu.edu.cn)



# Four Basic Techniques

- Static Black-Box Testing
  - Dynamic Black-Box Testing
  - Static White-Box Testing
  - Dynamic White-Box Testing
- 



# Four Basic Techniques

- Static Black-Box Testing
- **Dynamic Black-Box Testing**
- Static White-Box Testing
- Dynamic White-Box Testing



# DBBT Techniques

- Equivalence Partitioning
- Boundary Value Analysis
- Decision Table
- Cause-Effect Diagram
- Error Guessing



# Limitations of the DBBT Techniques

- It is possible for the DBBT techniques to check whether there are bugs in software, whereas some specific kinds of bugs can not be detected (e.g. the bugs that are not actual errors).

MOV BX, 2000H

MOV AX, 1000H

NOP

MOV [BX], AX



# Another Example

- 某疑似普通话考试成绩查询网站的输入框需要输入姓名和对应身份证号，若两者匹配则显示成绩，若不匹配则显示输入错误。使用黑盒测试的几种方法，均检查无问题。
- 但是仔细检查后，发现查询系统源代码不是调用远程成绩数据库，而竟是几百条if-else组成的代码。这样暴露了几百位考生的身份信息。这是一种严重的软件的缺陷。



# Four Basic Techniques

- Static Black-Box Testing
- Dynamic Black-Box Testing
- **Static White-Box Testing**
- Dynamic White-Box Testing



# Static White-Box Testing

- Static white-box testing is the process of **carefully and methodically reviewing the software design, architecture, or code** without executing it.
- The obvious reason to perform static white-box testing is to find bugs early and to find bugs that would be difficult to uncover or isolate with dynamic black-box testing.





# Personnel and Process

## ❖ Personnel (at least four)

- 一人负责协调：分发材料、安排进程、修正检查
- 被测试程序的编码人员
- 程序的设计人员和一名测试专家

## ❖ Process

- 协调人在代码检查前几天分发程序清单和设计规范
- 编码人员讲述程序的逻辑结构，其他人员提问题并判断是否存在错误
- 对照历来常见的编码错误列表分析程序
- 注意力集中在发现错误而非纠正错误上（非调试）
- 会议结束后，程序员会得到一份已发现错误的清单



# Generic Code Review Checklist

- In the checklist, various errors that might be encountered in the coding process are included. The checklists are in addition to comparing the code against a standard or a guideline and to making sure that the code meets the project's design requirements.
- Of course, the tester should at least know something about programming because, otherwise, the list will become something full of rigid clauses.



# Contents in the Review Checklist

- The list generally contains the follow eight issues.
  1. Data Reference Errors (数据引用错误)
  2. Data Declaration Errors (数据声明错误)
  3. Computation Errors (运算错误)
  4. Comparison Errors (比较错误)
  5. Control Flow Errors (流程控制错误)
  6. Subroutine Parameter Errors (子程序参数接口错误)
  7. Input / Output Errors (输入输出错误)
  8. Other Checks (其余错误类型)



# Data Reference Errors

- 变量使用前是否赋值或初始化？
- 数组下标的范围和类型？
- 通过指针引用的内存单元是否存在（虚调用）？
- 被引用的变量或内存的属性是否与编译器预期的一致？



# Data Reference Errors

- 变量使用前是否赋值或初始化？
- 数组下标的范围和类型？
- 通过指针引用的内存单元是否存在（虚调用）？
- 被引用的变量或内存的属性是否与编译器预期的一致？

```
int a[10];  
for(i=1;i<=10;i++)  
    a[i]=i;
```



# Data Reference Errors

- 变量使用前是否赋值或初始化？
- 数组下标的范围和类型？
- 通过指针引用的内存单元是否存在（虚调用）？
- 被引用的变量或内存的属性是否与编译器预期的一致？

```
int a[10];  
for(i=1;i<=10;i++)  
    a[i]=i;
```

```
int a[10], i;  
for(i=0;i<=9;i++)  
    a[i]=i;
```



