提高程序质量的方法

一、常见的错误

1. 语法错误
2. 逻辑错误
3. 常见错误举例
   （1）在应该出现句号的地方，忘写了句号。
   （2）两个项之间忘写空格
   （3）环境部中的 SELECT 子句中的内部文件名写错
   （4）数据部中的 FD 后文件名写错
   （5）对文件节中的数据赋初值
   （6）程序书写格式错误
   （7）定义数据项时长度不够导致截断
   （8）关键字拼写错误
   ……

二、程序调试的方法

1. 适当的加入 DISPLAY 产生 TRACE 文件
2. 适当的加入 ACCEPT 检查输入的数据是否合理

三、说明部分(DECLARATIVE)和使用(USE)语句的使用

在 COBOL 程序的过程部中，可以使用“说明部分”的功能。在某一过程或错误发生时，执行“说明部分”中指出的过程。

其一般形式为：

```
PRODUCE DIVISION

DECLARATIVES

  节名 1.

     USE BEFORE|AFTER 触发事件

      语句序列 1

  节名 2.

     USE BEFORE|AFTER 触发事件

      语句序列 2

……

END DECLARATIVES
```

说明：1. 可以在说明部分中设若干个节，每一个节都要有一个节头，一个 USE 语句和一个相应的过程。

2. 只有在触发事件发生时才会执行说明部分中的过程

3. USE 语句本身是不执行的，它指定过程执行的条件

4. 说明部分中的过程不能涉及到任何说明部分以外的过程，也不能在各说明节中任意跳转

四、提高程序质量的方法

1. 不要写一个庞大的主程序
2. 采用模块化的程序结构
3. 提高程序的可读性
4. 减少输入输出次数
5. 节约内存
6. 尽量少使用 GO TO 语句

五、END 结尾字的使用

COBOL 对 ADD , READ , IF , PERFORM 等语句加上了 END 结尾字，使其结构更加完整，易读。

六、多分支选择语句(EVALUATE 语句)

EVALUATE 语句的一般形式：

```
EVALUATE { 标识符 1 }
         { 常量 1  }
         { 条件式 1 }

    WHEN  TRUE|FALSE   语句序列 1
    WHEN  OTHER        语句序列 1
    END-EVALUATE
```

说明： 判断主体的个数可以不止一个，但此时判断对象的个数应与之相同

七、对 PERFORM 语句的改进

由于 PERFORM 语句要执行的操作总是置于另外的节或段中，这样虽然有利于程序的模块化，但对于使用 PERFORM 语句实现简单的循环来说，却削弱了程序的可读性。所以 COBOL 对 PERFORM 语句进行了两点扩展

1. 将被执行部分变成了可选部分，并增加了可选项“语句序列 END-PERFORM” 和内置语句部分

例：
```
PERFORM
        VARYING ITEMC FROM 1 BY 2
        UNTIL ITEMC > 7
        MOVE  A(ITEMC) TO B(ITEMC)
END-PERFORM.
```

2. 增加了“TEST BEFOR|AFTER”可选项来实现“DO WHILE”和“DO UNTIL”型循环

例：
```
PERFORM WITH TEST AFTER I > 7
         SUBTRACT 1 FROM I
```

END-PERFORM.

与

```
DO UNTILE I <= 7
   SUBTRACT 1 FROM I
END-DO.
```

是等价的

例：

```
PERFORM WITH TEST BEFORE I > 7
          ADD 1 TO J
END-PERFORM.
```

与

```
DO WHILE I <= 7
   ADD 1 TO J
END-DO.
```

是等价的