# 报告

## 相似度检查算法

首先考虑代码抄袭者将一份抄袭的代码可能做如何修改，可以归类为：1. 完全不修改 2.只修改部分标识符 3. 修改函数的上下位置。并增加声明语句使其编译通过 4. 修改部分简单的代码顺序，如a = b + c + d; 修改为a = b + c; a += d; 5. 修改简单的控制逻辑，如while替换为for循环 6. 修改大量的标识符、代码顺序、控制逻辑、函数调用等。

我们认为如果抄袭者真的有能力对上述第6点进行代码修改，那么可以理解为抄袭者已经对该份代码有了深刻的理解，知道每一行代码做什么，那么我们也没必要判其抄袭。所以接下来讨论的都是上述1-5点。

我发现汇编语言可以很好的针对上述5点，用户完成的高级语言通过编译器编译成汇编语言，能够有效的解决上述问题。1. 完全不修改，这样两者的汇编代码是一样的。2. 解决只修改标识符，因为汇编语言的寄存器寻址方式并不依赖于变量的标识符名称（立即寻址，寄存器直接寻址，寄存器间接寻址，寄存器基址加变址寻址等）。3. 解决函数上下位置关系，对一个cpp文件可以通过提取出每一个函数的汇编代码构成一个汇编代码块集合，然后两组代码块集合内的汇编代码互相计算相似度。4. 简单修改代码顺序，因为汇编语言底层还是将写在一起的语句拆分成一个一个指令执行，所以这个问题自然而然解决。5. 修改简单的控制逻辑，只需要将每一个函数代码块的标签去掉，然后做相似度对比即可。综上，通过对比每个函数的汇编代码块可以很好的防止作弊问题。

## 算法描述及工程实现细节

算法流程：

1. 通过clang统计得到该文件的标识符集合

2. 通过g++ 生成汇编语言（因为g++生成的底层汇编代码标签和高级语言函数名可以通过某种转换方式轻松获得）

3. 根据g++ 的汇编标签和函数名转换规则，找到该文件的汇编代码，同时去掉通过include进来的汇编代码得到两个汇编代码块集合，集合中每个元素都是一个汇编代码块。

4. 两个集合left,right为代码块集合，left集合的元素i通过计算 |right|次最长公共子序列，找到right集合中子序列最长的代码块，并累加到left\_sim变量中，然后将right集合中相应的代码块从集合中移除，然后继续为left集合中的元素计算left\_sim，直到left集合计算完毕。

5. 为了保证codesim的对称性，与第4步相反通过对right集合中的每一个元素找left集合中最长公共子序列，累加到right\_sim变量，然后移除left相应的代码块。

6. 计算(left\_sim + right\_sim)/(两个集合汇编指令的总行数)即可得到相似度。

通过该算法可以知道，完全相同的两个代码相似度是1，完全不同的代码相似度是0.