atitit.语法分析器 (Parser) 的原理

****语义分析与中间表示生成****

语义分析与中间表示生成的任务就是在语法分析的基础上，分析各语法单位的含义，并进行初步的翻译，即生成中间表示形式。有时，这两个任务是密不可分的，故通常将其合并为一个阶段讨论。语义分析主要是检查输入源程序的语义是否正确，例如，变量使用前是否定义、同一作用域内变量是否重名等。

中间表示的形式也非常多，包括四元组、三元组、语法树、DAG图等，并不一定是读者理解的通常的代码形式。例如，lcc的中间表示就是一种DAG的形式。当然，近似于汇编指令形式的四元组、三元组可能是最为常见的中间表示形

些编译器还包括两个非常重要的组成部分：符号表管理、出错处理。

****6．符号表管理****

符号表是一系列用于记录各个分析阶段所获取信息（如变量名、作用域、函数形参等）的数据表格，这些表格的维护贯穿于整个编译过程。显然符号表的设计和管理是编译器构造过程中的一项极其重要的任务。这里，设计者更多关注的是表格的完整性与访问效率。第4章将详细讨论符号表的构造。

****7．出错处理****

对于输入源程序的各种错误，编译器必须给出比较准确的出错报告，以便用户及时准确地定位、修改。编译过程的每一个阶段都可能检测出错误，其中，绝大多数错误可以在编译的前三个阶段检测出来。当然，真正的商用编译器并不限于此，可能还涉及更复杂的出错恢复。出错恢复主要是当编译器检测到错误之后，尽可能按照语义修正错误。注意，出错恢复的目的并不在于真正修复用户程序，而只是试图一次检测更多的错误。当然，这是基于编译器预测机制实现的