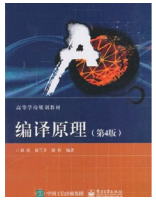
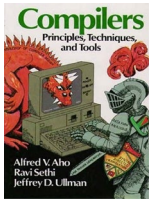
编译原理–第1章 引论

刘铭 骆婷 文明 张乾坤

参考书





理论：自编教材《编译原理》（第4版）

- 电子工业出版社

实验：许畅《编译原理实践与指导教程》

- 机械工业出版社

Alfred Aho 《Compilers: Principles, Techniques, and Tools》

- 人民邮电出版社

引论

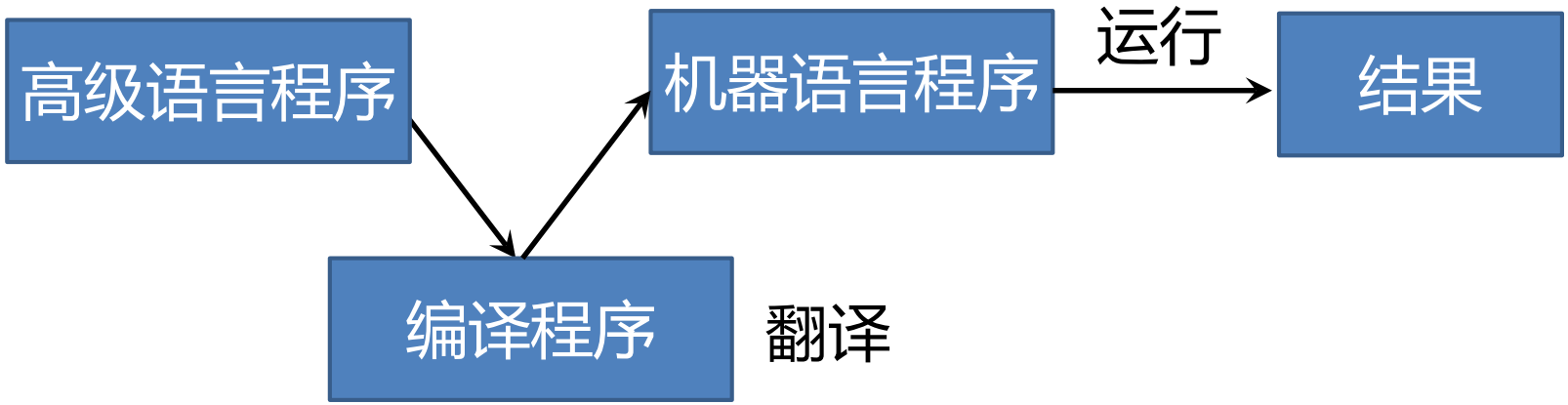
1. 什么是编译程序?
2. 为什么要学习编译原理?
3. 编译过程是什么?
4. 编译程序的结构如何?
5. 编译程序如何生成?

编译原理

1. 什么是编译程序?

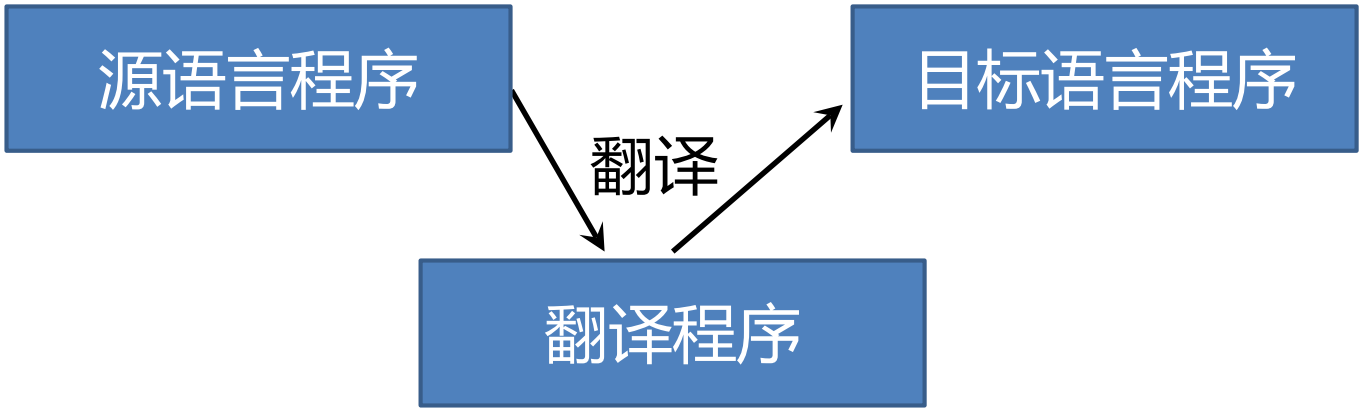
什么是编译程序?
—编译程序(Compiler)

- 把某一种高级语言程序等价地转换成另一种低级语言程序(如汇编语言或机器语言程序)的程序



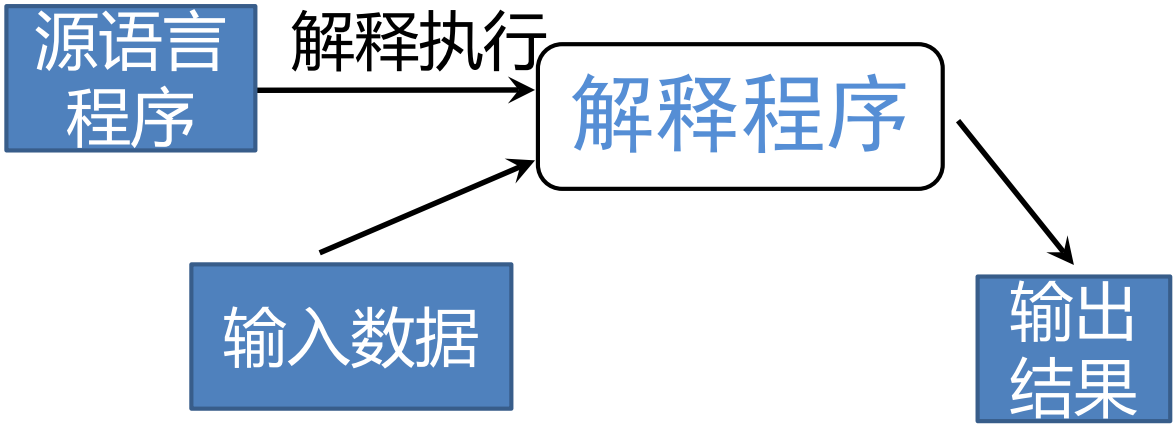
对比：翻译程序(Translator)

