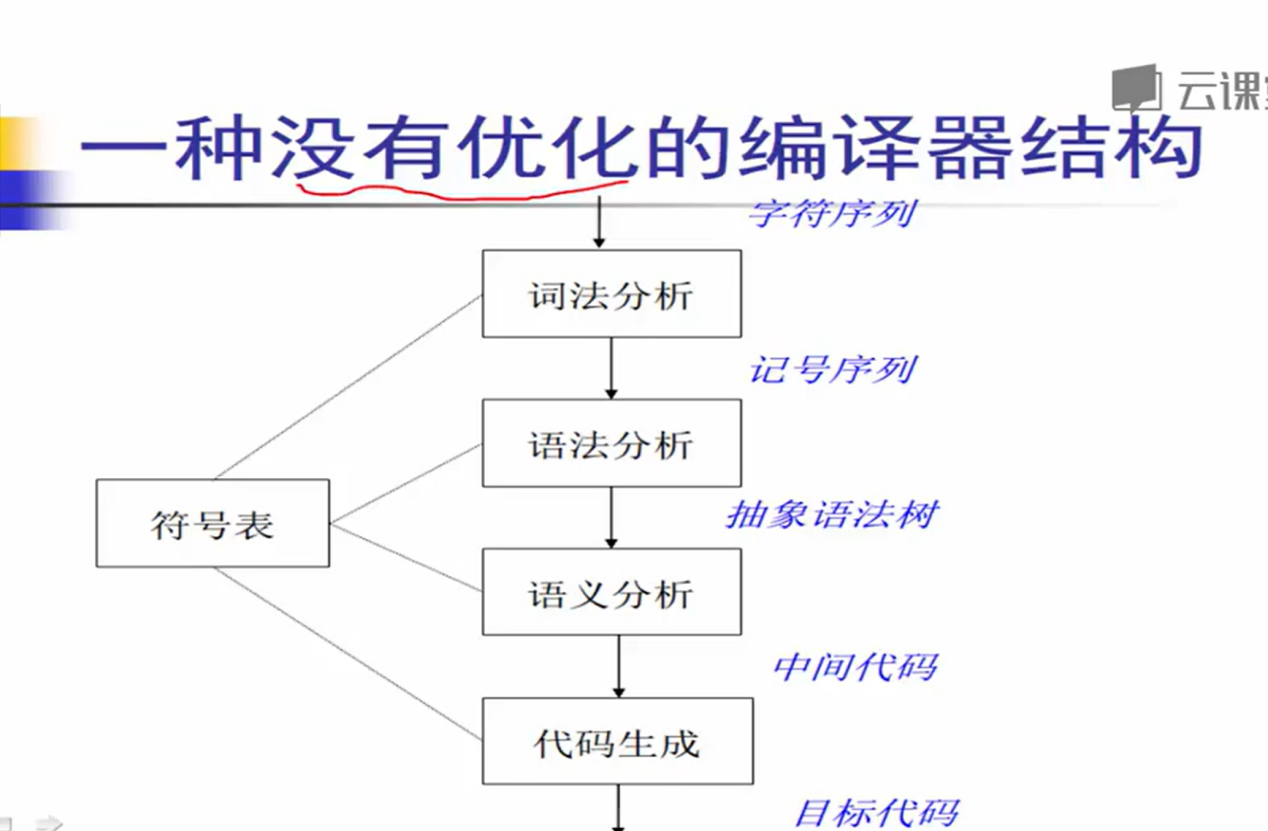
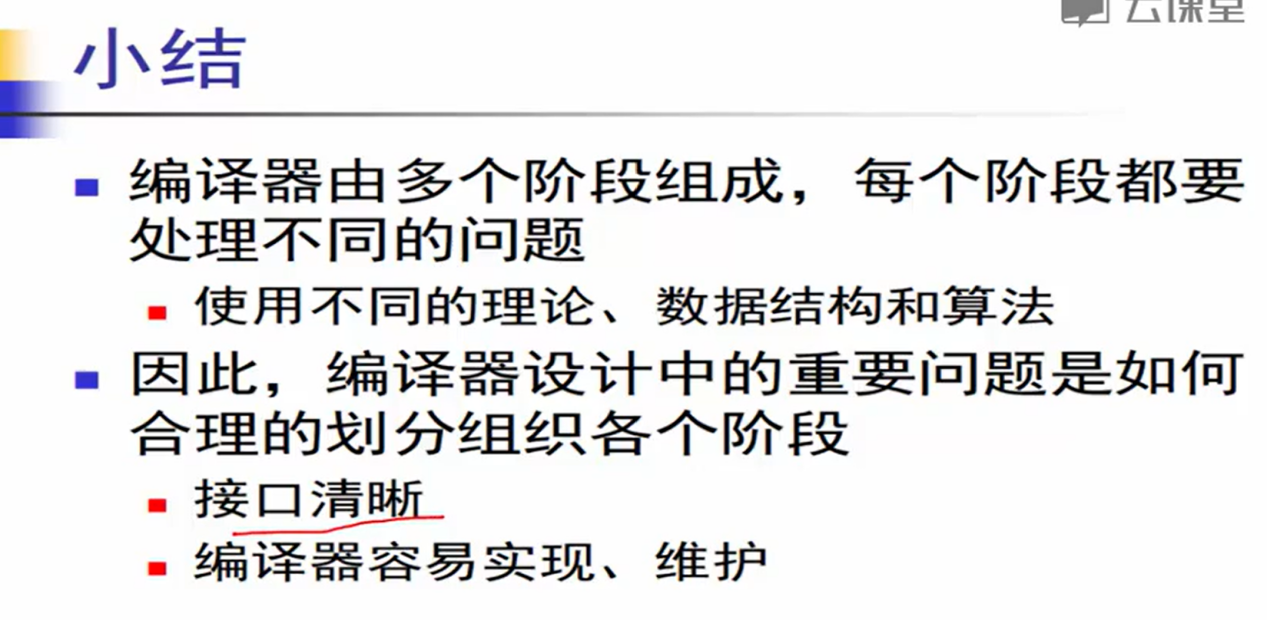
