

程序编码

董渊

(System Software & Software Engineering)

Department of Computer Science & Technology

Tsinghua University

内容提要

- 基本目标——可阅读性
- 程序设计风格
- **JAVA**编程风格
- 代码度量

“7. It is easier to write an incorrect program than understand a correct one.

7. 编写不正确的代码要比理解正确的代码简单得多。”

-- Alan Jay Perlis

“Epigramson in Programmimg”, 1982

The First Recipient of the A. M. Turing Award, 1966



For his influence in the area of
advanced programming techniques
and compiler construction.

基本目标——可阅读性

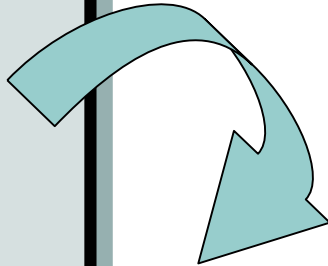
- 对软件开发人员
 - 思路清楚
 - 工作的连续性
 - 定位错误
 - 变更管理
- 对软件开发小组
 - 设计人员、开发人员、测试人员、维护人员
- 保证程序设计与实现相一致

程序设计风格 – 基本原则

- Top-down flow
 - Have the required action follow soon after the decision that generates it
- Generality is a virtue
 - Trade-off: generality, performance, understanding
- Dependence among components must be visible

```
benefit = minimum;  
if (age<75) goto A;  
benefit = maximum;  
goto C;  
if (age<65) goto B;  
if (age<55) goto C;  
A: if (age<65) goto B;  
   benefit = benefit * 1.5 + bonus;  
   goto C;  
B: if (age<55) goto C;  
   benefit = benefit * 1.5;  
C: next statement
```

```
benefit = minimum;  
if (age<75) goto A;  
benefit = maximum;  
goto C;  
if (age<65) goto B;  
if (age<55) goto C;  
A: if (age<65) goto B;  
   benefit = benefit * 1.5 + bonus;  
   goto C;  
B: if (age<55) goto C;  
   benefit = benefit * 1.5;  
C: next statement
```



```
if (age < 55) benefit = minimum;  
else if (age < 65) benefit = minimum + bonus;  
else if (age < 75) benefit = minimum * 1.5 + bonus;  
else benefit = maximum;
```

程序设计风格— 基本原则

- Do not sacrifice clarity and correctness for speed
 - Cost to write faster code
 - Cost to test more complex code
 - Cost to maintain more complex/less understandable code
 - Let compiler and other tools do it
- Choose data structure to simplify the program' s calculation

- 1. For the first \$10,000 of income, the tax is 10%.**
- 2. For the next \$10,000 of income above \$10,000, the tax is 12%.**
- 3. For the next \$10,000 of income above \$20,000, the tax is 15%.**
- 4. For the next \$10,000 of income above \$30,000, the tax is 18%.**
- 5. For any income above \$40,000, the tax is 20%.**

1. For the first \$10,000 of income, the tax is 10%.
2. For the next \$10,000 of income above \$10,000, the tax is 12%.
3. For the next \$10,000 of income above \$20,000, the tax is 15%.
4. For the next \$10,000 of income above \$30,000, the tax is 18%.
5. For any income above \$40,000, the tax is 20%.



Tax Table

Bracket	Base	Percent
0	0	10
10,000	1000	12
20,000	2200	15
30,000	3700	18
40,000	5500	20

$$\text{tax} = \text{base}[\text{level}] + \text{percent}[\text{level}] * (\text{taxable_income} - \text{bracket}[\text{level}])$$

程序设计风格—基本原则

- Localizing input and output
 - Hardware/software dependence
 - Easy to change
- Including pseudocode
 - Adapt design to chosen language
- Revising and rewriting, not patching
 - Module decomposition, control structure, data structure, algorithm, etc.

