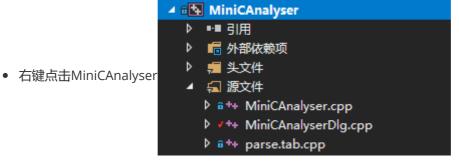
MiniC

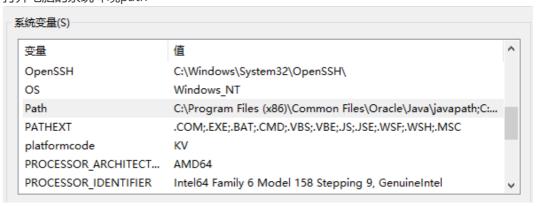
run in mfc

- 双击打开MiniCAnalyser.vcxproj
- 将项目目录下的 winflex 文件夹的目录 (项目目录/winflex) 添加进系统环境

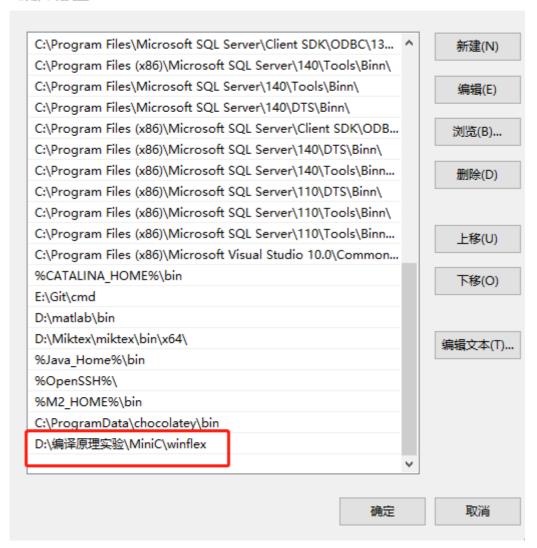
0



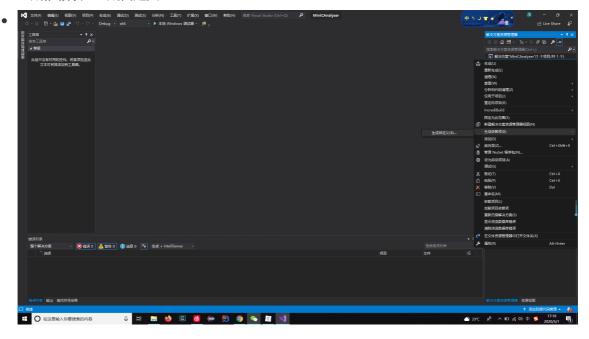
。 打开电脑的系统环境path



编辑环境变量



• 生成依赖项->生成自定义



• 查找现有的,在目录 项目目录\winflex\custom_build_rules\Bison 下找到文件 win_flex_bison_custom_build.targets 选择打开,并如上图选中点击确定

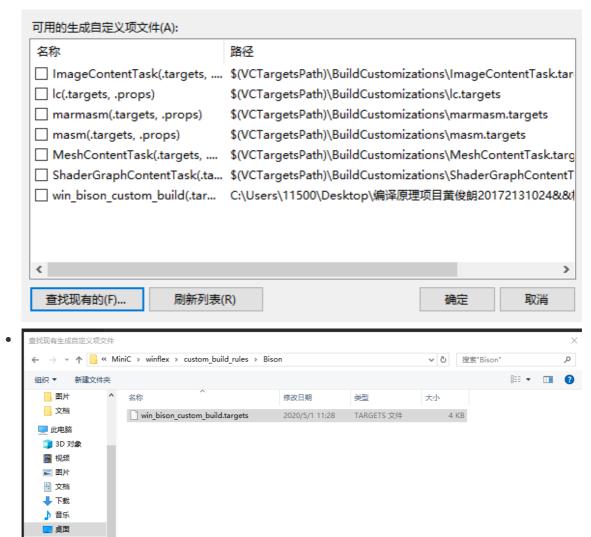
•

Visual C++ 生成自定义项文件 ∨

取消

打开(<u>O</u>)