# Data Reference Errors

- 变量使用前是否赋值或初始化？
- 数组下标的范围和类型？
- 通过指针引用的内存单元是否存在（虚调用）？
- 被引用的变量或内存的属性是否与编译器预期的一致？

```
double a[10], i;  
for(i=0;i<=9;i++)  
    a[i]=1.0/(i+1);
```



# Data Reference Errors

- 变量使用前是否赋值或初始化？
- 数组下标的范围和类型？
- 通过指针引用的内存单元是否存在（虚调用）？
- 被引用的变量或内存的属性是否与编译器预期的一致？

```
double a[10], i;  
for(i=0;i<=9;i++)  
    a[i]=1.0/(i+1);
```

```
double a[10];  
int i;  
for(i=0;i<=9;i++)  
    a[i]=1.0/(i+1);
```





# Data Declaration Errors

- 是否所有变量都已声明？
- 默认的属性（默认值）是否正确？
- 变量的初始化是否正确？变量的初始化是否与其存储空间的类型一致？
- 是否每个变量都有正确的长度、类型和存储类别？
- 是否存在相似名称的变量？



# Data Declaration Errors

- 是否所有变量都已声明？
- 默认的属性（默认值）是否正确？
- 变量的初始化是否正确？变量的初始化是否与其存储空间类型一致？
- 是否每个变量都有正确的长度、类型和存储类别？
- 是否存在相似名称的变量？

```
int a[10];  
for(i=0;i<=9;i++)  
    a[i]=i;
```

```
int i, a[10];  
for(i=0;i<=9;i++)  
    a[i]=i;
```



# Data Declaration Errors

- 是否所有变量都已声明？
- 默认的属性（默认值）是否正确？
- 变量的初始化是否正确？变量的初始化是否与其存储空间类型一致？
- 是否每个变量都有正确的长度、类型和存储类别？
- 是否存在相似名称的变量？

```
int i=1, sum=0;  
while(sum<10)  
    sum=sum+1/i; i++;  
printf(“%d\n”, i);
```



# Data Declaration Errors

- 是否所有变量都已声明？
- 默认的属性（默认值）是否正确？
- 变量的初始化是否正确？变量的初始化是否与其存储空间类型一致？
- 是否每个变量都有正确的长度、类型和存储类别？
- 是否存在相似名称的变量？

```
int i=1, sum=0;  
while(sum<10)  
    sum=sum+1/i; i++;  
printf(“%d\n”, i);
```

```
int i=1;  
double sum=0;  
while(sum<10)  
    sum=sum+1.0/i; i++;  
printf(“%d\n”, i);
```



# Data Declaration Errors

- 是否所有变量都已声明？
- 默认的属性（默认值）是否正确？
- 变量的初始化是否正确？变量的初始化是否与其存储空间类型一致？
- 是否每个变量都有正确的长度、类型和存储类别？
- 是否存在相似名称的变量？

```
double pi=3.1416, F=10;
```

```
double r1=5, r2=7;
```

```
p1=F/(pi*r1*r1);
```

```
p2=F/(pi*r2*r2);
```



# Data Declaration Errors

- 是否所有变量都已声明？
- 默认的属性（默认值）是否正确？
- 变量的初始化是否正确？变量的初始化是否与其存储空间类型一致？
- 是否每个变量都有正确的长度、类型和存储类别？
- 是否存在相似名称的变量？

```
double pi=3.1416, F=10;  
double r1=5, r2=7;  
p1=F/(pi*r1*r1);  
p2=F/(pi*r2*r2);
```

```
double pai=3.1416, F=10;  
double r1=5, r2=7;  
p1=F/(pai*r1*r1);  
p2=F/(pai*r2*r2);
```



# Computation Errors

- 是否存在非算术变量之间的运算？
- 是否存在混合模式的运算？
- 是否存在不同字长变量之间的运算？
- 目标变量大小是否小于所赋值的大小？
- 中间结果是否上溢或下溢？
- 是否存在除以0错误？
- 操作符的优先顺序是否正确？
- 整数除法精度是否正确？