程序设计风格— 基本原则

- Documentation : written description of **WHAT** the programs do and **HOW** they do it
 - Self-Documentation, Source code Internal documentation
 - Header comment block
 - Program comments
 - Meaningful variable names and statement labels
 - Formatting to enhance understanding
 - Documenting data
 - External documentation
 - Problem, algorithm, data

Java程序设计风格

- 命名规则
 - 包（**Packages**），体系结构
 - 接口（**Interfaces**），联系（内外部）
 - 类（**Classes**），基本模块
 - 方法（**Methods**），服务，消息
 - 变量（**Variables**），属性
 - 常量（**Constants**），属性
- “一致性”

命名规则

```
/*  
 * Copyright 2012 Shuyang Ji  
 *  
 * This file is part of Connect6  
 *  
 * Connect6 is free software: you can redistribute it and/or modify  
 * it under the terms of the GNU General Public License as published by  
 * the Free Software Foundation, either version 3 of the License, or  
 * (at your option) any later version.  
 *  
 * Connect6 is distributed in the hope that it will be useful,  
 * but WITHOUT ANY WARRANTY; without even the implied warranty of  
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the  
 * GNU General Public License for more details.  
 *  
 * You should have received a copy of the GNU General Public License  
 * along with Connect6. If not, see <http://www.gnu.org/licenses/>.  
 */
```

● 包

- 前缀（顶层）
 - 小写ASCII字母
 - ISO 3166(1981)的Internet域名规范：
 - 如com, edu, org, net, gov, mil
 - 如cn, tw, hk, jp, uk
- 后继可根据项目结构：
 - 所在组织的组织结构
 - 或内部命名规范
- 例如：
 - com.sun.security;
 - org.omg.CORBA
 - cn.edu.tsinghua.course11.group4

package cn.edu.tsinghua.se2012.connect6;

命名规则

● 接口

- 名词或形容词
- 名词或形容词组合中，每词首字母大写
- 尽量简单明确
- 尽量用全名，或通用缩写如URL，HTML
- 例如：
 - Serializable; Adjustable;
 - Runnable; Cloneable
 - ActionListener;
 - MouseListener
 - Iterator; Enumeration

命名规则

```
package cn.edu.tsinghua.se2012.connect6;
```

```
import android.app.Activity;
import android.content.pm.Acti
import android.media.AudioMa
import android.media.SoundPo
import android.os.Bundle;
import android.view.View;
import android.view.Window;
import android.widget.Button;
```

```
/**
 * 游戏设置界面
 *
 * @version 1.0
 * @author Shuyang Jiang, Yip
 *
 */
```

```
public class GameSettingActivity extends Activity {
```

```
... ..
```

● 类

- 系统基本模块：名词
- 名词组合中，每个名词的首字母大写
- 尽量简单明确
- 尽量用全名，或通用缩写如URL，HTML
- 例如：
 - Button; TextField
 - HTMLEditorKit; XMLReader
 - InetAddress
 - VendingMachine
 -

命名规则

```
... ..  
/** 声音效果按钮 */  
private Button soundOpen  
/** 振动效果按钮 */  
private Button vibrateOper  
/** 确定按钮 */  
private Button okBtn;  
/** SoundPool对象，用来持  
private SoundPool soundp
```

```
/** 按下声音效果按钮，振  
final int OPTION_BUTTON  
/** 按下确定按钮发出的声  
final int OK_BUTTON = 1;
```

```
/**
```

```
* 创建界面，做一些数据的初始化工作  
*/
```

```
@Override
```

```
public void onCreate(Bundle savedInstanceState) {
```

```
... ..
```

● 方法

- 服务：动词
- 词的组合中，第一个词首字母小写，其余词首字母大写
- 例如：
 - add(); addAll();
 - addElement();
 - toString ();
 -

命名规则

```
... ..  
/** 声音效果按钮 */  
private Button soundOpenBtn;  
/** 振动效果按钮 */  
private Button vibrateOpenBtn;  
/** 确定按钮 */  
private Button okBtn;  
/** SoundPool对象，用来  
private SoundPool soundp  
  
/** 按下声音效果按钮，振  
final int OPTION_BUTTON  
/** 按下确定按钮发出的声  
final int OK_BUTTON = 1;  
  
/**  
* 创建界面，做一些数据的  
*/  
@Override  
public void onCreate(Bundle  
... ..
```

● 变量

- 除临时变量外，名词组合，第一个单词首字母小写，其余单词首字母大写
- 避免以“_”及“\$”字符开头
- 变量名称简短明了，说明变量的用处
- 临时变量可采用单字母，如i、j、k、m、n用于整数变量，a、b、c、d用于字符变量
- 例如：
 - String label;
 - String command;
 - Button okButton;

