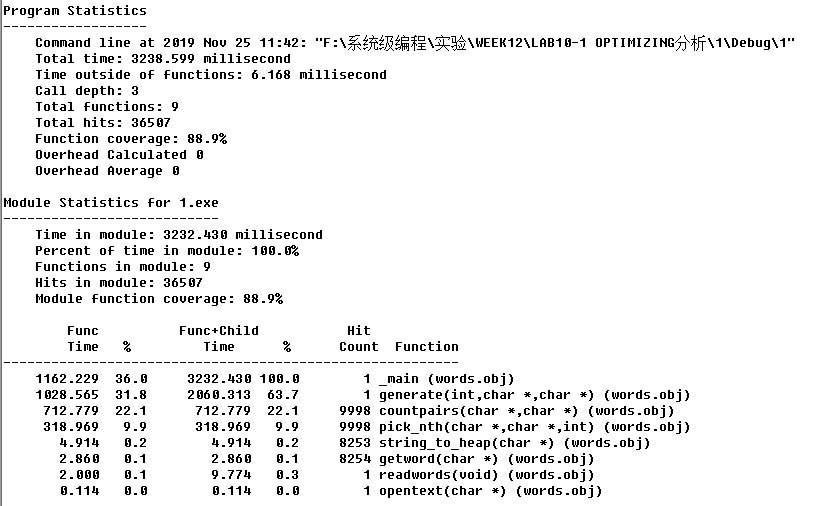
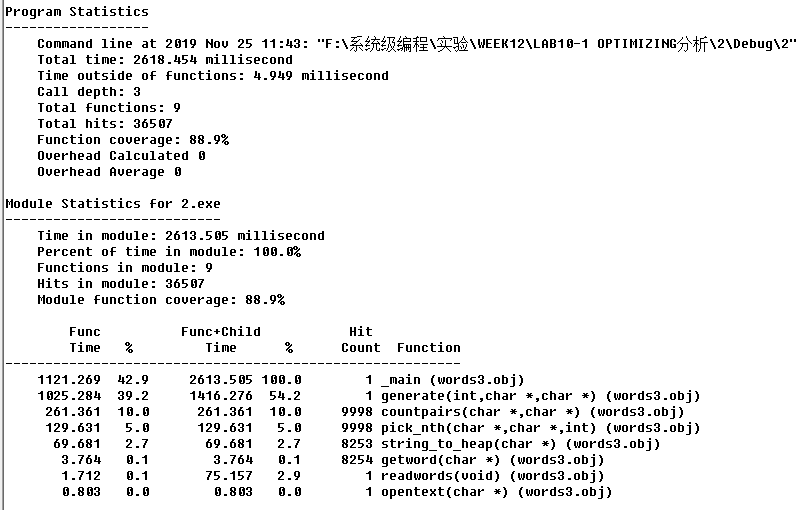
两个配置文件的输出对比：

words.cpp：

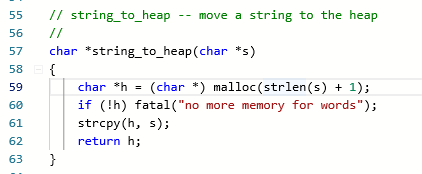


words3.cpp：

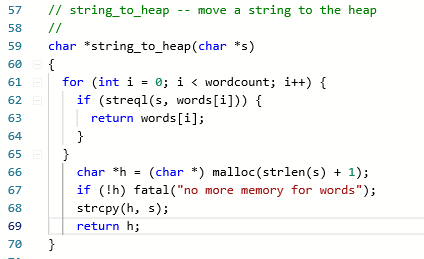


char \*string\_to\_heap(char \*s)函数对比如下：

优化前：



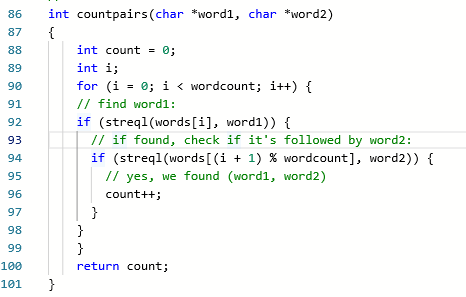
优化后：



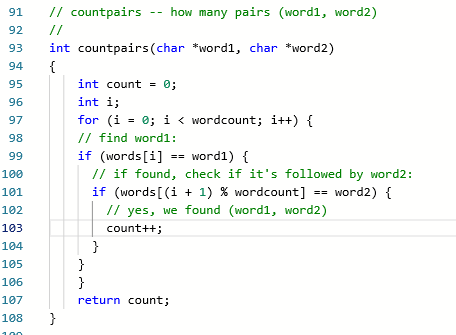
优化后的代码先通过一个循环判断，单词是否已经存在，若在直接返回。这样就不会使得重复的单词重复分配空间，往往分配空间是十分耗时的，所以这样做能提高程序的效率。

int countpairs(char \*word1, char \*word2)函数：

优化前：



优化后：



优化前是通过调用strcmp函数来找到字符串，这个函数的开销就比较大。优化后的代码，通过比较地址是否相同来判断是否是同一个单词。因为在前面那个分配空间的函数中，已做了优化，相同的单词使用相同的空间，故能通过比较地址是否相同来判断单词是否相同。