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