命名规则

```
... ..  
/** 声音效果按钮 */  
private Button soundOp  
/** 振动效果按钮 */  
private Button vibrateO  
/** 确定按钮 */  
private Button okBtn;  
/** SoundPool对象，用  
private SoundPool sour
```

```
/** 按下声音效果按钮，发出从本按钮发出的声音的标记 */
```

```
final int OPTION_BUTTON = 0;
```

```
/** 按下确定按钮发出的声音的标记 */
```

```
final int OK_BUTTON = 1;
```

```
/**
```

```
* 创建界面，做一些数据的初始化工作
```

```
*/
```

```
@Override
```

```
public void onCreate(Bundle savedInstanceState){
```

```
... ..
```

● 常量

- 大写字母
- 单词之间用“_”连接，例如：
 - Color.BLACK;
 - JOptionPane.ERROR_MESSAGE;
 -

Java程序设计风格

- Java文件组织
 - 基本结构
 - 文件注释、版权声明
 - **Package**和**Import**
 - 类或接口声明
 - 基本要求
 - 每个源文件由多个部分组成
 - 各部分之间用空行间隔，并加以适当的注释
 - 每个文件避免过长（< 2000行）
- “一致性”

I. 文件注释，版权声明

```
/*  
 * Copyright 2012 Shuyang Jiang, Yipeng Ma and Bo Liu  
 *  
 * This file is part of Connect6.
```

Connect6 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

Connect6 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with Connect6. If not, see <<http://www.gnu.org/licenses/>>.

```
*/
```

```
package cn.edu.tsinghua.se2012.connect6;
```

```
... ..
```

II. Package和Import

... ..

```
package cn.edu.tsinghua.se2012.connect6;
```

```
import android.app.Activity;  
import android.content.pm.ActivityInfo;  
import android.media.AudioManager;  
import android.media.SoundPool;  
import android.os.Bundle;  
import android.view.View;  
import android.view.Window;  
import android.widget.Button;
```

```
/**
```

```
 * 实现点击主菜单"关于我们"按钮弹出的界面
```

```
 *
```

```
 * @version 1.0
```

```
 * @author Shuyang Jiang, Yipeng Ma and Bo Liu
```

```
 *
```

```
 */
```

```
public class AboutUsActivity extends Activity{
```

III.类声明：类注释、名称

```
... ..  
package cn.edu.tsinghua.se2012.connect6;  
  
import android.app.Activity;  
import android.content.pm.ActivityInfo;  
import android.media.AudioManager;  
import android.media.SoundPool;  
import android.os.Bundle;  
import android.view.View;  
import android.view.Window;  
import android.widget.Button;  
  
/**  
 * 实现点击主菜单"关于我们"按钮弹出的界面  
 *  
 * @version 1.0  
 * @author Shuyang Jiang, Yipeng Ma and Bo Liu  
 *  
 */  
  
public class AboutUsActivity extends Activity{  
... ..
```

III.类声明： 属性

... ..

```
public class AboutUsActivity extends Activity{
    /** 确定按钮 */
    Button okBtn;
    /** SoundPool对象，用来播放确定按钮按下的声音 */
    private SoundPool soundpool;

    /**
     * 创建界面，做一些数据的初始化工作
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        this.setRequestedOrientation(
            ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);// 设置为竖屏
        //先去除应用程序标题栏 注意：一定要在setContentView之前
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        //将我们定义的窗口设置为默认视图
        setContentView(R.layout.aboutus);
        ... ..
    }
}
```


III.类声明：方法

```
/**
 * 创建界面，做一些数据的初始化工作
 */
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    this.setRequestedOrientation(
        ActivityInfo.SCREEN_ORIENTATION_PORTRAIT); // 设置为竖屏
    // 先去除应用程序标题栏 注意：一定要在 setContentView 之前
    requestWindowFeature(Window.FEATURE_NO_TITLE);
    // 将我们定义的窗口设置为默认视图
    setContentView(R.layout.aboutus);

    okBtn = (Button) findViewById(R.id.aboutusok);
    soundpool = new SoundPool(2, AudioManager.STREAM_SYSTEM, 0);
    okBtn.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            playSound(); finish();
        }
    });
} // onCreate 方法结束
```