- 把某一种语言程序(称为源语言程序)等价地转换成另一种语言程序(称为目标语言程序)的程序

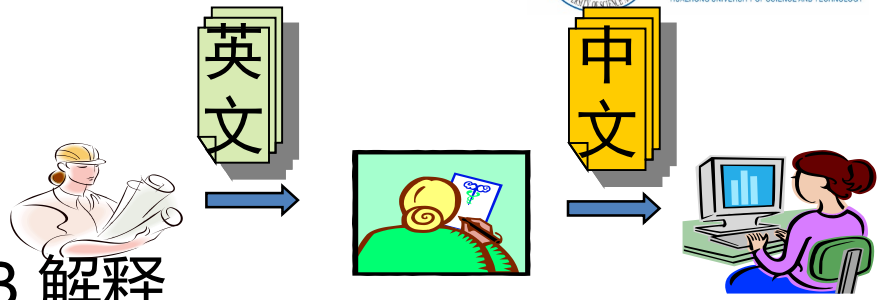


对比：解释程序(Interpreter)

- 把源语言写的源程序作为输入，但不产生目标程序，而是边解释边执行源程序



举例：编译vs. 解释



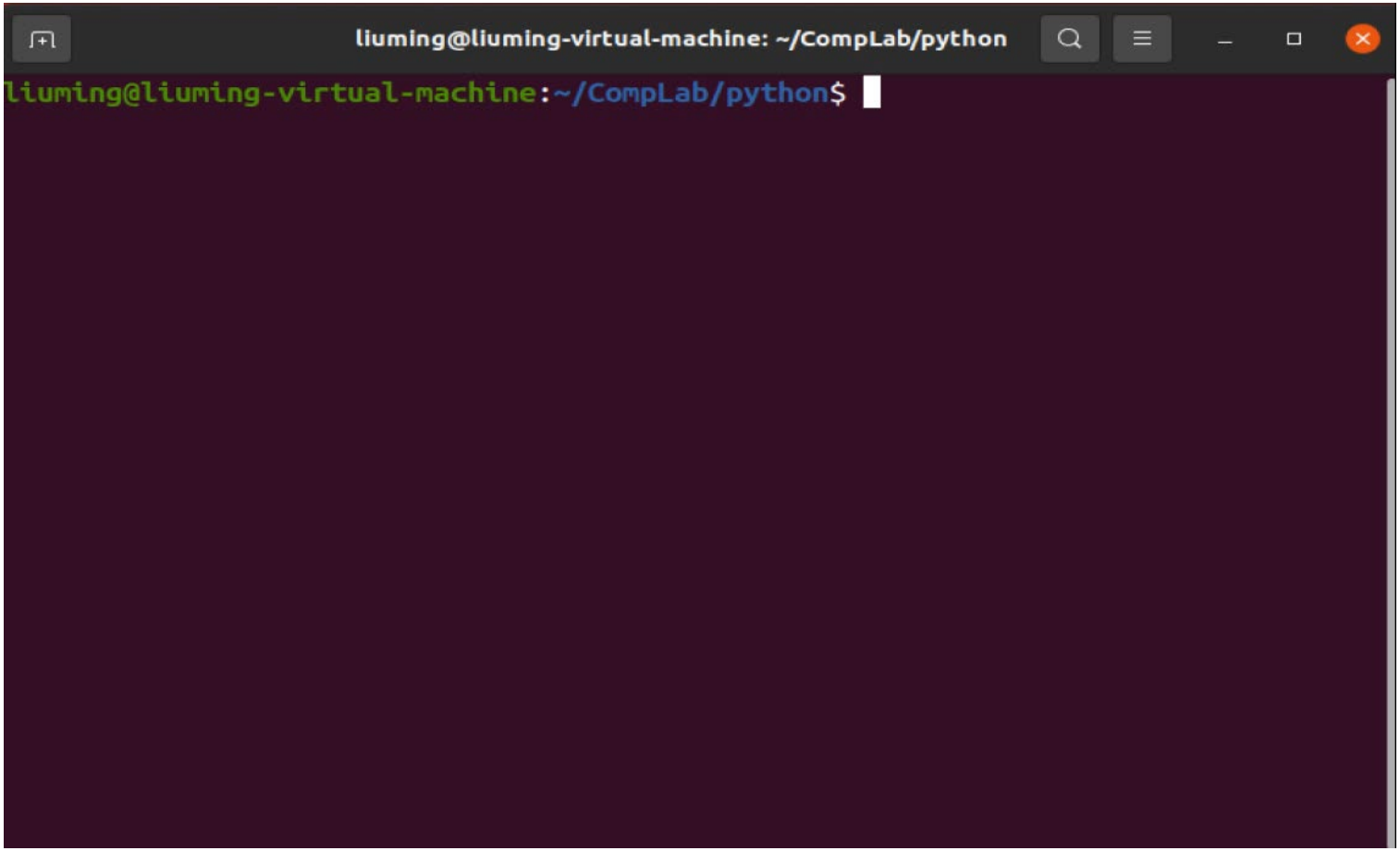
A.编译　　B.解释

```
liuming@liuming-virtual-machine: ~/CompLab/addc
liuming@liuming-virtual-machine:~/CompLab/addc$ ls -l
总用量 4
-rw-rw-r-- 1 liuming liuming 73 10月  7 11:20 add.c
liuming@liuming-virtual-machine:~/CompLab/addc$
```

gcc –E
gcc –S
gcc -c

编译 vs. 解释

Simultaneous Interpretation 同声传译



编译原理

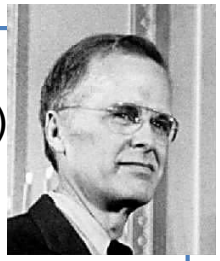
2. 为什么要学习编译原理?

为什么要学习编译原理?

1954 IBM 704

- Software > Hardware price
- 如何提高软件的产量？

John Backus 1924-2007



- **Speed coding** 1953 (10-20 slower,解释执行 300bytes=30%memory)
- FORTRAN 1 (Formulas Translation) 1954-1957
- 1958 50% (www.thocp.net)

FORTRAN1

- 第一个编译器 意义重大
- 诞生大量理论+实践
- 现代编译器仍遵循

为什么要学习编译原理?