# Computation Errors

- 是否存在非算术变量之间的运算？
- 是否存在混合模式的运算？
- 是否存在不同字长变量之间的运算？
- 目标变量大小是否小于所赋值的大小？
- 中间结果是否上溢或下溢？
- 是否存在除以0错误？
- 操作符的优先顺序是否正确？
- 整数除法精度是否正确？

```
int i=1;  
float a=0.5;  
int x;  
x=i+a;
```





# Computation Errors

- 是否存在非算术变量之间的运算？
- 是否存在混合模式的运算？
- 是否存在不同字长变量之间的运算？
- 目标变量大小是否小于所赋值的大小？
- 中间结果是否上溢或下溢？
- 是否存在除以0错误？
- 操作符的优先顺序是否正确？
- 整数除法精度是否正确？

```
int i=1;  
float a=0.5;  
int x;  
x=i+a;
```

```
float i=1;  
float a=0.5;  
float x;  
x=i+a;
```



# Computation Errors

- 是否存在非算术变量之间的运算？
- 是否存在混合模式的运算？
- 是否存在不同字长变量之间的运算？
- 目标变量大小是否小于所赋值的大小？
- 中间结果是否上溢或下溢？
- 是否存在除以0错误？
- 操作符的优先顺序是否正确？
- 整数除法精度是否正确？

(8086CPU)

MOV AH, 90H

MOV AL, 90H

ADD AL, AH



# Computation Errors

- 是否存在非算术变量之间的运算？
- 是否存在混合模式的运算？
- 是否存在不同字长变量之间的运算？
- 目标变量大小是否小于所赋值的大小？
- 中间结果是否上溢或下溢？
- 是否存在除以0错误？
- 操作符的优先顺序是否正确？
- 整数除法精度是否正确？

(8086CPU)

MOV AH, 90H

MOV AL, 90H

ADD AL, AH

MOV AX, 0090H

MOV BX, 0090H

ADD AX, BX



# Computation Errors

- 是否存在非算术变量之间的运算？
- 是否存在混合模式的运算？
- 是否存在不同字长变量之间的运算？
- 目标变量大小是否小于所赋值的大小？
- 中间结果是否上溢或下溢？
- 是否存在除以0错误？
- 操作符的优先顺序是否正确？
- 整数除法精度是否正确？

```
double a[10];  
int i;  
for(i=0;i<=9;i++)  
    a[i]=1.0/i;
```



# Computation Errors

- 是否存在非算术变量之间的运算？
- 是否存在混合模式的运算？
- 是否存在不同字长变量之间的运算？
- 目标变量大小是否小于所赋值的大小？
- 中间结果是否上溢或下溢？
- 是否存在除以0错误？
- 操作符的优先顺序是否正确？
- 整数除法精度是否正确？

```
double a[10];  
int i;  
for(i=0;i<=9;i++)  
    a[i]=1.0/i;
```

```
double a[10];  
int i;  
for(i=0;i<=9;i++)  
    a[i]=1.0/(i+1);
```



# Comparison Errors

- 是否有不同类型数据的比较运算？
- 是否有混合模式或不同长度数据的比较运算？
- 比较运算符是否正确？
- 布尔表达式（与、或、非）是否正确？
- 比较运算符是否与布尔表达式相混合？
- 是否存在浮点数的比较？
- 优先顺序是否正确？
- 布尔表达式的计算方式



# Comparison Errors

- 是否有不同类型数据的比较运算？
- 是否有混合模式或不同长度数据的比较运算？
- 比较运算符是否正确？
- 布尔表达式（与、或、非）是否正确？
- 比较运算符是否与布尔表达式相混合？
- 是否存在浮点数的比较？
- 优先顺序是否正确？
- 布尔表达式的计算方式