III.类声明：方法

```
....  
/**  
 * 播放确定按钮按下的声音  
 */  
public void playSound(){  
    if (StartActivity.soundOpen) {  
        final int sourceId = soundpool.load(AboutUsActivity.this,  
            R.raw.okbutton, 1);  
        soundpool.setOnLoadCompleteListener(  
            new SoundPool.OnLoadCompleteListener() {  
                public void onLoadComplete(SoundPool soundPool,  
                    int sampleId, int status) {  
                        soundPool.play(sourceId, 1, 1, 0, 0, 1);  
                    }  
            });  
    }  
} //playSound方法结束  
  
} //类结束
```

Java程序设计风格

- 程序注释
 - 实现注释
 - 文档注释
- 空格和缩进
 - 合理利用，控制版面，帮助理解

- “可理解性”

程序注释

- Implementation Comments
 - `/*.....*/; //`
 - 注释程序块
 - 单句注释
 - 在语句末尾的补充注释
 - “删除”无用语句

```
/*  
 * Here is a block comment.  
 */
```

```
if (condition) {  
    /* Handle the condition. */  
    ...  
}
```

```
if (a == 2) {  
    return TRUE;    /* special case */  
}  
else {  
    return isPrime(a);    /* works only for odd a */  
}
```

```
if (foo > 1) {  
    // Do a double-flip.  
    ...  
}  
else {  
    return false;    // Explain why here.  
}  
  
//if (bar > 1) {  
//    // Do a triple-flip.  
//    ...  
//}  
//else {  
//    return false;  
//}
```

- Documentation Comments

- Javadoc
- `/** */`
- `@return`
- `@param`
- `@see`
- `@since`
- `@link`
- HTML标识

注释：类、属性、方法

```
... ..  
/**  
 * 实现点击主菜单"关于我们"按钮弹出的界面  
 *  
 * @version 1.0  
 * @author Shuyang Jiang, Yipeng Ma and Bo Liu  
 */  
  
public class AboutUsActivity extends Activity{  
    /** 确定按钮 */  
    Button okBtn;  
    /** SoundPool对象，用来播放确定按钮按下的声音 */  
    private SoundPool soundpool;  
  
    /**  
     * 创建界面，做一些数据的初始化工作  
     */  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        ... ..  
    }  
}
```


Class AboutUsActivity

```
java.lang.Object
    Activity
        cn.edu.tsinghua.se2012.connect6.AboutUsActivity
```

```
public class AboutUsActivity
    extends Activity
```

实现点击主菜单"关于我们"按钮弹出的界面

Version:

1.0

Author:

Shuyang Jiang, Yipeng Ma and Bo Liu

Field Summary

Fields

Modifier and Type	Field and Description
(package private) Button	<code>okBtn</code> 确定按钮
private SoundPool	<code>soundpool</code> SoundPool对象，用来播放确定按钮按下的声音

程序的空格和缩进

- 行的长度不超过70个字符每行
- 断行的基本原则
 - 在逗号之后
 - 在操作符之前
 - 将不同层次的操作分隔开
 - 新行缩进4或8个字符

- “留白”

```
someMethod(longExpression1,  
            longExpression2,longExpression3,  
            longExpression4, longExpression5);
```

```
var = someMethod1(longExpression1,  
                  someMethod2(longExpression2,  
                              longExpression3));
```

```
longNm1 = longNm2 * (longNm3 + longNm4 - longNm5)  
          + 4 * longNm6; // PREFER
```

```
longNm1 = longNm2 * (longNm3 + longNm4  
                    - longNm5) + 4 * longNm6; // AVOID
```

- 每个声明一行，并添加注释

```
//preferred  
int level; // indentation level  
int size; // size of table
```

```
//avoid  
int level, size;
```

- 类型和标识符间由空格（或Tab）分隔

```
int    level;           // indentation level  
int    size;            // size of table  
Object currentEntry;    //currently selected table entry
```

Java程序设计风格：程序声明

- 初始化：尽量在变量声明的同时赋初值
- 声明的位置：尽量在程序块的开始部分

```
void myMethod() {  
    int int1 = 0;    // beginning of method block  
  
    if (condition) {  
        int int2 = 0; // beginning of "if" block  
        ...  
    }  
}
```