- 编译理论与技术
  - 计算机科学与技术中理论和实践相结合的最好典范
- ACM 图灵奖
  - 授予在计算机技术领域作出突出贡献的科学家
  - 程序设计语言、编译相关的获奖者是最多的

Analysis of Algorithms **Artificial Intelligence** Combinatorial Algorithms Compilers
Computational Complexity Computer Architecture Computer Hardware
Cryptography Data Structures Databases Education Error Correcting Codes Finite Automata Graphics
Interactive Computing Internet Communications List Processing Numerical Analysis
Numerical Methods Object Oriented Programming Operating Systems Personal Computing
Program Verification Programming
Programming Languages Proof Construction Software **Theory** Software Engineering
Verification of Hardware and Software Models Computer Systems Machine Learning
Parallel Computation

http://amturing.acm.org/bysubject.cfm

为什么要学习编译原理?

与编译相关的ACM图灵奖获得者

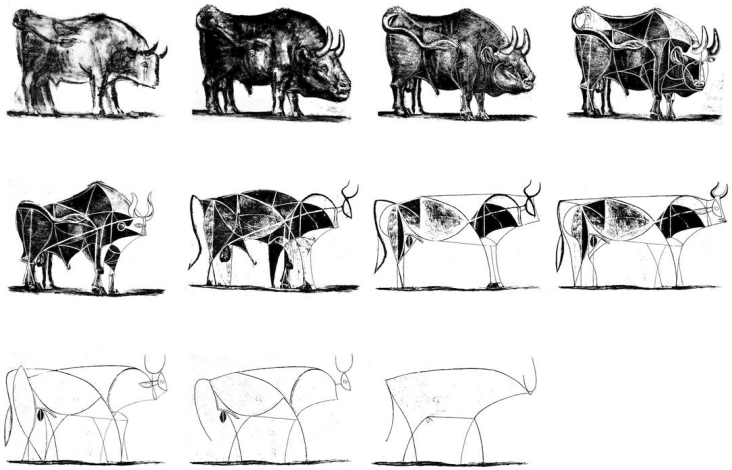
- Alan J. Perlis (1966) -- ALGOL
- Edsger Wybe Dijkstra (1972) -- ALGOL
- Michael O. Rabin & Dana S. Scott (1976) --非确定自动机
- John W. Backus (1977) -- FORTRAN
- Kenneth Eugene Iverson (1979) -- APL程序语言
- Niklaus Wirth (1984) -- PASCAL
- John Cocke (1987) -- RISC & 编译优化
- O. Dahl，K.Nygaard (2001) -- Simula语言和OO概念
- Alan Kay(2003) -- SmallTalk语言和面向对象程序设计
- Peter Naur(2005) -- ALGOL60以及编译设计
- Frances E. Allen(2006)-- 优化编译器
- Barbara Liskov(2008)--编程语言和系统设计的实践与理论

为什么要学习编译原理?

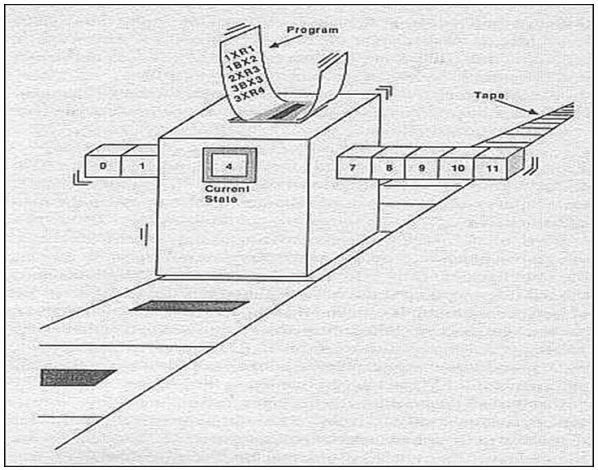
- 理解计算系统
- 设计计算系统
- 训练计算思维(Computational Thinking)
  - 编译理论与技术是计算机科学与技术中理论和实践相结合的最好典范、体现了很多典型的计算思维方法

为什么要学习编译原理?

- 编译原理中的"抽象"
  - 有限自动机
  - 形式文法
  - …



--毕加索《牛》



--图灵机

为什么要学习编译原理?

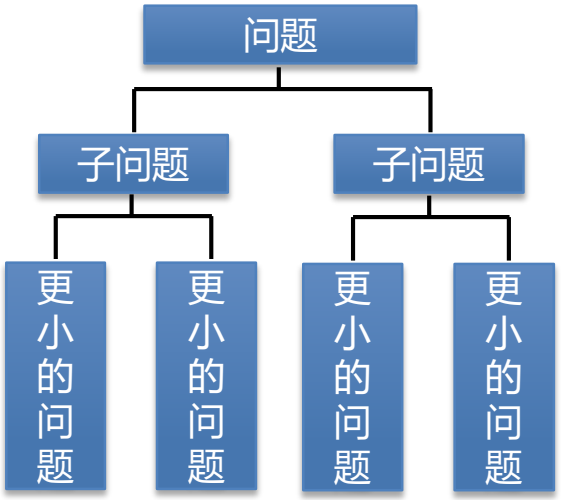
- 自动化(Automation)
  将抽象思维的结果在计算机上实现，是一个将计算思维成果物化的过程，也是将理论成果应用于技术的实践
- 编译原理中的"自动化“
  - ✓ 有限自动机
  - ✓ 预测分析程序
  - ✓ 算符优先分析
  - ✓ LR分析
  - ✓ …

控制程序

分析表

为什么要学习编译原理?

- 分解(Decomposition)
  - 将大规模的复杂问题分解成若干个较小规模的、更简单的问题加以解决
- 编译原理中的"问题分解"
  - 为什么编译分成多个阶段?
  - 为什么分析过程分成多遍?
  - 为什么编译程序引入中间语言?



为什么要学习编译原理?

- 递归(Recursion)
  - 问题的解决依赖于类似问题的解决，后者的复杂程度或规模较原来的问题更小
- 编译原理中的"递归"
  - 递归下降分析
  - 基于树遍历的属性计算
  - 语法制导翻译
  - …

为什么要学习编译原理?

