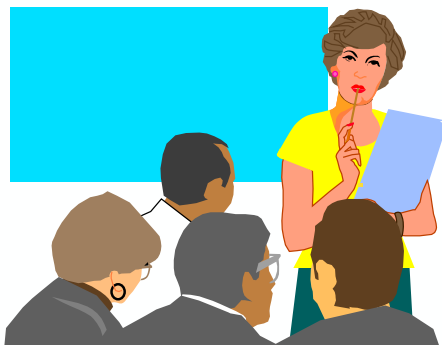


编译技术

编译原理及编译程序构造



杨海燕 蒋竞

2018.9-2019.1

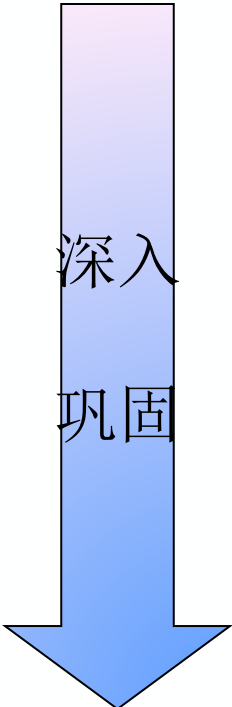
课程目的

目的:

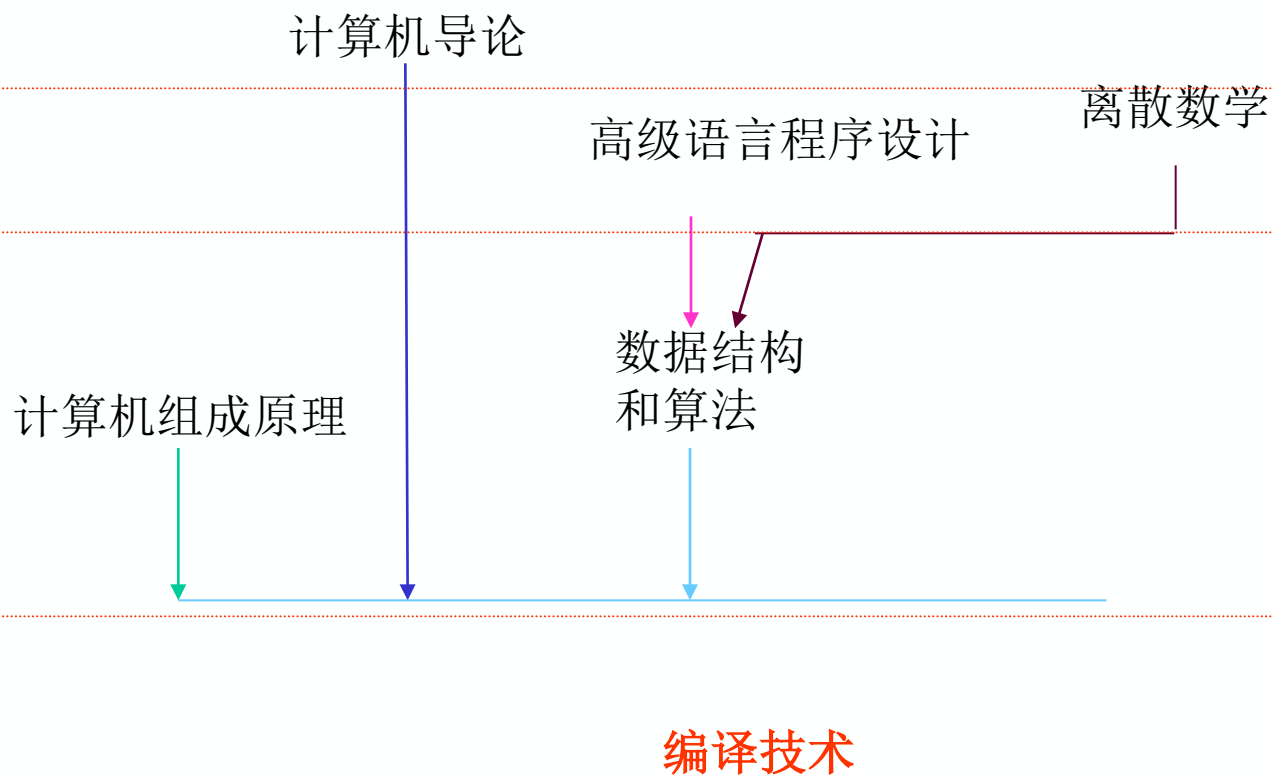
- 掌握编译的**基本理论**、常用的**编译技术**，了解编译过程及编译系统的构造（结构和机理）。
- 能运用所学技术解决实际问题，能独立编写一个小型编译系统。（**课程设计大作业，可选不同难度等级**）