- 避免局部变量与高层变量重名。

```
int count;  
...  
myMethod() {  
    if (condition) {  
        int count = 0; // AVOID!  
        ...  
    }  
    ...  
}
```

- 函数名与参数列表前的括号之间不留空格。
- “{” 在函数名声明的同一行末尾
- “}” 单独一行，并适当缩进，与声明的起示位置对齐
- 空函数 “{” 与 “}” 一起置于函数声明一行的末尾
- 函数之间以空行分隔

Java程序设计风格：程序声明

```
class Sample extends Object {  
    int ivar1;  
    int ivar2;  
  
    Sample(int i, int j) {  
        ivar1 = i;  
        ivar2 = j;  
    }  
  
    int emptyMethod() {}  
  
    ...  
  
}
```


- 简单语句
 - 每一条语句单独一行
- 复合语句
 - 用缩进明确语句间的层次关系
 - if-then-else语句
 - for
 - while
 - do-while
 - switch
 - try-catch

```
if (condition) {  
    statements;  
}
```

```
if (condition) {  
    statements;  
} else {  
    statements;  
}
```

```
if (condition) {  
    statements;  
} else if (condition) {  
    statements;  
} else {  
    statements;  
}
```

```
for (initialization; condition; update) {  
    statements;  
}
```

```
for (initialization; condition; update);
```

```
while (condition) {  
    statements;  
}
```

```
while (condition);
```

```
do {  
    statements;  
} while (condition);
```

```
switch (condition) {  
    case ABC:  
        statements;  
        /* falls through */  
  
    case DEF:  
        statements;  
        break;  
  
    case XYZ:  
        statements;  
        break;  
  
    default:  
        statements;  
        break;  
}
```

```
try {  
    statements;  
} catch (ExceptionClass e) {  
    statements;  
}
```

```
try {  
    statements;  
} catch (ExceptionClass e) {  
    statements;  
} finally {  
    statements;  
}
```

- 命名规则
- 文件组织
- 程序注释、空行、缩进
- 程序声明
- 语句结构

- “可阅读性”

编程实践

- 类的变量尽量为私有变量(封装)
- 变量赋值
 - 避免在一个语句中有多个赋值，如：
`foo.aChar = bar.bChar = 'c' ;`
 - 避免为提高程序运行效率，在一个语句中有嵌套其他语句，如：
`d = (a = b+c) +e;`

- 为了增加程序的清晰性，不要因为有缺省的运算优先级而省略括号。如：
if (a == b && b == c) // Avoid !!
if ((a==b) && (b==c)) // Correct!!
- 风格检查工具：
 - stylecheck
 - findbugs (书写风格中隐含错误)
- “自文档化”、“一致性”

程序复杂性度量

- 程序复杂性主要指模块内程序的复杂性。
 - 软件开发费用
 - 软件开发周期
 - 软件内部潜伏错误量
- 程序复杂性度量的主要参数
 - 规模
 - 难度
 - 结构

代码行度量法

- 前提假设
 - 程序复杂性随着程序规模的增加不均衡地增长
 - 控制程序规模的方法最好是采用分而治之的办法，将一个大程序分解成若干个简单的可理解的程序段
- 程序出错率与代码行间的关系
 - Trayer:
 - 0.04%~7%之间
 - 出错率与源程序行数之间不存在简单的线性关系
 - Lipow:
 - 对于小程序，每行代码出错率为1.3%~1.8%；对于大程序，每行代码的出错率增加到2.7%~3.2%之间
 - 对于少于100个语句的小程序，源代码行数与出错率是线性相关的；随着程序的增大，出错率以非线性方式增长