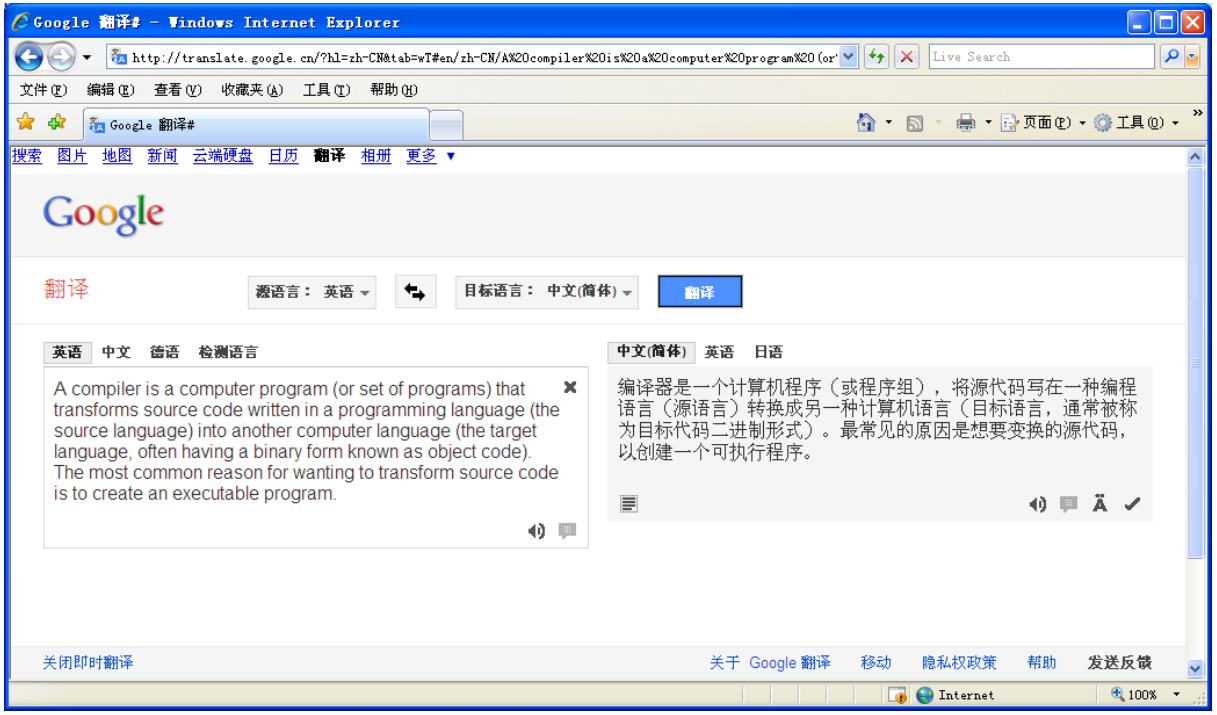
- 权衡(折衷，Tradeoff)
  - 理论可实现 vs. 实际可实现
- 编译原理中的"权衡"
  - 用上下文无关文法来描述和处理高级程序设计语言
  - 优化措施的选择
  - ...

为什么要学习编译原理?

- 编译原理体现了一系列计算机科学的思维方法
  - 提高对计算机系统总体认识
  - 感悟、训练计算思维
  - 更好地理解高级语言
  - 运用编译原理和方法构造实用工具
  - 享受计算之美、用“计算”的眼光看世界
  - 用计算解决实际问题