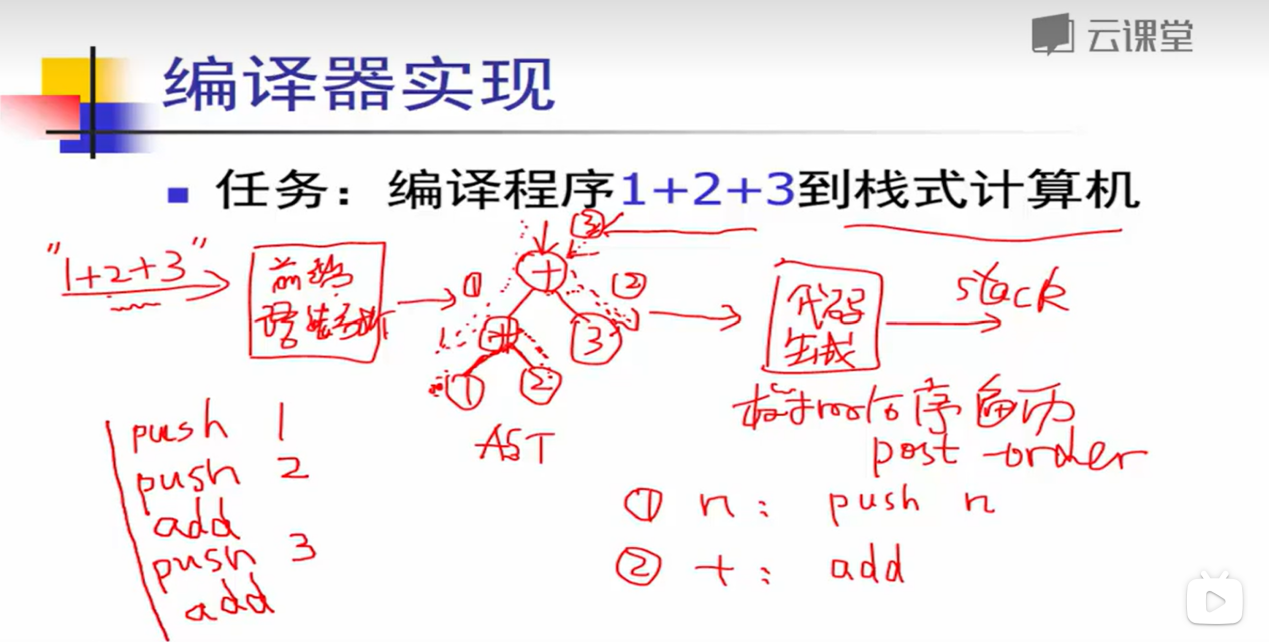