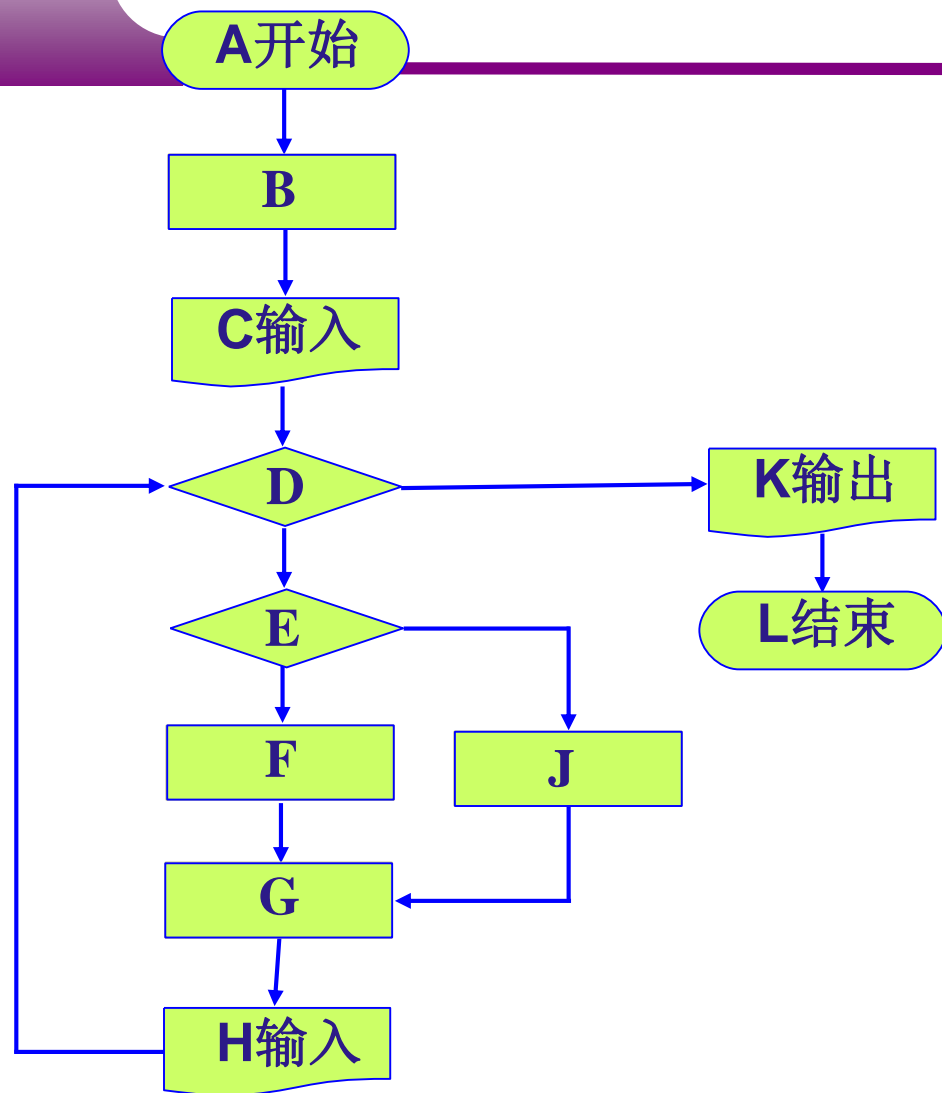
代码行度量法

- 六子棋游戏代码行数

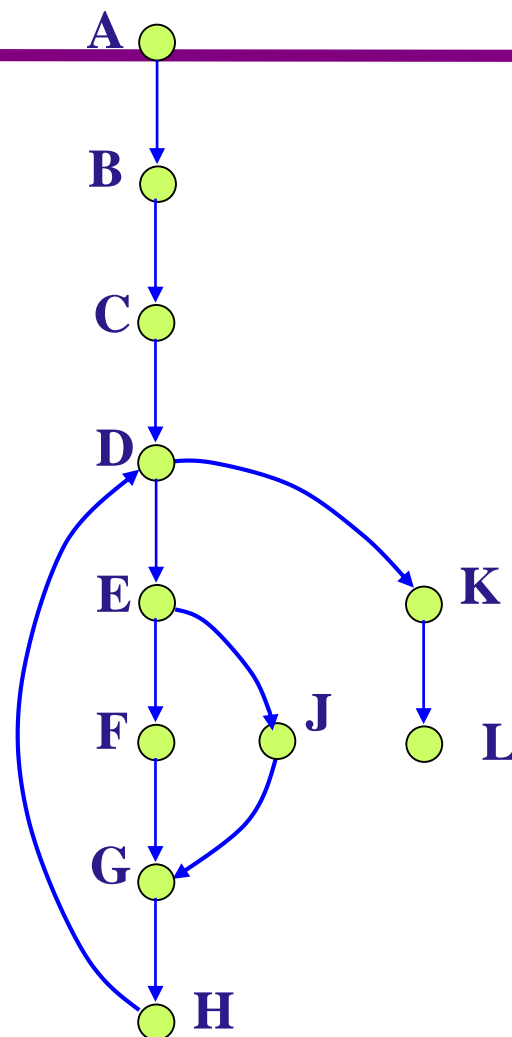
版本编号	JAVA 代码行数	XML 代码行数	代码总行数	累计 提交次数	开发团队
V20101224	962	0	962	7	一起编
V20120717	819	122	941	39	Alick
V20130104	1822	776	2598	60	喜迎十八大
V20140111	1954	51	2005	95	这个以后再议
V20150103	3776	827	4603	100	Alick

