编译总结感想

这学期的编译实验不能说是一帆风顺吧，也可以说是命运多舛吧，确实是有点不顺利，最后的结果也不是很理想，但是也学到了很多东西，总体来说也算是一次不错的学习经历。

最先开始编译实验时，选择语言让我纠结了一阵，最终考虑到想增进一下C++的使用技巧，确实也是好一阵没有用C++了，所以最终选择了C++。综合一学期来看，感觉其实如果对智能指针，STL，raii机制有一定了解的话，选C++和Java实现起来的难度没有差距很大。这些机制可以帮助避免考虑内存分配，内存回收等内存管理方面的细节，调试起来没有那么恼火。此外Java的好处可能在于其封装的包更完善一点吧，比如大整数类这样的，C++遇到大整数就最好自己手写了；但C++的引用，和`std::move()`这套传值or传引用用好了感觉会比java的深复制浅复制方便一点。

开始的几次作业没有遇到太大的阻碍，因为一直记得往届的提醒要为下一次设计预留一些端口，所以每次的改动都没有太大。在错误处理的时候遇到了一点小困难。因为课程组给的辅助样例就只是把每种错误都测了一次，强度一般，因此debug花了比较久的时间，后面还是依靠讨论区同学的测试样例才通过。

后面遇到了颇令人头大的代码生成，这个任务拆成了代码生成1和代码生成2。1是针对较为简单的部分，我感觉这个难点还是在于调试吧，因为辅助样例库很多都涉及到了代码生成1中还未覆盖的模块，但是也由于代码生成1所涵盖的模块不多，可以像OO那样进行全面性的检验。根据该其可能出现的各种情况，列出一张流程图，根据这个流程图构造数据，实现全面覆盖，检查bug。

代码生成1要求在相对比较短的时间完成，而那时候在课外上还有一些科研方面较为繁重的任务，因此我在做代码生成1的时候并没有进行寄存器的分配，而是直接将每个变量都存在内存里，需要时再把它取出来进行运算，得到的结果再存回内存中。这样的写法可以比较方便地通过代码生成1，代码生成2，但是在竞速的时候就自然吃了大亏。寄存器的指派和分配是能提高寄存器分配的最好用的利器，因此基本上这一方面需要重新再写过。但后面遇到了疫情，本人也是受到了一些影响，加上期末临近各科的压力也变大了，所以真正抽出的用来做代码优化的时间并不多。我也没有很好地做好编码前的设计，经常写了一半发觉不妥，对于像是编译器类应该预留怎么样的接口，应该需要怎样的功能，都是写了再调整再修改，也导致了各个类耦合度较多，封装的不够，不利于调试，也严重影响了效率。最后也只完成了除法优化和临时寄存器分配。这个经历留下的教训还是蛮深刻的。

建议：

- 代码生成1的窗口期是否有可能延长几天。因为这个任务的难度还是比较大的，而且在课外上还有一些科研方面的任务，因此在这个任务上花的时间可能会比较多，如果能够延长一点时间，可能会有更好的效果。
- 设计报告建议增加一个编码时修改模块，有的时候是在设计的过程中发现之前设计的不妥之处，但写的时候不太清楚应该加入到哪个模块当中

感谢一学期来老师、助教、同学们对我的帮助！