Stage-2 实验报告

吴垒 2020010916

实验内容

仅需将__init__.py、my_parser.py 复制进原框架，并完成 my_parser.py 中的所有 TODO 即可。包括 p_relational，完全模仿 p_equality；p_logic_and，完全模仿 p_logic_or；p_assignment，先调用 p_conditional 匹配可匹配部分，这里不仅匹配了 conditional，同时也可匹配 Identifier '=' expression 的 Identifier，之后匹配 Assign（=），再匹配右侧 expression；p_expression，直接调用 p_assignment 即可；p_statement，仅需补全匹配 if、return 关键字部分即可；p_declaration，补全有初值时匹配 init_expr 的部分即可；p_block，循环判断部分及对 first 集合的判断部分助教已经写好，仅需补全匹配 statement 或 declaration 的部分；p_if，按照 if 块的文法规范依次匹配括号、expression 即可；p_return，仅需匹配关键字、返回内容及分号；p_type，目前只有 int 型变量，因此变量类型定义只会匹配 int。

思考题

Parser-stage：1.EBNF 转化为 LL(1)

addictive : addictive ‘+’ multiplicative | addictive ‘-’ multiplicative | multiplicative
消除左递归:

addictive : multiplicative X

X : ‘+’ multiplicative X | ‘-’ multiplicative X | ε

无需消除左公因子，已经是 LL(1)文法。

Parser-stage：2.错误恢复示例

如以下代码段：（testcase step6 else.c 添加了一个‘=’）

```
int main() {
    int a = 0;=
    if (a)
        return 1;
    else
        return 2;
}
```

首先程序通过 p_program 完成 int main() { 的匹配，通过 p_block 开始对块语句内内容的匹配；在 p_block 内用 p_block_item 完成对 int a = 0 的匹配；之后匹配到‘=’，不属于 block_item 的 first 集合，故报错；但直到匹配到第一个‘;’才能结束对当前 block 块的匹配，因此 if 部分被忽略；继续解析，接下来对下一个块语句进行匹配，但由于 if 被忽略，else 也不属于 block_item 的 first 集合，故会继续报错，直到分号结束 p_block 匹配完毕，退回 p_program 匹配 RBrace，结束语法分析。

Vscode 给出的语法分析结果是第二行应输入一个表达式，第五行应输入一个语句，感觉应该符合对 my_parser 分析到错误后继续分析的猜测。

Parser-stage：3.框架分析

作业难度因为有了 TODO 而变得非常简单，是框架的优点，必须吹一波，助教是真的善良。

但正因为作业简单了，导致有些文法在做的过程中可能理解得不够透彻，单纯为了填空而填空。

感觉可能需要一个提前把所有文法规则列出来的文件，以及适当讲解一下文法的产生方式，因为有些文法表述稍有迷惑性，比如"return : 'return' expression ';'"是否第二个 return 的首字母应该大写，因为是终结符；又如 expression：assignment 这一表达式，是不是为了方便理解才出现 expression？感觉其实所有 expression 都可以直接用 assignment 替代。

借鉴内容

实验思路指导与问答墙