McCabe环路复杂性度量

- 基于程序控制流的复杂性度量方法
 - 程序控制流图
 - 程序流程图的退化模型，结点为一个处理，结点间由有向弧连接。
 - 描述程序的控制流程，不表现对数据的具体操作以及分支和循环的具体条件。



程序流程图



程序控制流图

McCabe环路复杂性度量

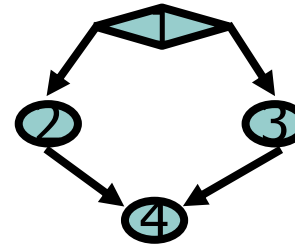
- 有向图 G 所封闭的环路的个数 $V(G)$:

$$\begin{aligned} V(G) &= G \text{ 中的边数} - G \text{ 中的节点数} + 2 \\ &= G \text{ 中的判定的节点数} + 1 \\ &= G \text{ 中的区域数（单连通域）} \end{aligned}$$



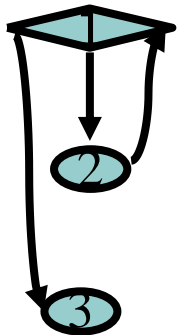
$$\begin{aligned}
 V &= 1 - 2 + 2 \\
 &= 0 + 1 \\
 &= 1
 \end{aligned}$$

顺序型



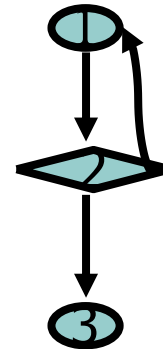
$$\begin{aligned}
 V &= 4 - 4 + 2 \\
 &= 1 + 1 \\
 &= 2
 \end{aligned}$$

选择型



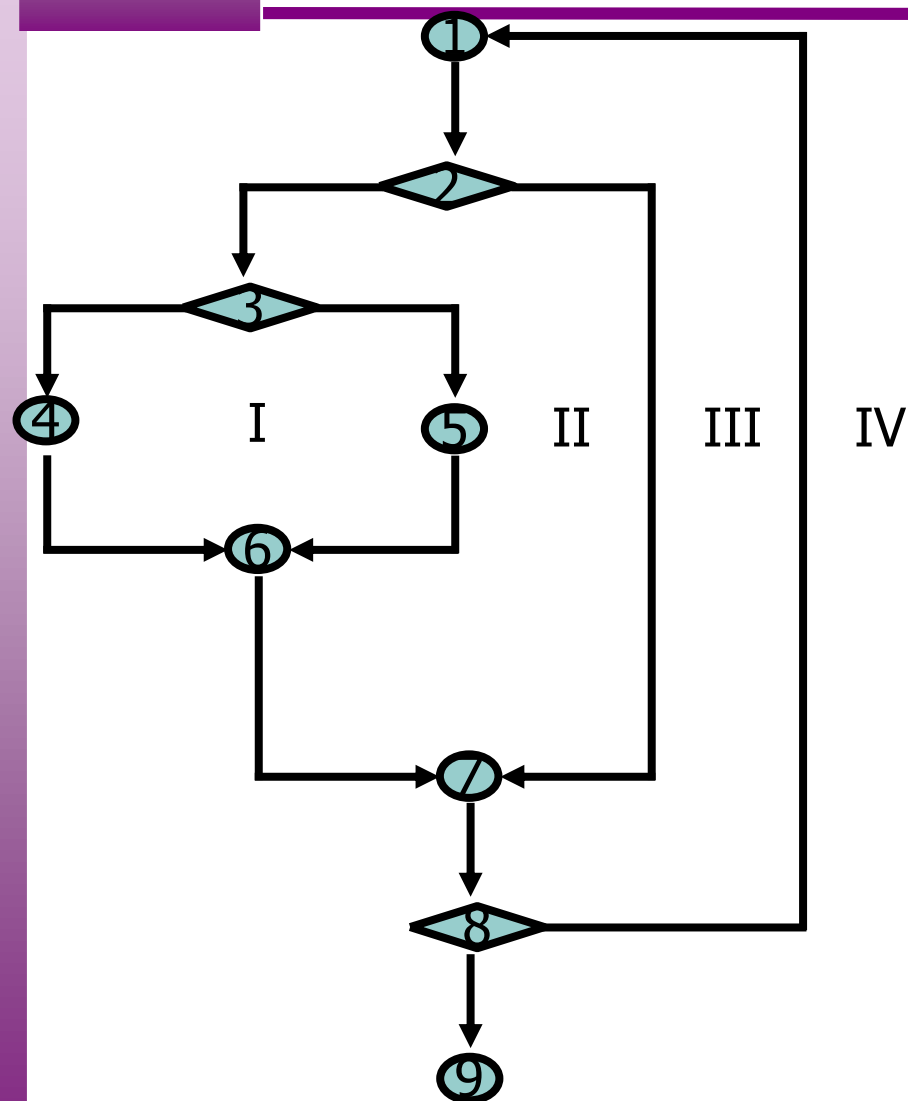
$$\begin{aligned}
 V &= 3 - 3 + 2 \\
 &= 1 + 1 \\
 &= 2
 \end{aligned}$$