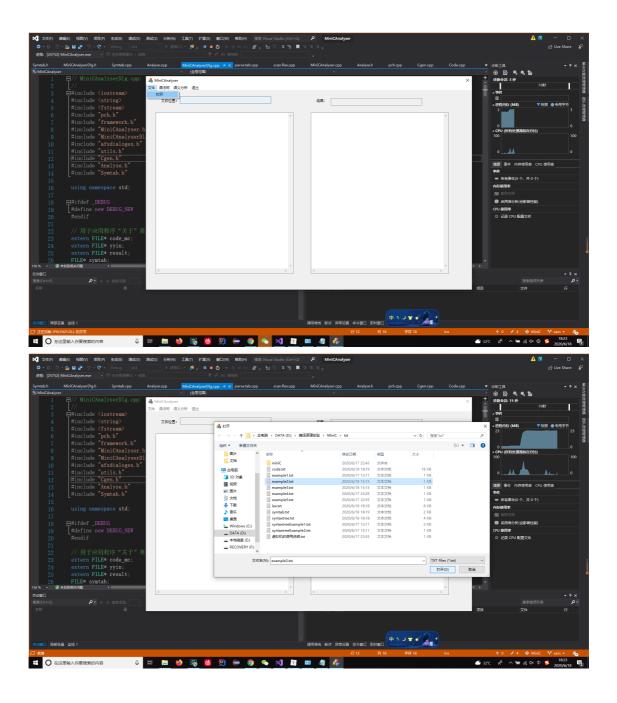


• 编译运行即可

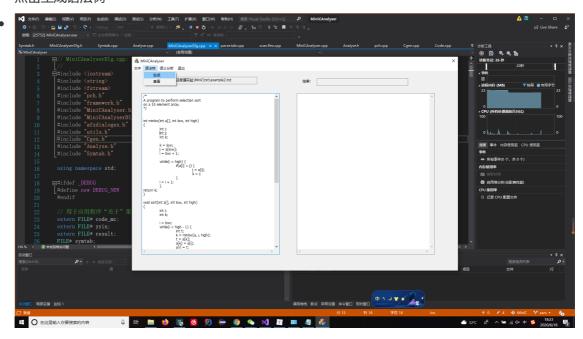
Windows (C:) DATA (D:) 🕳 本地磁盘 (E:) RECOVERY (F:)

文件名(N): win_bison_custom_build.targets

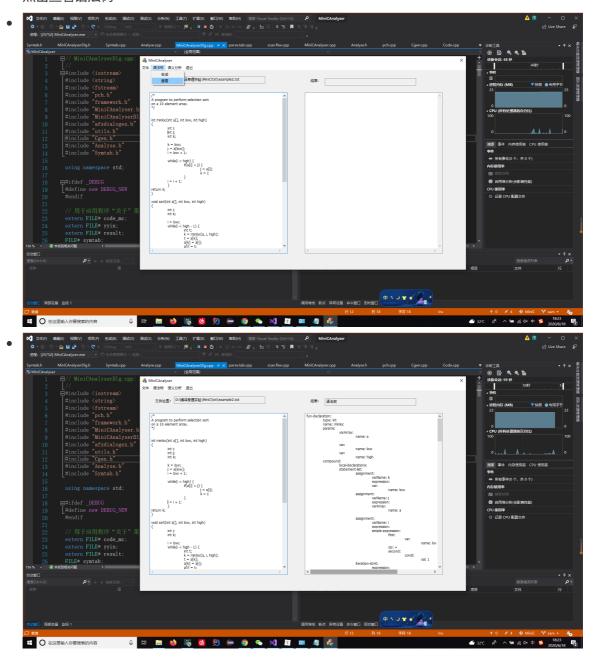
● 选择打开的文件, example{X}.txt都是符合MiniC语法的,选择example2.txt,这是排序的程序, 兼容了MiniC的全部语法