- 欲输出数组：  
{0 1 2 3 4 4 3 2 1 0}

```
int i, a[10];  
for(i=0;i<=10;i++){  
    if i<=5  
        a[i]=i;  
    else  
        a[i]=10-i;  
}
```



# Comparison Errors

- 是否有不同类型数据的比较运算？
- 是否有混合模式或不同长度数据的比较运算？
- 比较运算符是否正确？
- 布尔表达式（与、或、非）是否正确？
- 比较运算符是否与布尔表达式相混合？
- 是否存在浮点数的比较？
- 优先顺序是否正确？
- 布尔表达式的计算方式

- 欲输出数组：  
{0 1 2 3 4 4 3 2 1 0}

```
int i, a[10];  
for(i=0;i<10;i++){  
    if i<5  
        a[i]=i;  
    else  
        a[i]=9-i;  
}
```





# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int x=1, flag=0;  
int sum=0;  
while (flag==0){  
    sum=sum+x;  
}
```



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int x=1, flag=0;
int sum=0;
while (flag==0){
    sum=sum+x;
}
```

```
int x=1, flag=0;
int sum=0;
while (flag==0){
    sum=sum+x;
    if sum>10000
        flag=1;
}
```



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i=1, x=100;  
int sum=0;  
while (sum>x){  
    sum=sum+i;  
    i++;  
}
```



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i=1, x=100;  
int sum=0;  
while (sum>x){  
    sum=sum+i;  
    i++;  
}
```

```
int i=1, x=100;  
int sum=0;  
while (sum<x){  
    sum=sum+i;  
    i++;  
}
```



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i=1, x=2;  
double sum=0;  
while (sum<x){  
    sum=sum+1/(i*i);  
    i++;  
}
```



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i=1, x=2;  
double sum=0;  
while (sum<x){  
    sum=sum+1/(i*i);  
    i++;  
}
```

```
int i=1, x=2;  
double sum=0;  
while (sum<x&& i<=10000){  
    sum=sum+1/(i*i);  
    i++;  
}
```

# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i=1, x=2;
double sum=0;
while (sum<x){
    sum=sum+1/(i*i);
    i++;
}
```

```
int i=1, x=2;
double sum=0;
while (sum<x&& i<=10000){
    sum=sum+1/(i*i);
    i++;
}
```

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} \approx 1.6$$





# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i;  
int sum=0;  
for(i=0;i<=100;i--)  
    sum=sum+i;
```



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i;  
int sum=0;  
for(i=0;i<=100;i--)  
    sum=sum+i;
```

```
int i;  
int sum=0;  
for(i=0;i>=-100;i--)  
    sum=sum+i;
```



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i;  
int sum=0;  
for(i=0;i<=-100;i--)  
    sum=sum+i;
```



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i;  
int sum=0;  
for(i=0;i<=-100;i--)  
    sum=sum+i;
```

```
int i;  
int sum=0;  
for(i=0;i>=-100;i--)  
    sum=sum+i;
```



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i;  
int sum=0;  
for(i=1;i<100;i++)  
    sum=sum+i;
```

- 欲求 $1+2+\dots+100$ 的值



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int i;  
int sum=0;  
for(i=1;i<100;i++)  
    sum=sum+i;
```

```
int i;  
int sum=0;  
for(i=1;i<=100;i++)  
    sum=sum+i;
```

- 欲求 $1+2+\dots+100$ 的值



# Control Flow Errors

- 是否所有循环都能终止?
- 是否存在由于入口条件不满足而跳过循环体?
- 是否存在仅差一个的循环错误?
- 程序结构中括号是否匹配? if、else和elseif是否匹配?