为什么要学习编译原理?
--编译原理和方法应用广泛

- 自然语言翻译



为什么要学习编译原理?
--编译原理和方法应用广泛

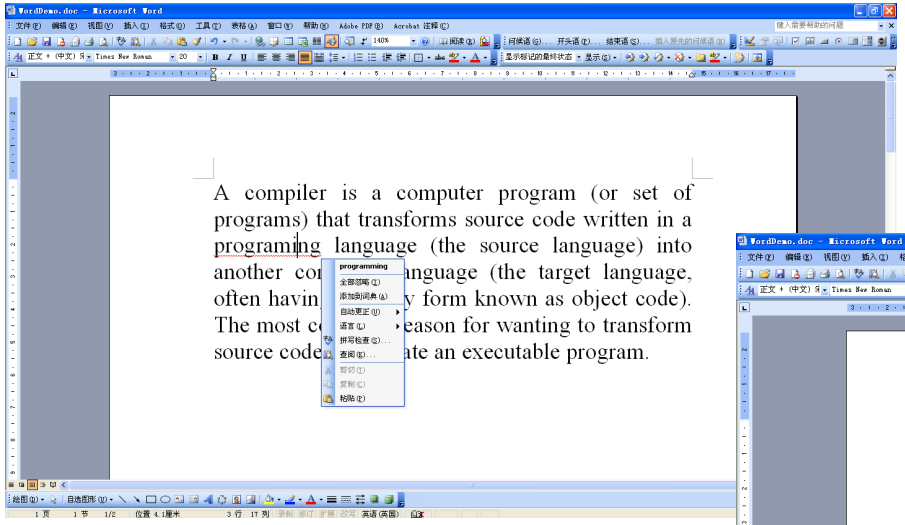
- Web应用中使用的HTML/XML语言分析

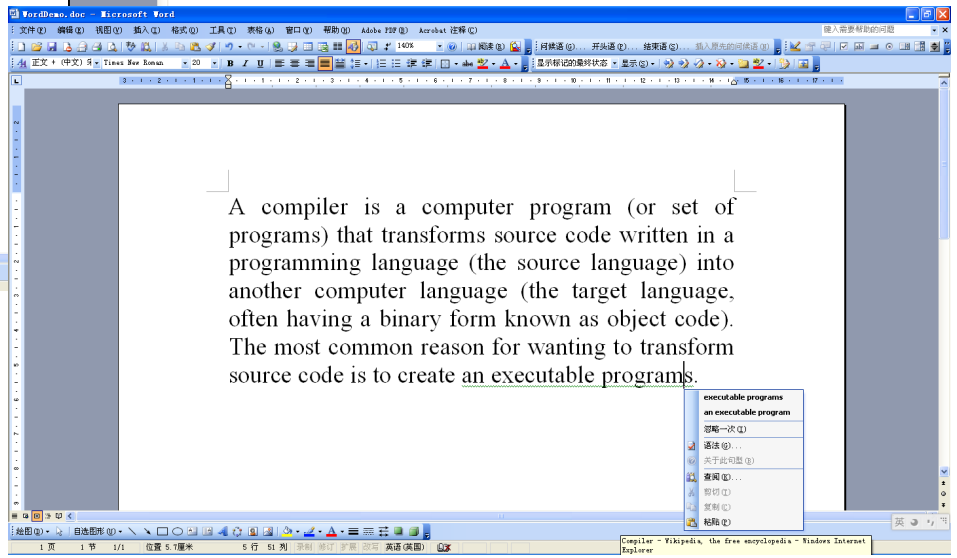

```
<HTML><HEAD><TITLE>HTML 语言教程</TITLE>
<META http-equiv=Content-Type content="text/html; charset=gb2312">
<META content="MSHTML 6.00.6000.16705" name=GENERATOR></HEAD>
<BODY text=#000000 bgColor=#ffffff>
<H1 align=center><FONT size=+3>HTML 语言教程：目录</FONT></H1>
<P>
<HR>

<P></P>
<CENTER>
<P><A href="http://www.gzsums.edu.cn/webclass/html/new.html">新的特色</A> *** <A
href="http://www.gzsums.edu.cn/webclass/html/tag-index.html">HTML
标记(Tag)的索引(Index)</A> </P></CENTER>
<CENTER><A
href="http://www.gzsums.edu.cn/webclass/html/page.html">页面(Page)</A>     <A
href="http://www.gzsums.edu.cn/webclass/html/font.html">字体(Font)</A>     <A
href="http://www.gzsums.edu.cn/webclass/html/text_style.html">文字布局(Text
Style)</A>     <A
href="http://www.gzsums.edu.cn/webclass/html/image.html">图象(Image)</A>     <A
href="http://www.gzsums.edu.cn/webclass/html/form.html">表单(Form)</A>
<P><A href="http://www.gzsums.edu.cn/webclass/html/table.html">表格(Table)</A>     <A
href="http://www.gzsums.edu.cn/webclass/html/table02.html">表格进阶(Advanced)</A>     <A
href="http://www.gzsums.edu.cn/webclass/html/frame.html">多窗口页面(Frames)</A>
<P><A
href="http://www.gzsums.edu.cn/webclass/html/marquee.html">会移动的文字(Marquee)</A>     <A
href="http://www.gzsums.edu.cn/webclass/html/inline.html">多媒体页面(Alternative
Inline Elements)</A> </CENTER>
<P>
```

为什么要学习编译原理?
--编译原理和方法应用广泛

- 自然语言处理（词法、语法）





为什么要学习编译原理?
--编译原理和方法应用广泛

- 搜索引擎

为什么要学习编译原理?
--编译原理和方法应用广泛
课堂练习1：

- 请列举在哪些计算机技术中应用了语言的翻译或变换?

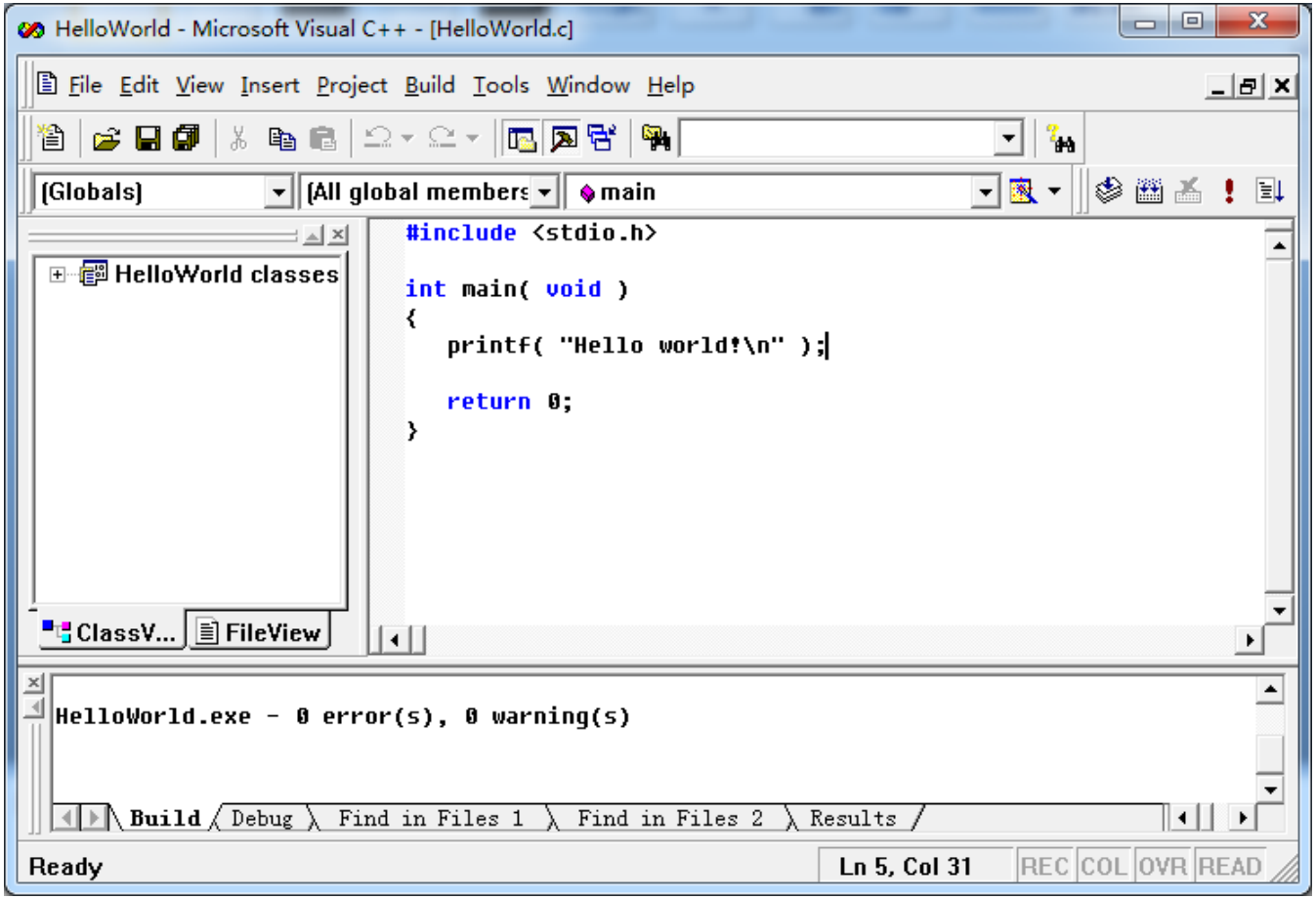
用户接口：Shell命令解释器， …
查询语言：SQL，XQuery， …
网络协议：HTTP， SOAP，…
...

