1．概要部分

（1）代码能符合需求和规格说明么？大体符合

（2）代码设计是否有周全的考虑？否，还有一些没有考虑到的，会继续完善的

（3）代码容易维护么？比较容易维护

（4）代码的每一行都执行并检查过了吗？是的，都可以执行

2．设计规范部分

（1）设计是否遵从已知的设计模式或项目中常用的模式？遵从

（2）有没有硬编码或字符串/数字等存在？有

（3）代码有没有依赖于某一平台，是否会影响将来的移植（如Win32到Win64）？有，理论上不会有影响

（4）有没有无用的代码可以清除？暂时没有发现

3．代码规范部分

（1）修改的部分符合代码标准和风格么?符合

4．具体代码部分

（1）有没有对错误进行处理？对于调用的外部函数，是否检查了返回值或处理了异常？对错误都进行了处理，没有异常

（2）参数传递有无错误，字符串的长度是字节的长度还是字符（可能是单/双字节）的长度，是以0开始计数还是以1开始计数？并未运用大量字符串

（3）循环有没有可能出现死循环？没有可能，可以重复抽取

（4）有没有使用断言（Assert）来保证我们认为不变的条件真的得到满足？没有

（5）有没有可能导致资源泄露（内存、文件、各种GUI资源、数据库访问的连接，等等）？有没有可能优化？不会导致资源泄露，可以优化

（6）数据结构中有没有用不到的元素？没有

5．效能

（1）代码的效能（Performance）如何？最坏的情况是怎样的？达到了具体任务要求的程度，最坏的情况是代码报错

（2）代码中，特别是循环中是否有明显可优化的部分（C++中反复创建类，C#中 string 的操作是否能用StringBuilder 来优化）？有，可以反复创建类，但我们目前还不太熟

（3）对于系统和网络调用是否会超时？如何处理?目前没有出现超时的现象。假如出现了我们会杀毒；整理系统，减少运行的进程，释放内存、cpu，释放c盘空间；

6．可读性

代码可读性如何？有没有足够的注释？可以读取，有注释，不过较少

7．可测试性

代码是否需要更新或创建新的单元测试？暂时不需要