```
int x=1, flag=0, sum=0;
while (flag==0){
sum=sum+x;
if sum>10000{
flag=1;printf(“%d\n”, x);
}
```



# Control Flow Errors

- 是否所有循环都能终止？
- 是否存在由于入口条件不满足而跳过循环体？
- 是否存在仅差一个的循环错误？
- 程序结构中括号是否匹配？if、else和elseif是否匹配？

```
int x=1, flag=0, sum=0;
while (flag==0){
    sum=sum+x;
    if sum>10000{
        flag=1;printf(“%d\n”, x);
    }
```

```
int x=1, flag=0, sum=0;
while (flag==0){
    sum=sum+x;
    if sum>10000
        flag=1;printf(“%d\n”, x);
}
```





# Subroutine Parameter Errors

- 形参和实参的数量是否相等？
- 形参的属性是否与实参的属性相匹配？
- 形参的顺序是否与实参的顺序相匹配？
- 形参的单位是否和实参匹配？
- 是否改变了某个仅作为输入值的形参？
- 全局变量的定义是否一致？



# Subroutine Parameter Errors

- 形参和实参的数量是否相等？
- 形参的属性是否与实参的属性相匹配？
- 形参的顺序是否与实参的顺序相匹配？
- 形参的单位是否和实参匹配？
- 是否改变了某个仅作为输入值的形参？
- 全局变量的定义是否一致？

```
function [x,y]=rcsum(A)
```

---

```
U=[1,2;3,4];
```

```
a=rcsum(U);
```

- 子程序对矩阵分别对行对列求和



# Subroutine Parameter Errors

- 形参和实参的数量是否相等？
- 形参的属性是否与实参的属性相匹配？
- 形参的顺序是否与实参的顺序相匹配？
- 形参的单位是否和实参匹配？
- 是否改变了某个仅作为输入值的形参？
- 全局变量的定义是否一致？

```
function [x,y]=rcsum(A)
```

---

```
U=[1,2;3,4];  
a=rcsum(U);
```

```
function [x,y]=rcsum(A)
```

---

```
U=[1,2;3,4];  
[a,~]=rcsum(U);
```

- 子程序对矩阵分别对行对列求和



# Subroutine Parameter Errors

- 形参和实参的数量是否相等？
- 形参的属性是否与实参的属性相匹配？
- 形参的顺序是否与实参的顺序相匹配？
- 形参的单位是否和实参匹配？
- 是否改变了某个仅作为输入值的形参？
- 全局变量的定义是否一致？

```
function [c,U]=FCM(A, option)  
% 'A' is a vector, and 'option' is 4 × 1.  
w=[1,2,3,4,9,10,11,12];  
B=[2,100,2,0.0001];  
[c,U]=FCM(B, w);
```



# Subroutine Parameter Errors

- 形参和实参的数量是否相等？
- 形参的属性是否与实参的属性相匹配？
- 形参的顺序是否与实参的顺序相匹配？
- 形参的单位是否和实参匹配？
- 是否改变了某个仅作为输入值的形参？
- 全局变量的定义是否一致？

```
function [c,U]=FCM(A, option)
```

```
% 'A' is a vector, and 'option' is  $4 \times 1$ .
```

---

```
w=[1,2,3,4,9,10,11,12];
```

```
B=[2,100,2,0.0001];
```

```
[c,U]=FCM(B, w);
```

```
W=[1,2,3,4,9,10,11,12];
```

```
b=[2,100,2,0.0001]';
```

```
[c,U]=FCM(W, b);
```



# Subroutine Parameter Errors

- 形参和实参的数量是否相等？
- 形参的属性是否与实参的属性相匹配？
- 形参的顺序是否与实参的顺序相匹配？
- 形参的单位是否和实参匹配？
- 是否改变了某个仅作为输入值的形参？
- 全局变量的定义是否一致？

```
function [x1,x2]=f(a,b,c)
x1=(-b+sqrt(b*b-4*a*c))/(2*a);
x2=(-b-sqrt(b*b-4*a*c))/(2*a);
```



# Subroutine Parameter Errors

- 形参和实参的数量是否相等？
- 形参的属性是否与实参的属性相匹配？
- 形参的顺序是否与实参的顺序相匹配？
- 形参的单位是否和实参匹配？
- 是否改变了某个仅作为输入值的形参？
- 全局变量的定义是否一致？