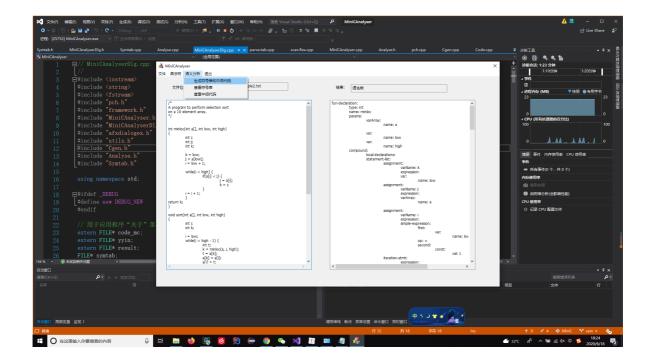
• 点击生成语法树



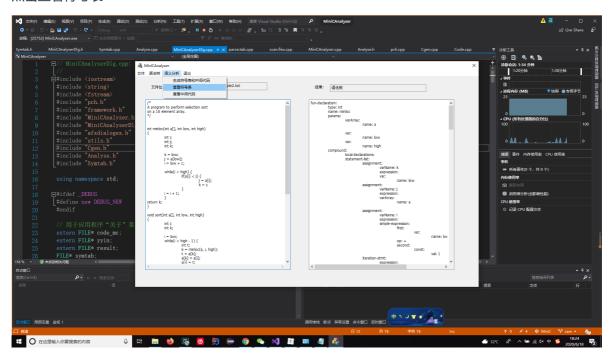
• 点击查看语法树

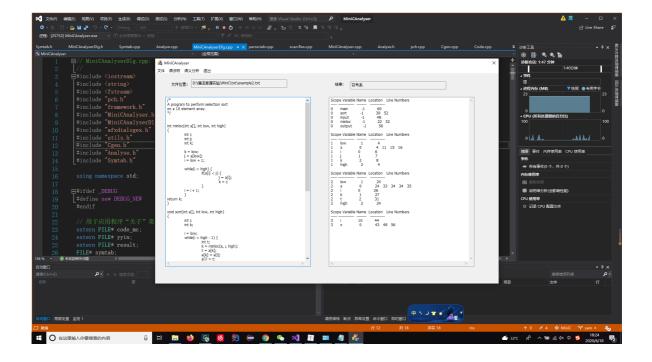


点击生成符号表和中间代码

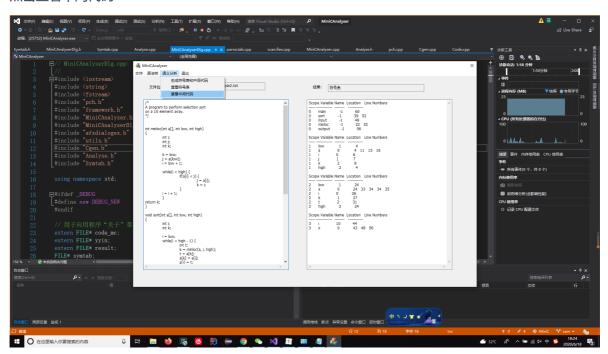


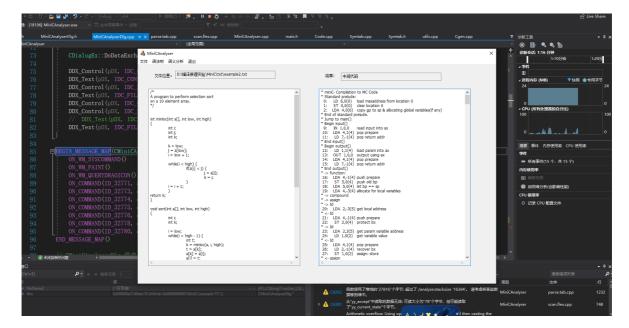
点击查看符号表





点击查看中间代码

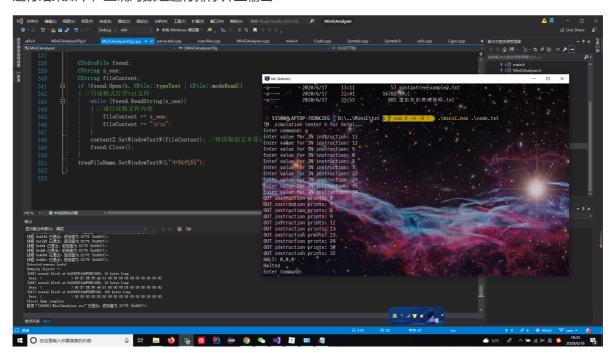




中间代码保存在txt文件夹的code.txt里面

在命令行进行项目目录的txt文件夹,执行命令 .\miniC.exe .\code.txt 使用虚拟机来运行中间代码如果miniC.exe 执行不了,参考虚拟机的使用说明

运行结果如下,正确对数组进行排序并且输出



txt文件夹里有五个测试用例,分别是example1.txt 到 example5.txt,相应的语法树、符号表、中间代码以及虚拟机的运行结果都在实验报告的第三部分里有展示

程序清单

utils.cpp // 生成节点,打印语法树等工具的实现scan.flex.cpp // 由 lex 生成的 cpp 文件parse.tab.cpp // 由 yacc 生成的 cpp 文件parse.y // yacc定义文件scan.l // lex 定义文件Symtab.cpp // 查询插入符号表的cpp文件Analyse.cpp // 构建符号表的cpp文件Code.cpp // 记录指令的cpp文件Cgen.cpp // 生成中间代码的cpp文件