Lab1 report

191250034 高灏

文件结构

1. main.c 主程序
2. lexical.l flex 文件
3. syntax.y bison 文件
4. STree.h 定义了语法树节点 `STnode_t`
5. Token.h 定义了`Token`信息 `token_info_t` 作为词法分析器的输出 `YYSTYPE`
6. Common.h 定义了一些公用宏

词法分析

通过正则表达式识别词法单元（注意顺序）

定义了宏 `YY_USER_ACTION` 用于处理位置信息，`NOMAL_ACTION` 用于为词法单元创建语法树节点（`STnode_t`）并添加相应的词法信息（`token_info_t`）。

通过正则表达式 `.` 识别词法错误，`yylex()` 返回 `YYUNDEF` 并设置全局变量 `lexical_error` 为 `1`。当语法分析器发现无法匹配时，通过 `lexical_error` 的值判断是词法错误还是语法错误。

语法分析

通过产生式识别语法单元

取负运算不需要指定优先级

将语法树构建过程加入宏 `YYLLOC_DEFAULT(Cur, Rhs, N)`(https://www.gnu.org/software/bison/manual/html_node/Location-Default-Action.html)（注意当 `N = 0` 时产生式规约为 ϵ，也要构建对应的语法树节点）。

通过 `yyr1[yyn]` 获取当前 `symbol` 编号(https://chenyuzhuwhiskey.github.io/bison-parser-analysis/)。

打印语法树时，通过 `yysymbol_name`(https://www.gnu.org/software/bison/manual/html_node/Syntax-Error-Reporting-Function.html#index-yypcontext_005ft)取得语法单元名称。

实验心得

lexical.l 中一上来认为没有其他需求所以使用了拓展性差的每个action复制黏贴的方式，然而实际上做到后面发现是有拓展需求的，要添加token信息。希望下次能想到整个要做的内容，想到后面要构建语法树。“想清楚再动手。”

遇到问题，实在想不明白的话直接花长时间看 log， 而不是化更长时间看代码空想。不要嫌看 log 麻烦。

吐槽

希望以后届的实验能使用 bison 的 `c++` 框架，感觉必须要用 yyr1[yyn] 这些 bison 手册上没有的东西，说明框架可能有些过时。

实验手册附录上第123页的“小数点的前后必须有数字出现”有歧义，不知道是前and后还是前or后，同时按照GNU的`c`规范是允许`.5`和`5.`的。

鸣谢

感谢王子鉴同学提供的测试数据。