```
function [x1,x2]=f(a,b,c)
x1=(-b+sqrt(b*b-4*a*c))/(2*a);
x2=(-b-sqrt(b*b-4*a*c))/(2*a);


---


p=1;q=2;r=-3; %px2+qx+r=0
[x1,x2]=f(a,b,c);
```



# Subroutine Parameter Errors

- 形参和实参的数量是否相等？
- 形参的属性是否与实参的属性相匹配？
- 形参的顺序是否与实参的顺序相匹配？
- 形参的单位是否和实参匹配？
- 是否改变了某个仅作为输入值的形参？
- 全局变量的定义是否一致？

```
function [x1,x2]=f(a,b,c)
x1=(-b+sqrt(b*b-4*a*c))/(2*a);
x2=(-b-sqrt(b*b-4*a*c))/(2*a);


---


p=1;q=2;r=-3; %px2+qx+r=0
[x1,x2]=f(a,b,c);
```

```
function [x1,x2]=f(a,b,c)
x1=(-b+sqrt(b*b-4*a*c))/(2*a);
x2=(-b-sqrt(b*b-4*a*c))/(2*a);


---


p=1;q=2;r=-3; %px2+qx+r=0
[x1,x2]=f(p,q,r);
```





# Input / Output Errors

- 文件属性是否正确？打开文件的语句是否正确？
- 缓冲区、内存大小是否足够来保留程序将读取的文件？
- 文件在使用前是否打开？
- 文件在使用后是否关闭了？
- 文件结束条件是否本正确处理？
- 是否处理了IO错误？
- 打印或输出的文本信息中是否存在拼写或语法错误？



# Input / Output Errors

- 文件属性是否正确？打开文件的语句是否正确？
- 缓冲区、内存大小是否足够来保留程序将读取的文件？
- 文件在使用前是否打开？
- 文件在使用后是否关闭了？
- 文件结束条件是否本正确处理？
- 是否处理了IO错误？
- 打印或输出的文本信息中是否存在拼写或语法错误？  
(从地址为20H的外设端口读一个字节到AH里)

MOV AH, [20H]



# Input / Output Errors

- 文件属性是否正确？打开文件的语句是否正确？
  - 缓冲区、内存大小是否足够来保留程序将读取的文件？
  - 文件在使用前是否打开？
  - 文件在使用后是否关闭了？
  - 文件结束条件是否本正确处理？
  - 是否处理了IO错误？
  - 打印或输出的文本信息中是否存在拼写或语法错误？
- （从地址为20H的外设端口读一个字节到AH里）

MOV AH, [20H]

IN AH, 20H



# Other Checks

- 是否存在未引用过的变量？
- 每个变量的属性和赋予的默认值是否一致？
- 编译通过的程序是否存在“警告”或“提示”信息？
- 程序或模块是否对输入的合法性进行了检查？
- 程序是否遗漏了某个功能？



# Other Checks

- 是否存在未引用过的变量？
- 每个变量的属性和赋予的默认值是否一致？
- 编译通过的程序是否存在“警告”或“提示”信息？
- 程序或模块是否对输入的合法性进行了检查？
- 程序是否遗漏了某个功能？

```
int i, j;  
double a[10];  
for(i=0;i<=9;i++){  
    a[i]=(i+1)/(i+2);  
    printf(“%f\n”,a[i]);  
}
```



# Other Checks

- 是否存在未引用过的变量？
- 每个变量的属性和赋予的默认值是否一致？
- 编译通过的程序是否存在“警告”或“提示”信息？
- 程序或模块是否对输入的合法性进行了检查？
- 程序是否遗漏了某个功能？

```
int i, j;  
double a[10];  
for(i=0;i<=9;i++){  
    a[i]=(i+1)/(i+2);  
    printf(“%f\n”,a[i]);  
}
```