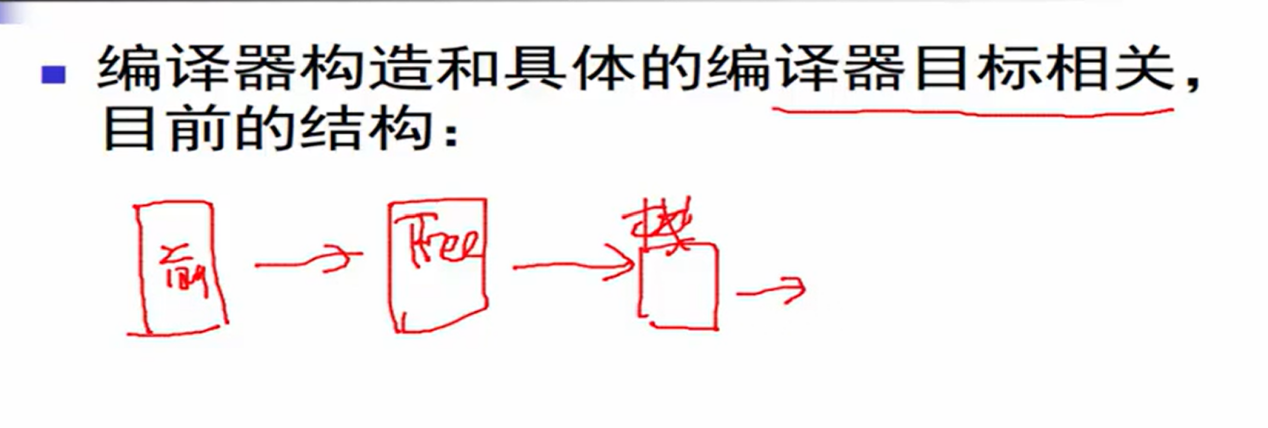
解释型直接生成结果无中间代码

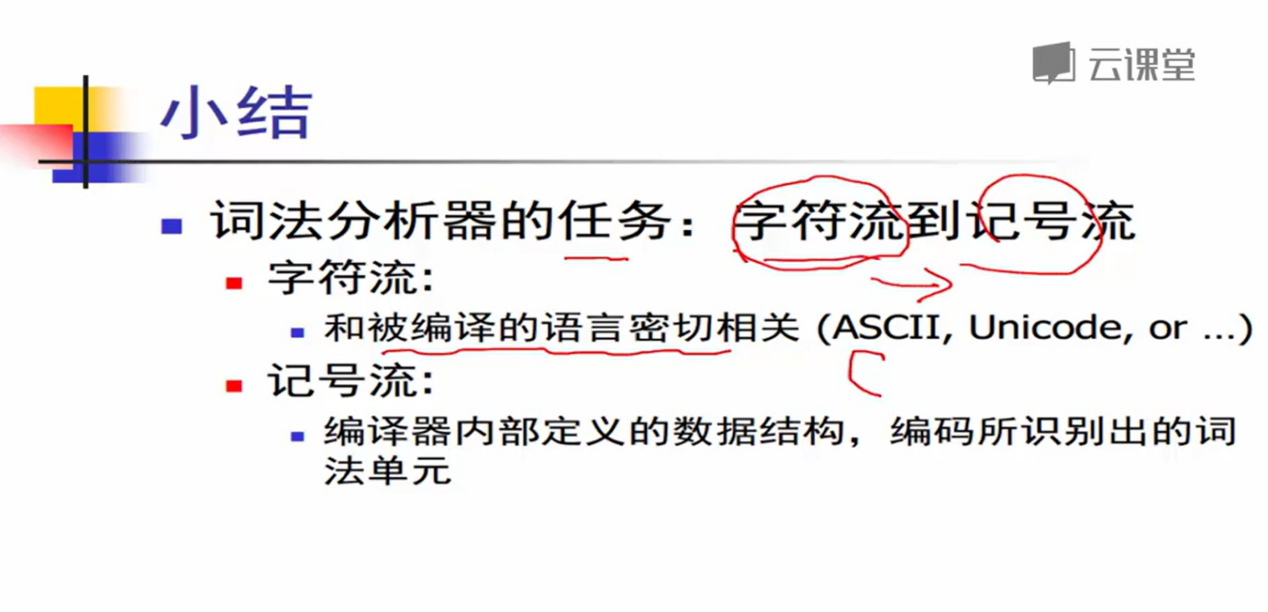






jvm是栈式计算机的原理





编译器、操作系统、硬件组成了软件系统

源程序与目标程序完全等价

词法分析：

1. 分词、正确拼词（例如==）
2. 关注单个符号出现的位置
3. 有限（穷举），无限（规则）

语法分析：

1. 关注符号的\*\*位置\*\*
2. 单词出现的\*\*位置\*\*

语义分析：

1. meaning是否符合规定
2. 检查是否有二义性
