编译原理 Project 简单说明

基本要求:

从文本文件中读取一个上下文无关文法的算符文法，构建算符优先分析表并以文本文件形式输出

实验文件读入简单说明:

终结符：简化为所有大写字母

非终结符：ASCALL 码（33-64 以及所有小写字母）其中除去'|'字符

且在读入的上下文无关文法中认为出现'$'为不合法。

产生式合法形式：Vn->Vn 与 Vt 的组合，其他形式将会被认为不合法

数据保存容器:

定义：**vector<string> array** 按行对读入文法进行保存

定义：**vector<pair<string,string>> str_array** 保存了去除"->"；并且按照'|'将一个产生式分为多个的字符串（如： T->E*F|F 则 str_array 保存了字符串:"TE*F"与"TF" 并且以 pair 形式保存，即<"T","E*F">与<"T" , "F">）

定义：**map<char,int> Vt, Vn** 将读入的非终结符保存在 Vn 中，读入的终结符保存在 Vt 中，方便通过字符直接俄找到其对应的标号；

定义：**FIRSTVT[MAX][MAX]，LASTVT[MAX][MAX]**来表示终结符与非终结符，终结符与终结符之间关系

定义：**relation_table[MAX][MAX]** 来表示算符优先关系并打印出算符优先分析表

定义：**vector<pair<char,int>> vec_map** 将 **map<char,int> Vt**，终结符放进向量中进行排序，方便打印

定义：**count_Vt, count_Vn** 代表终结符与非终结符的数量

主要函数:

```
bool isVt(char vt)：  用于判断产生式右部字符是否为终结符。
void make_first()：  用于生成 FIRSTVT；使用到栈
void make_last()：  用于生成 LATSTVT；使用到栈
void make_relation_table()：生成算符优先分析表并打印在输出窗口
Bool Compare (pair<char,int> v1,pair<char,int> v2)：将算符按照标号进行排序
```

样例文法（书本样例文法）以及实验输出结果：

E->E+T|T

T->T*F|F

F->(E)|i