.h文件在g++编译以后产生.h.gch

First Set 表示 非终结符可能产生的终结符（包含ε）

Follow Set 表示非终结符后面的东东（包含$）

例题： <https://www.jambe.co.nz/UNI/FirstAndFollowSets.html>

词法分析可以理解为进行分词，同时进行词法错误查找；语法分析可以理解为进行语法分块（表达式、语法结构），同时进行语法差错（比如需要考虑表达式的类型检查）。

传说词法分析不容错，但是是可以做到且应该做到容错的，因为基本上都是用空白符隔开了，把当前词加入报错列表然后开始下一个词的匹配即可。语法分析存在容错。容错即尽可能继续向下分析。词法分析的容错机制如果有了，那么语法分析怎么配合处理也是个麻烦。

”注释“的分析结构和语法分析里面很像，但是考虑到注释的作用（内部文字对程序无效），所以采用了更好的做法，加入到词法分析的DFA中，一块注释作为一个token分析。

语法分析的时候需要token list，你可以在scan（词法分析）的时候一个个传，也可以弄一个list存好（代码文件不会很大，所以不担心内存。使用list至少每一个元素都要有一个指向下一个元素的地址，占用4字节，但是token一般长度都不是很长，从利用率上讲不好，不过也算是一种不错的方法。没准使固定长度的char\*[],后来发现中间的字符串如果要记录下来就要不停malloc，所以我打算直接使用char[]，不同token之间使用空格隔开，但是我这种数组的方式，不合适记录行号），在词法分析之后一起传。我还见到封装出一个getToken()来获取之后的一个token，词法语法分析都能用，相当于边词法分析边传（推荐）。数组连着数组（每一个数组最后几位作为下一个数组的起始地址，如果没有下一个数组，需要留出4字节00）.