# 1、白盒测试方法

条件覆盖CC（Condition Coverage），设计足够多的测试用例，运行被测程序，使得每一判定语句中每个逻辑条件的可能取值至少满足一次。

条件覆盖率的公式：条件覆盖率=被评价到的条件取值的数量/条件取值的总数X100%[1]

条件覆盖的缺点：只考虑到每个判定语句中的每个表达式，没有考虑到各个条件分支（或者涉及不到全部分支），即不能够满足判定覆盖.

条件组合覆盖，也称多条件覆盖MCC （Multiple Condition Coverage），设计足够多的测试用例，

使得每个判定中条件的各种可能组合都至少出现一次（以数轴形式划分区域，提取交集，建立最少的测试用例）。

这种方法包含了“分支覆盖”和“条件覆盖”的各种要求。满足条件组合覆盖一定满足判定覆盖、条件覆盖、判定条件覆盖。

条件组合覆盖率的公式：条件组合覆盖率=被评价到的条件取值组合的数量/条件取值组合的总数条件组合覆盖的缺点：判定语句较多时，条件组合值比较多。

语句覆盖 SC（Statement Coverage），就是设计若干个测试用例，运行被测程序，使得程序中每一可执行语句至少执行一次。

这里的“若干个”，意味着使用测试用例越少越好。语句覆盖在测试中主要发现缺陷或错误语句。

判定条件覆盖CDC（Condition/ Decision Coverage），设计足够多的测试用例，使得判定中的每个条件的所有可能（真/假）至少出现一次，

并且每个判定本身的判定结果也至少出现一次。[1]

判定条件覆盖率的公式：条件判定覆盖率=被评价到的条件取值和判定分支的数量/（条件取值总数+判定分支总数）.

判定条件覆盖的缺点：没有考虑单个判定对整体结果的影响，无法发现逻辑错误。

# 2、圈复杂度

1. V（G）=P+1 （P是判定节点）
2. V（G）=D （D是区域数）
3. V（G）=E-N+2（E是边的条数，N是节点数）

# 3、桩模块和驱动模块

桩模块（Stub）是指模拟被测的模块所调用的模块，而不是软件产品的组成的部分。

主模块作为驱动模块，与之直接相连的模块用桩模块代替。

# 4、自顶向下测试和自底向上测试

**自顶向下测试**：是从程序的初始模块开始测试。

（1）该方\*\*\*在早期发现顶层的错误。

（2）早期的程序框架可以进行演示

（3）需要开发桩模块辅助测试。有些甚至需要多个桩模块辅助，加大了桩模块本来的错误影响。

（4）测试完一个上层模块后，挑选哪个模块作为下一个测试模块，以及测试的顺序没有唯一的界定标准。

**优点**：较早地验证了主要控制和判断点；按深度优先可以首先实现和验证一个完整的软件功能；功能较早证实，带来信心；只需一个驱动，减少驱动器开发的费用；支持故障隔离。

**缺点**：柱的开发量大；底层验证被推迟；底层组件测试不充分。

**自底向上测试**：是从程序的底层模块开始测试。

（1）I/O操作可以提前测试，更好提交测试用例。

（2）测试后比较容易观察输出。

（3）需要开发驱动模块。

（4）直到最后一个模块提交，程序才能完整的系统测试。

**优点**：对底层组件行为较早验证；工作最初可以并行集成，比自顶向下效率高；减少了桩的工作量；支持故障隔离。

**缺点**：驱动的开发工作量大；对高层的验证被推迟，设计上的错误不能被及时发现。

# 5、关于测试工具的描述

* LoadRunner-负载压力测试：预测系统性能。
* JMeter+Badboy：基于JAVA的压力测试工具，Badboy用来进行脚本的录制
* 功能测试：通过自动录制、检测和回放用户的应用操作。将输出记录同预先给定的记录比较。
* Junit：白盒测试工具：针对代码测试
* 测试管理工具：对测试需求、计划、用例、实施进行管理
* 测试辅助工具：本身不执行，可以生成测试数据，为测试提供数据准备
* 负载压力测试：LoadRunner:预测系统行为和性能的工业标准级负载测试工具。模拟上千万用户同时实施并发操作，来实时监控可能发生的问题。
* 功能测试： QTP(quicktest professional):自动测试工具
* 白盒测试：C++ TEST（做C和C++的白盒测试）、JUnit（Java白盒测试）
* 缺陷管理工具：Mantis、BugFree、QC、TD
* 用例管理工具：TestLink、QC
* 测试辅助工具：SVN

# 6、代码走查和单元测试

代码走查：只检查代码中是否有错误。

单元测试：是指对软件中的最小可测试单元进行检查和验证。

# 7、静态测试和动态测试

按照是否需要运行代码分为静态测试和动态测试；

静态中的代码测试主要检查代码的逻辑结构、功能正确性。

静态测试包括代码检查、静态结构分析、代码质量度量等。它可以由人工进行，充分发挥人的逻辑思维优势，也可以借助软件工具自动进行。

人工测试技术主要包含三种静态测试技术，分别是走查、审查和正式评审。

# 8、集成测试

1）非渐增组装测试（非增式集成测试）：将单元测试后的模块按照总体的结构图一次性集成起来，然后把连接的整体进行程序测试。

一般用黑盒法来编写测试集并进行测试。

程序错误易出现，不容易集成成果。单元测试使用的辅助模块多，适合于规模小的开发系统。

2）渐增组装测试（增式集成测试）：在单元测试的基础上，采用自顶向下或自底向上逐层安装测试，知道最后安装测试完毕。

也可采用自顶向下与自底向上相结合集成测试，单元测试与集成测试相结合来进行集成测试。

将错误分解，容易找到错误并测试成功，适合于大规模的开发系统。

非增量--每个模块测试完了再连接；

增量：测一个模块，就连接一个模块。

# 9、说明书的英文

SOW:statement of work，工作任务说明书

HLD: High Level Design,概要设计说明书

LLD: Low Level Design，详细设计说明书

UTC: Unit Testing Cases，单元测试用例

# 10、白盒-路径测试

路径测试属于白盒测试，而在单元测试和集成测试中用到了白盒测试。

# 11、软件调试技术

软件调试技术:

试探法（强行排错法），

回溯法：人工沿着程序的控制流程往跟踪代码，直到找出错误根源为止

对分查找法：缩小错误的范围，

归纳演绎

原因排除法

# 12、单元测试的策略

1. 逻辑覆盖、
2. 循环覆盖、
3. 同行评审、
4. 桌前检查、
5. 代码走查、
6. 代码评审、
7. 景泰数据流分析

# 13、测试框架

OCUnit 是 OC 官方测试框架， 现在被 XCTest 所取代。  
XCTest 是与 Foundation 框架平行的测试框架。  
GHUnit 是第三方的测试框架。  
OCMock都是第三方的测试框架。