实现领域特定语言（DSL)，HTML、SQL，Verilog、MATLAB、latex，Markdown
eg：实际中美团、阿里ToB业务中优惠券计算优惠组合

编译原理

3. 编译过程是什么？

例子：Hello World！程序是如何被翻译成机器语言或者汇编程序的？



编译过程是什么？

The compiler can translate a program from source language to target language.

编译程序能够把一个程序从源语言翻译成目标语言。

- 把英文翻译为中文
  - 识别出句子中的一个个单词
  - 分析句子的语法结构
  - 根据句子的含义进行初步翻译
  - 对译文进行修饰
  - 写出最后的译文

编译程序工作的五个阶段

- 词法分析
- 语法分析
- 中间代码产生
- 优化
- 目标代码产生

编译过程是什么？——词法分析

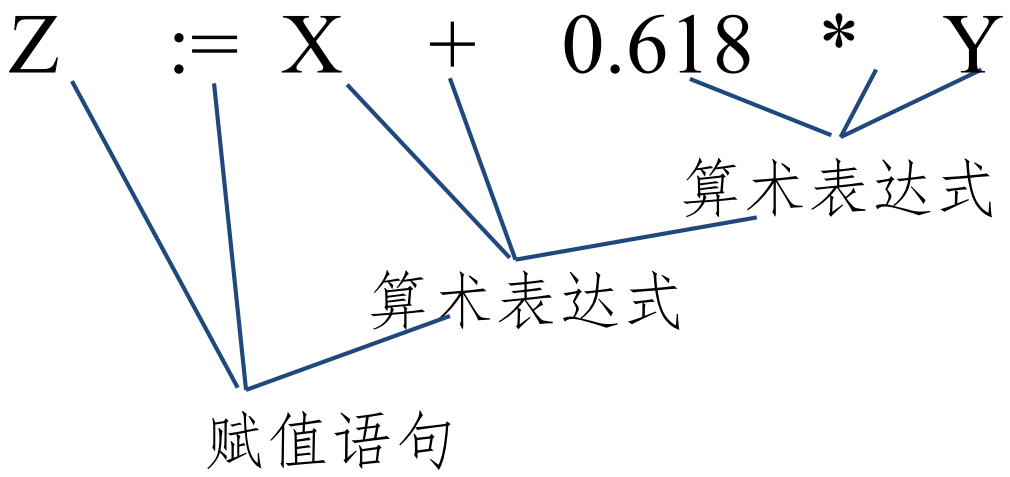
- 任务：输入源程序，对构成源程序的字符串进行扫描和分解，识别出单词符号
- 依循的原则：构词规则
- 描述工具：有限自动机

For	**K**	**:=**	**1**	**To**	**100**	**do**
基本字	标识符	赋值号	整常数	基本字	整常数	基本字

float r, h, s;　　s = 2*3.1416 * r *(r + h);

编译过程是什么？——语法分析

- 任务：在词法分析的基础上，根据语法规则把单词符号串分解成各类语法单位(语法范畴)
  - 依循的原则：语法规则
  - 描述工具：上下文无关文法



编译过程是什么？——中间代码产生

- 任务：对各类语法单位按语言的语义进行初步翻译
  - 依循的原则：语义规则
  - 描述工具：属性文法
  - 中间代码：三元式，四元式，树，...
  - 例子：Z:=X + 0.618 * Y翻译成四元式为

序号	OPR	OPN1	OPN2	RESULT	注释
(1)	*	0.618	Y	T_1	T_1:=0.618*Y
(2)	+	X	T_1	T_2	T_2:=X+T_1
(3)	:=	T_2		Z	Z:=T_2

编译过程是什么？——优化

- 任务：对前阶段产生的中间代码进行加工变换，以期在最后阶段产生更高效的目标代码
  - 依循的原则：程序的等价变换规则
  - 例子：

```
FOR K:=1 TO 100 DO
BEGIN
     X:=I+1;
     M := I + 10 * K;
     N := J + 10 * K;
END
```

编译过程是什么？优化示例

```
FOR K:=1 TO 100 DO
BEGIN
     X:=I+1;
     M := I + 10 * K;
     N := J + 10 * K;
END
```

序号	OPR	OPN1	OPN2	RESULT	注释
(1)	:=	1		K	K:=1
(2)	j<	100	K	(10)	if (100<K) goto (10)
(3)	+	I	1	X	X:=I+1
(4)	*	10	K	T_1	T_1=10*K
(5)	+	I	T_1	M	M:=I+T_1
(6)	*	10	K	T_2	T_2=10*K
(7)	+	J	T_2	N	N:=J+T_2
(8)	+	K	1	K	K:=K+1
(9)	J			(2)	goto (2)
(10)					

统计？次加法
？次乘法

400次加法
200次乘法

编译过程是什么？
优化后的等价代码

```
FOR K:=1 TO 100 DO
BEGIN
     X:=I+1;
     M := I + 10 * K;
     N := J + 10 * K;
END
```

序号	OPR	OPN1	OPN2	RESULT	注释
(1)	+	I	1	X	X:=I+1
(2)	:=	I		M	M:=I
(3)	:=	J		N	N:=J
(4)	:=	1		K	K:=1
(5)	j<	100	K	(10)	if (100<K) goto (10)
(6)	+	M	10	M	M:=M+10
(7)	+	N	10	N	N:=N+10
(8)	+	K	1	K	K:=K+1
(9)	J			(5)	goto (5)
(10)					

301次加法

编译过程是什么？——目标代码产生

- 任务：把中间代码变换成特定机器上的目标代码
  - 依赖:硬件系统结构和机器指令的含义
- 目标代码三种形式
  - 汇编指令代码：需要进行汇编
  - 绝对指令代码：可直接运行
  - 可重新定位指令代码：需要连接

编译过程是什么？——目标代码产生

- 例：

  b=a+2

```
MOV a, R1
ADD #2, R1
MOV R1, b
```

两遍扫描

操作码	寄存器操作数	操作数标志	第二操作数/地址
0001	01	00	00000000 *
0011	01	10	00000010
0100	01	00	00000100 *

