**代码规范表**

1. 开头注释
2. 需要在开头注明作者、开发日期、代码的功能概述
3. 每一次维护需要在开头的尾部注明时间和维护的具体项
4. 需要开头声明此代码文件是否与个人信息有关、是否可以公布
5. 类（类、接口、枚举、结构体、联合体）、函数（函数、方法）的注释
6. 每一个类需要对其功能、继承项进行注释说明（如果本文件仅仅就一个类，功能叙述部分过于冗余，那么不用写）
7. 每一个函数需要对其功能、参数进行注释说明（如果本文件仅仅就一个类，功能叙述部分过于冗余，那么不用写）。如果参数多于5个，需要编写参数传递的实例（实例，包括测试参数、测试的返回结果、测试时间）
8. 对于类中的方法和属性，我们需要分类（构造、setget、reset、派生类、运算类、应用类、基础类、静态类和其他类）展示
9. 如果声明了全局变量，需要对其作用进行解释说明
10. 较为复杂的过程需要分布进行解释（先一个笼统的过程解释，表明干了什么事；然后把每一步做了什么解释出来、需要排版）
11. 类（类、接口、枚举、结构体、联合体）的命名
12. 是否为一个或多个的首字母大写的单词或单词简称所组成的类名
13. 类名的长度是否小于20
14. 类名是否不包含数字和下划线
15. 类名是否不与其余已定义好的类的类名重复
16. 类的名称是否与类的功能或实体相符合
17. 函数（函数、方法）、参数、变量、属性的命名
    1. 是否为首字母小写的英文单词或其简写的组合（多个单词需要中间加下划线）
    2. 命名是否与实际用途有关（中途形参不做限制，像for 循环里的 i）
18. 异常部分
    1. 对于有可能出现异常的代码是否有错误抛出检测
    2. 是否能够显示详细的错误信息
    3. 是否对代码进行了详细的测试
19. 结构
    1. 本文件的代码在结构上的安排是否合理
    2. 有无无用代码段
    3. 有无无用参数
    4. 是否易于再次使用