白盒测试

白盒测试也称结构测试或逻辑驱动测试，是针对被测单元内部是如何进行工作的测试。它根据程序的控制结构设计测试用例，主要用于软件或程序验证。

白盒测试法检查程序内部逻辑结构，对所有逻辑路径进行测试，是一种穷举路径的测试方法。但即使每条路径都测试过了，仍然可能存在错误。因为：

穷举路径测试无法检查出程序本身是否违反了设计规范，即程序是否是一个错误的程序；穷举路径测试不可能查出程序因为遗漏路径而出错；穷举路径测试发现不了一些与数据相关的错误。

白盒测试主要是检查程序的内部结构、逻辑、循环和路径。常用测试用例设计方法有：逻辑覆盖：以程序的内部逻辑结构为基础，分为语句覆盖、判定覆盖、条件覆盖、判定-条件覆盖、条件组合覆盖等。基本路径测试：在程序控制流程的基础上，分析控制构造的环路复杂性，导出基本可执行路径集合，从而设计测试用例。

黑盒测试

黑盒测试也称功能测试或数据驱动测试，它是在已知产品所应具有的功能，通过测试来检验每个功能都是否能够正常使用。在测试时，把程序看作一个不能打开的黑盒子，在完全不考虑程序内部结构和内部特性的情况下，测试人员进行直接测试，检查系统功能是否按照需求规格说明书的规定正常使用、是否能适当的接收输入数据而输出正确的结果等，检查相应的文档是否采用了正确的模板、是否满足规范需求。

黑盒测试有两种基本方法，即“通过测试”和“失败测试”。在进行“通过测试”时，实际上是确认软件能做什么，而不会考验其能力如何。软件测试员只运用最简单、直观的测试案例。在设计和执行测试案例时，总是先进行通过测试。在进行破坏性试验之前，看一看软件基本功能是否能够实现。在确信了软件正确运行之后，就可以采取各种手段通过搞垮软件来找出缺陷。纯粹为了破坏软件而设计和执行的测试案例，被称为失败测试或迫使出错测试。

优点：比较简单，不需要了解程序内部的代码以及实现；与软件内部实现无关；从用户角度出发，很容易的知道用户会用到哪些功能，会遇到哪些问题；在做软件自动化测试时较为方便；基于软件开发文档，所以也能知道软件实现了文档中的哪些功能。

缺点：不可能覆盖所有代码，覆盖率较低，大概只能达到总代码量的30%；自动化测试的复用性较低。