While型循环

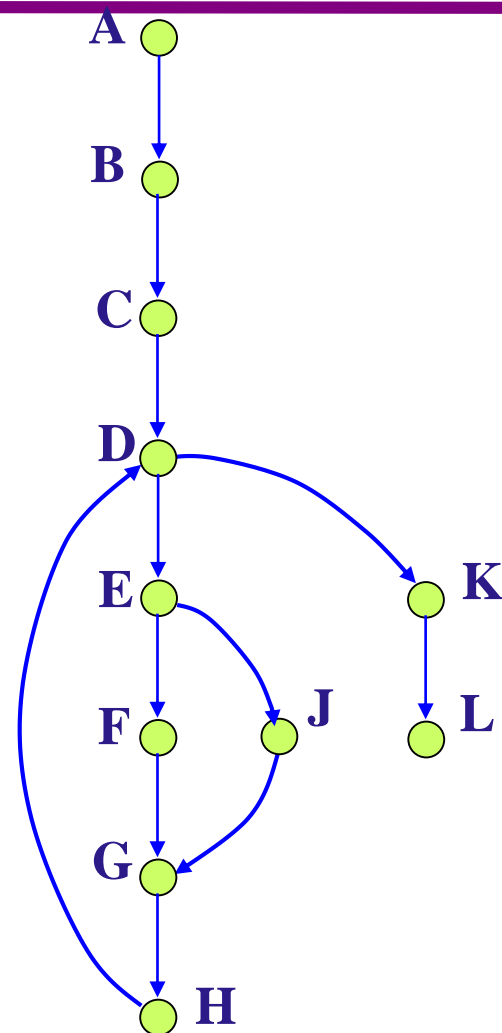


$$\begin{aligned}
 V &= 3 - 3 + 2 \\
 &= 1 + 1 \\
 &= 2
 \end{aligned}$$

Until型循环



$$V=11-9+2=3+1=4$$



$$V=12-11+2=2+1=3$$

McCabe环路复杂性度量

- 环路复杂度取决于程序控制结构的复杂度。当程序的分支数目或循环数目增加时其复杂度也增加。
- **McCabe**环路复杂度隐含前提是：错误与程序的判定加上例行子程序的调用数目成正比。
- **McCabe**建议，对于复杂度超过10的程序，应分成几个小程序，以减少程序中的错误。**Walsh**用实例证实了这个建议的正确性。他发现，在**McCabe**复杂度为10的附近，存在出错率的间断跃变。

小结

- 程序复杂性度量
 - 代码行数
 - McCabe环路复杂性(程序控制图中单连通域个数)
- 控制程序的复杂度
 - 便于理解、便于维护