重定位

L=00001111

0001	01	00	00001111
0011	01	10	00000010
0100	01	00	00010011

汇编器符号表

标识符	地址
a	0
b	4

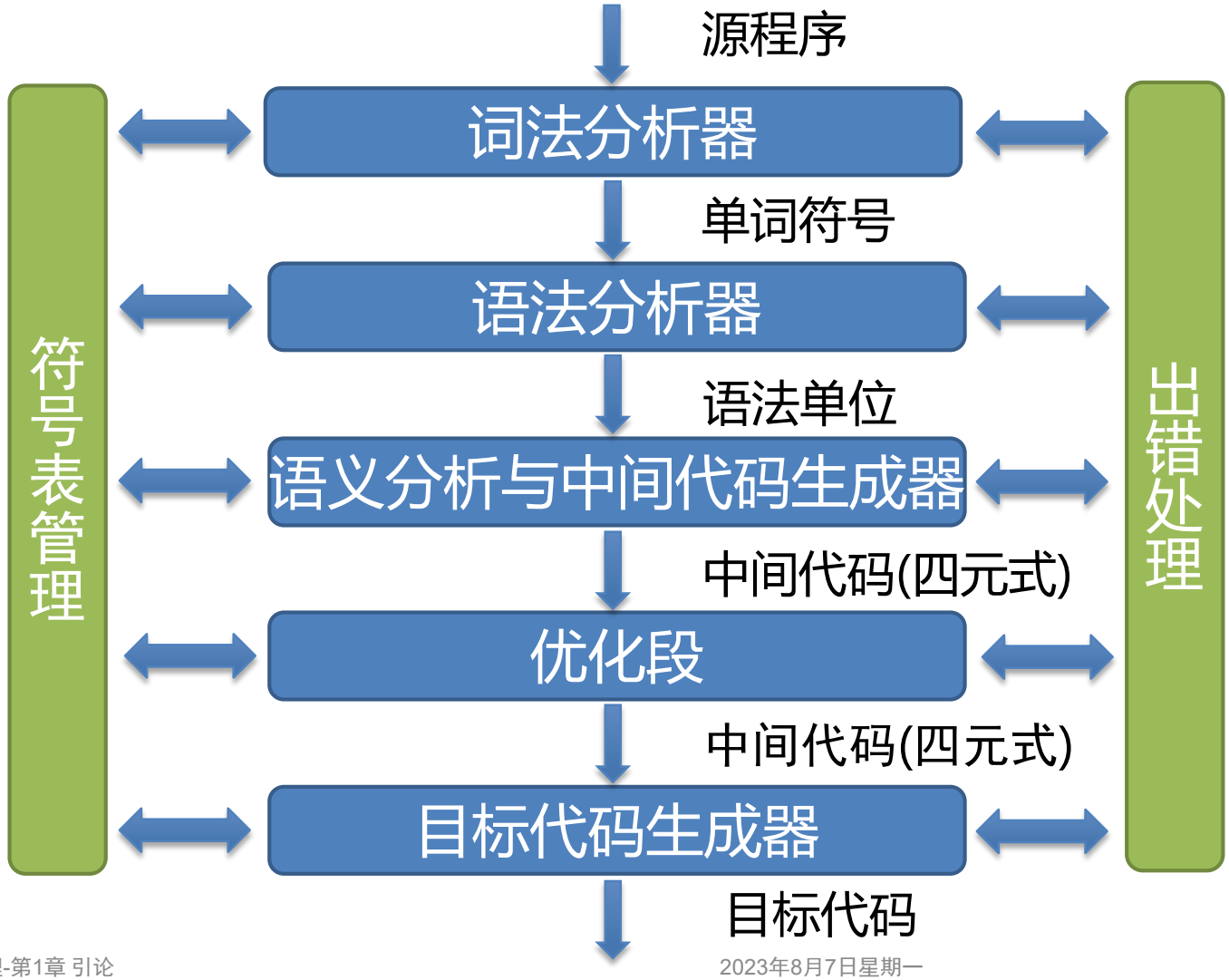
课堂练习

- 可以直接运行的目标代码是（）
  - A. 汇编指令代码
  - B. 可重新定位指令代码
  - C. 绝对指令代码

编译原理

4. 编译程序的结构如何？

编译程序的结构如何？--框架



编译程序的结构如何？--例子

词法分析

语法分析

语义分析

中间代码生成

代码优化

目标代码生成

position :=init+rate*60

Id1:=id2+id3*60

```
   :=
id1      +
     id2      *
          id3    60
```

```
   :=
id1      +
     id2      *
          id3    inttoreal
                      60
```

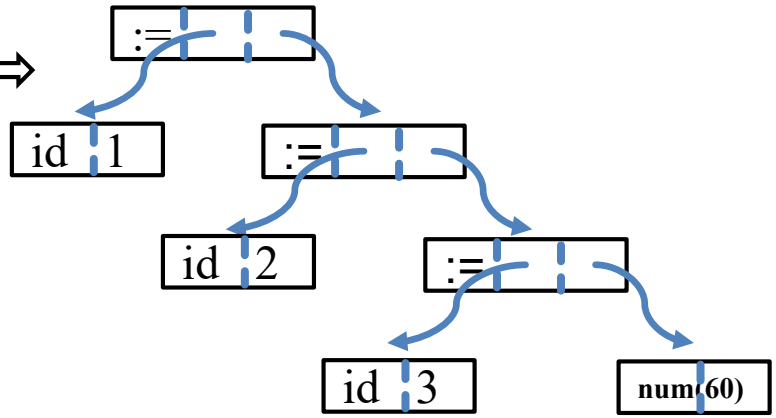
```
temp1:=inttoreal(60)
temp2:=id3*temp1
temp3:=id2+temp2
id1:=temp3
```

```
temp1:=id3*60.0
id1:=id2+temp1
```

```
MOVF id3,R2
MULF #60.0,R2
MOVF id2,R1
ADDF R2,R1
MOVF R1,id1
```

符号表

1	position	…
2	init	…
3	rate	…
4		



编译程序的结构如何？--出错处理

- 出错处理程序
  - 发现源程序中的错误，把有关错误信息报告给用户
- 语法错误
  - 源程序中不符合语法（或词法）规则的错误
  - 非法字符、括号不匹配、缺少 ；、...
- 语义错误
  - 源程序中不符合语义规则的错误
  - 说明错误、作用域错误、类型不一致、...