```
int i;  
double a[10];  
for(i=0;i<=9;i++){  
    a[i]=(i+1)/(i+2);  
    printf(“%f\n”,a[i]);  
}
```



# Other Checks

- 是否存在未引用过的变量？
- 每个变量的属性和赋予的默认值是否一致？
- 编译通过的程序是否存在“警告”或“提示”信息？
- 程序或模块是否对输入的合法性进行了检查？
- 程序是否遗漏了某个功能？

```
int i, j, a[10][10]
for(i=0;i<=10;i++){
    for(j=0;j<=10;j++){
        a[i][j]=i+j;
    }
}
```



# Other Checks

- 是否存在未引用过的变量？
- 每个变量的属性和赋予的默认值是否一致？
- 编译通过的程序是否存在“警告”或“提示”信息？
- 程序或模块是否对输入的合法性进行了检查？
- 程序是否遗漏了某个功能？

```
int i, j, a[10][10]
for(i=0;i<=10;i++){
    for(j=0;j<=10;j++){
        a[i][j]=i+j;
    }
}
```

```
int i, j, a[10][10];
for(i=0;i<=9;i++){
    for(j=0;j<=9;j++){
        a[i][j]=i+j;
    }
}
```





# Walkthrough

- In a walkthrough (代码走查), the programmer who wrote the code formally presents (walks through) it to a small group of five or so other programmers and testers.
- The reviewers should receive copies of the software in advance of the review so they can examine it and write comments and questions that they want to ask at the review. Having at least one senior programmer as a reviewer is very important.
- The presenter reads through the code line by line, or function by function, explaining what the code does and why.
- The reviewers listen and question anything that looks suspicious.



# Walkthrough

- 如要求从1至正整数 $n$ 的累加和，程序员给出下面的代码。

```
#include <stdio.h>
int main(void){
    int i,n;
    int sum=0;
    scanf("%d",&n);
    for(i=1;i<n;i++)
        sum=sum+i;
    printf("The sum from 1 to n is %d.\n", sum);
}
```



# Walkthrough

- 现在进行走查操作，假设 $n=10$ ，则题目要求需要求1至10的累加和。

```
#include <stdio.h>
int main(void){
    int i,n; //假设n=10
    int sum=0;
    scanf("%d",&n);
    for(i=1;i<n;i++)
        sum=sum+i;
    printf("The sum from 1 to n is %d.\n", sum);
}
```



# Walkthrough

- 当 $i=9$ 时，执行完for循环，然后通过 $i$ 变为了10，再回到for循环里，已经不满足 $i < n=10$ 条件，因此循环终止。sum的值是1至9的累加和，不符合题目要求。

```
#include <stdio.h>
```

```
int main(void){
```

```
    int i,n; //假设n=10
```

```
    int sum=0;
```

```
    scanf("%d",&n);
```

```
    for(i=1;i<n;i++)
```

```
        sum=sum+i;
```

```
    printf("The sum from 1 to n is %d.\n", sum);
```

```
}
```



# Walkthrough

- 所以只能是 $i < n$ 这个条件出问题了。在此，可以把条件改为 $i \leq n$ 或 $i < n + 1$ 。这种把程序在人脑中按计算机的运算模式走一遍的过程就是走查。

```
#include <stdio.h>
```

```
int main(void){
```

```
    int i,n;
```

```
    int sum=0;
```

```
    scanf("%d",&n);
```

```
    for(i=1;i<n;i++)
```

```
        sum=sum+i;
```

```
    printf("The sum from 1 to n is %d.\n", sum);
```

```
}
```



# Desk Checking

- 桌面检查是一种古老的检查代码方式。这个过程中只有一个人来评审代码，评审方式可以根据上述对照错误列表一一检查，也可以进行代码走查，还可以把两者相结合，没有任何约束性。
- 桌面检查效率和检查效果远远低于小组式的检查和走查，缺乏合作带来的动力和相互促进。而且桌面检查者不宜是编写代码人员本身，应该由非参与编写程序的测试员担当。



**THANK YOU!**