3. 检测上下文是否匹配

中间代码生成：

1.四元式

优化

1.内存空间时间开销减小

归约和推导互为逆过程

划分前端和后端提升效率，便于移植

## 第二章

主要围绕PL/0语言展开

1. PL/0语言的特点：

调用关系(动态)：可能存在条件判断，每次不一定执行

定义关系（静态）：

不能引用自己调用过程的变量

1. >Q main -> P 则main无法调用Q

引用变量不是在本过程定义为非局部变量

要使用栈管理过程调用

Load取 store存 call 只有这三个过程有level diference

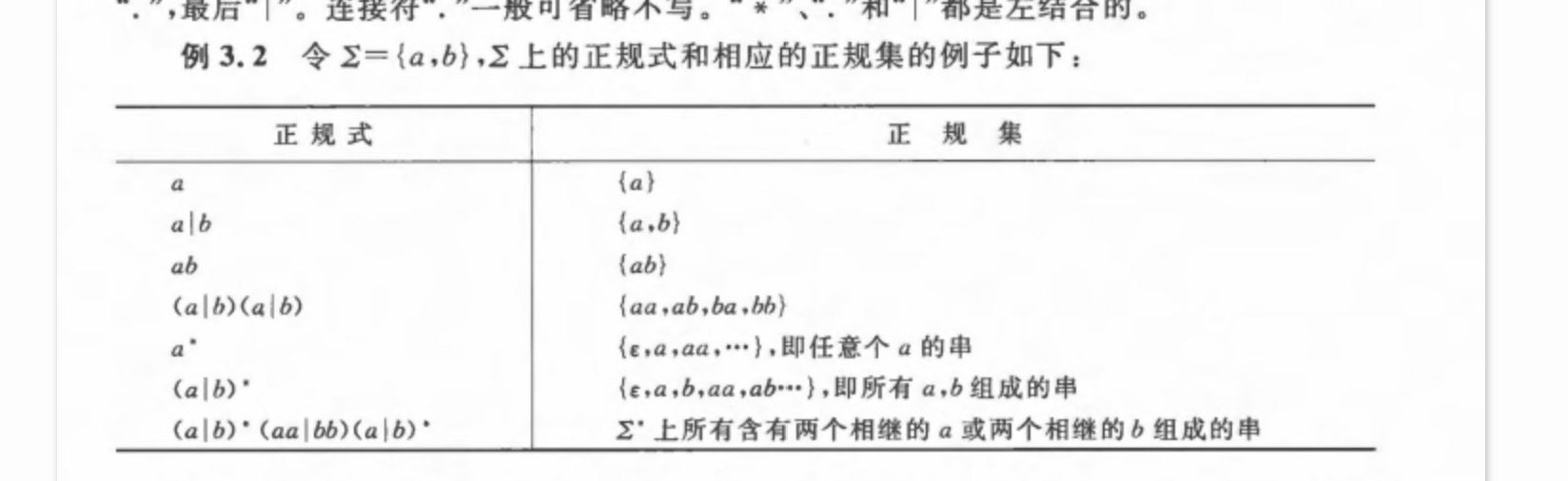
编译期和运行期

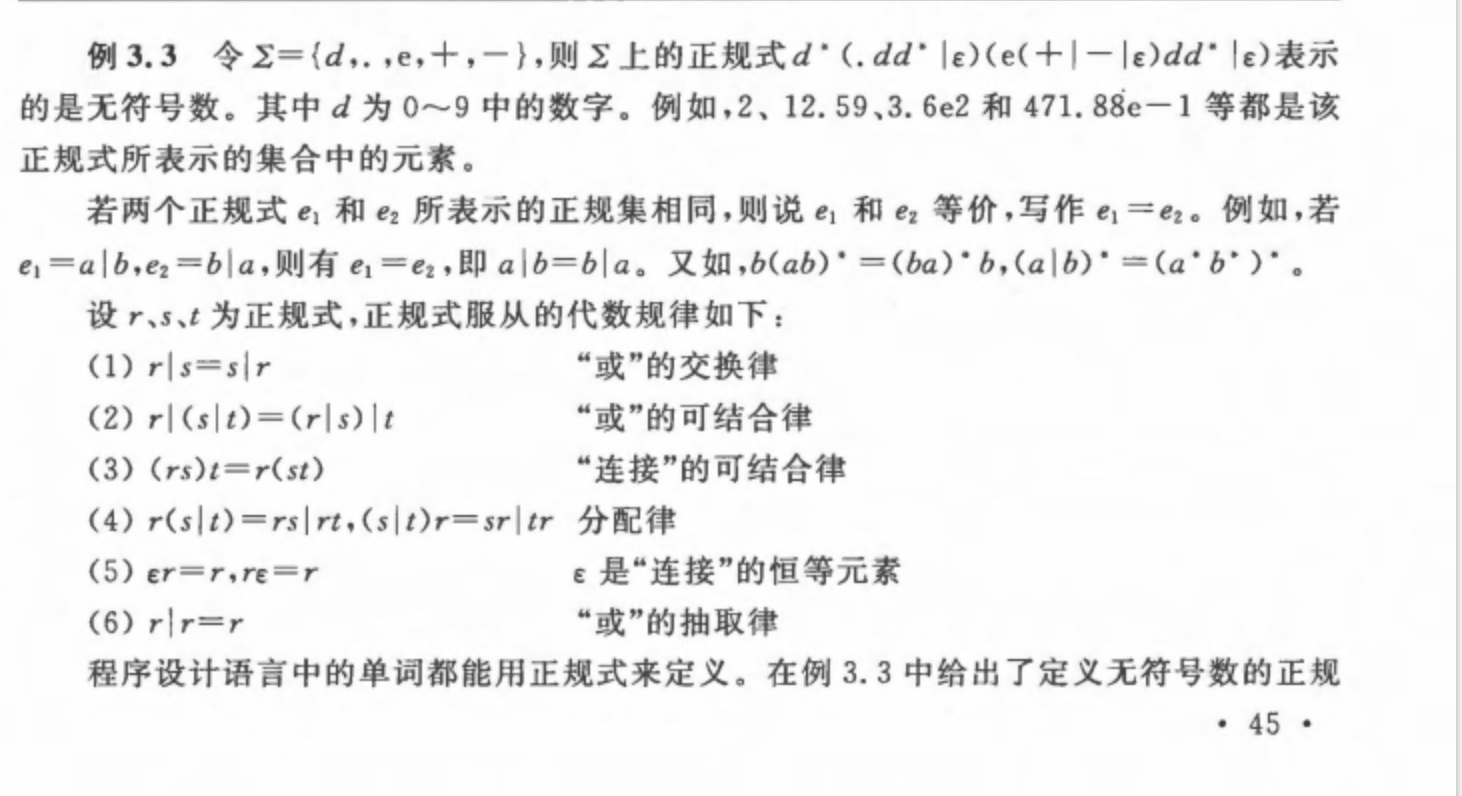
1. 听得懂
2. 看得懂
3. 做得对
4. 做得又快又对

## 第三章

1. 使用正规式表示

需要buffer,有很多需要看多一步的标识符如 ’>=’。‘+=’





语法语义分析器

词法分析器 getsym

代码生成 gem

不能在语法分析器中扩充，不能在block里面打印

改了#用<>作为不等于，那在用#就会报错

编译器里源代码的行号表示编译成的目标代码的行号范围，编译错误会在源代码对应错误的行号标注