编译程序的结构如何？
什么是遍(pass)

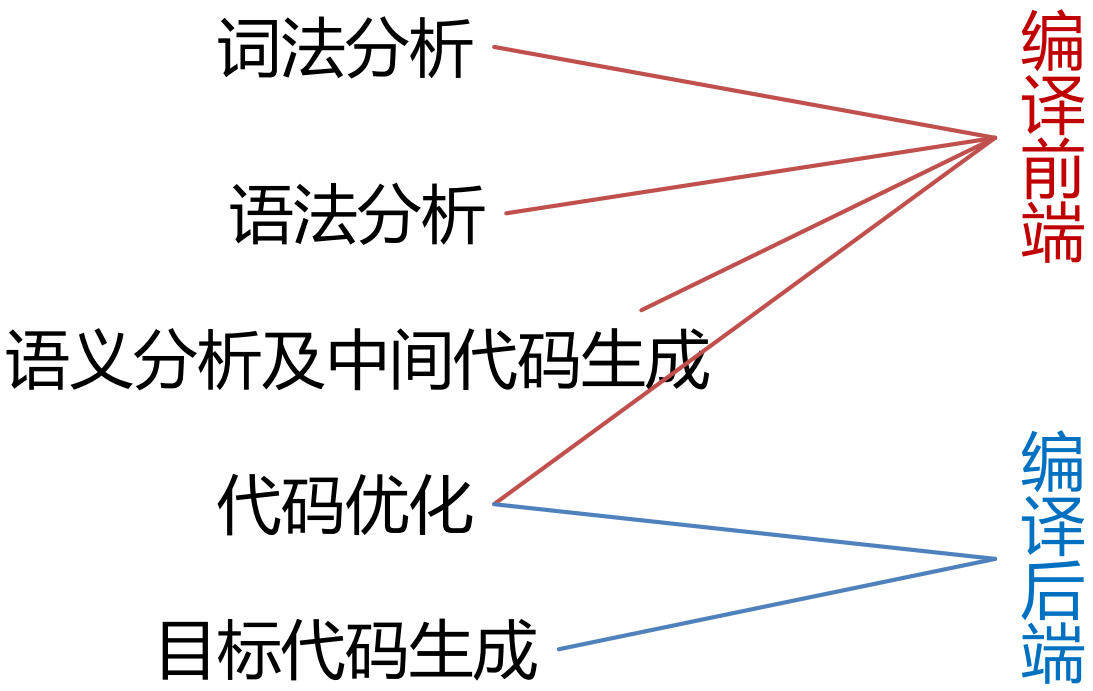
- 所谓"遍"， 就是对源程序或源程序的中间表示从头到尾扫描一次
- 阶段与遍是不同的概念
  - 一遍可以由若干段组成
  - 一个阶段也可以分若干遍来完成

编译程序的结构如何？
编译前端与后端

- 编译前端
  - 与源语言有关，如词法分析、语法分析、语义分析与中间代码产生，与机器无关的优化
- 编译后端
  - 与目标机有关，与目标机有关的优化、目标代码产生
- 优点：程序逻辑结构清晰、优化更充分，有利于移植

课堂练习

- 请指出编译过程和编译程序结构之间的关系



编译原理

5. 编译程序如何生成？

编译程序如何生成？

- 用汇编语言/机器语言编写编译器
  - 优点： 可以针对具体的机器，充分发挥计算机的系统功能；生成的程序效率高
  - 缺点： 程序难读、难写、易出错、难维护、生产的效率低
- 用高级语言编写
  - 程序易读、易理解、容易维护、生产的效率高

编译程序如何生成？

- 交叉编译：在某种机器上运行的编译器可能为另外一种机器生成目标代码。

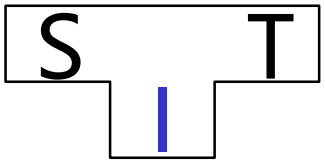


图1 T型图（Bratman[1961]）
S：编译器要编译的源语言
T：编译器要产生的目标程序
I：编译器所用的实现语言
缩写为$S_I T$

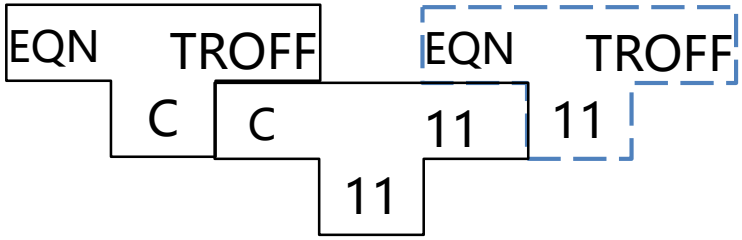
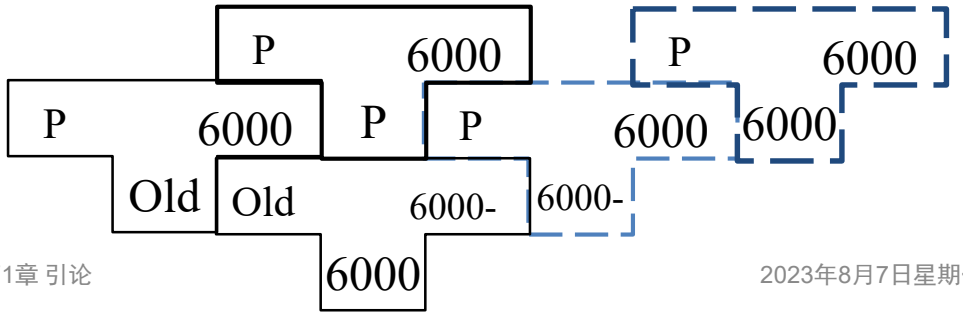


图2 交叉编译

EQN编译器最初版本以C为实现语言，为文本格式化工具TROFF生成代码。在运行于PDP-11上的C编译器C_{11}11的基础上，用EQN_CTROFF得到运行于PDP-11上的EQN的交叉编译器。

编译程序如何生成？

- 自举(bootstrapping)
  - 使用语言提供的功能来编译该语言自身。
    - 为不断扩充的语言子集构建编译器。
    - 例子1：Pascal（60%）几次自举后得到整个Pascal编译器
    - 1972年CDC 6000系列机上Pascal自举。Old:“老”Pascal；P :修正后的Pascal语言，“-”:粗劣实现。修正版编译器用老版本编写，被翻译成拙劣编译器P_{6000-}6000。手工将P_{Old}6000翻译成P_P6000，再用前面的得到最终的清晰实现。



编译程序如何生成？

- 编译程序自动生成
  - 编译程序-编译程序，编译程序产生器，编译程序书写系统
  - LEX：词法分析程序产生器
  - YACC：语法分析程序产生器

小结

- 课程概述
  - 内容、意义
- 什么是编译程序
  - 翻译、编译、解释
- 学习编译原理的原因
- 编译的基本过程
- 编译程序的结构
  - 阶段、遍、前端/后端
- 编译程序生成的几种方法