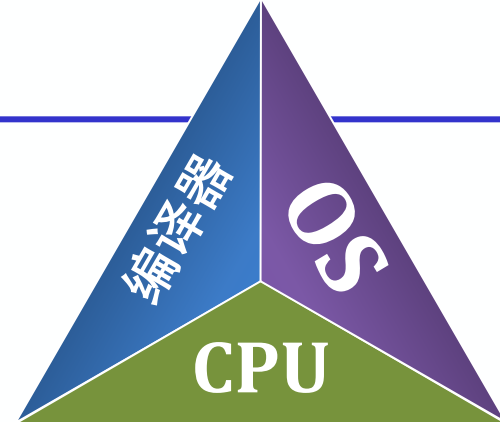
三次学习编译全过程

- 第一次：概述
 - 大致了解编译的过程和编译程序的构造
- 第二次：第3-10，14，15章
 - 详细学习各部分的原理和方法
- 第三次：课程设计
 - 自己动手实现一个完整编译器



深入
巩固





- 3大核心技术
 - CPU、操作系统和编译器
- 3大必修课程
 - 计算机组成、操作系统、编译技术
- 3个基本问题
 - 能够开发1个CPU吗？
 - 能够开发1个操作系统核心吗？
 - 能够开发1个编译器吗？

来自小鹏院长的ppt：结合学院本科培养目标



• 教材和参考书

- 张莉，史晓华，杨海燕、金茂忠，《编译技术》，高等教育出版社，2016,9
- 张莉，杨海燕，史晓华、金茂忠、高仲仪，《编译原理及编译程序构造》，清华大学出版社，2011,6。
- Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, Compilers—Principles, Techniques, and Tools. 机械出版社（翻译版）
- A. W. Apple, J. Palsberg 著，《Modern Compiler Implementation in Java》
- Kenneth C. Loudon 著，《编译原理及实践》，机械工业出版社
- 陈火旺，刘春林，谭庆平等 编著，《程序设计语言编译原理》，国防工业出版社出版

特点

- **经典课程：**
 - **国内外大部分学校都开设**
 - **历史悠久**
 - **有经典教材**
 - **重视实践教学，尤其是国内外重点大学**
 - **强调教学过程**

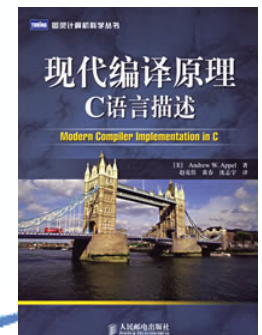
编译技术

- 国外大学：名称不一样：
 - **Compiler Design** : Carnegie Mellon
 - **Programming Languages and Compilers**:
University of Illinois–Urbana-Champaign,
University of California–Berkeley,
 - **Compilers and Interpreters** :Yale
 - **Compiler Design and Implementation** :Harvard
 - **Compiling Techniques**: Princeton
 - **Compilers**: Stanford
- 国内：编译原理、编译技术
- 内容差异不大。

Purdue

- ECE 468 Introduction to Compilers and Translation Engineering
- ECE 573 Compilers and Translation Systems
- ECE 495S Introduction to Compilers and Translation Engineering
- ECE 663 Advance Optimized Compilers
- EE 573 Compilers & Translator Writing Systems

- 1.**龙书(Dragon book)**：书名是Compilers: Principles, Techniques, and Tools；作者是：Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman；国内所有的编译原理教材基本都是参考了本书，重点是编译的前端技术。
- 2.**鲸书(Whale book)**：书名是：Advanced Compiler Design and Implementation；作者是：Steven S. Muchnick；也就是高级编译原理。
- 3.**虎书(Tiger book)**：书名是：Modern Compiler Implementation in Java/C++/ML, Second Edition；作者是：Andrew W. Appel, with Jens Palsberg。这本书是3本书中最薄的一本！



- **CMU** : 作业80% , 期中考试 20%
- **UIUC**:
 - 8 道题目 , 20%
 - 2 次期中考试 , 各占25%
 - 1 次期末考试 , 占30%
- **Berkeley** :
 - 编程作业 : 7个编程作业 , 30%
 - 考试 : 两次平时考试和一次期末考试 , 分别占10% , 10% , 30%
 - 问题集 (Problem sets) : 提交 “学习问题” , 15%
 - 参加讨论 : 根据出勤、讨论课和协作情况给分 , 5%
- **Priceton** :
 - 不完成高级项目 (total = 100%)
 - » 标准编程作业 (1-10): 50%
 - » 期中和期末考试 50%
 - 完成高级项目 (total = 100%)
 - 高级项目: $N\%$ ($0 < N < 50$; N 与难度有关.)
 - 标准编程作业(1-5): $(50-3N/4)\%$
 - 期中和期末考试: $(50-N/4)\%$

课程要求

★ 课时:48学时 (1—13周)

(一) 304

– 周一下午 : 2:00~3:35

– 周四上午 : 9:50-11:25

★ 分为两部分:(分别计分)

– 理论基础 (3学分) : 课堂教学, 按时交作业。

- 作业10分;
- 3-6次随堂考试, 共计30分; (缺考不补)
- 期末闭卷考试, 60分
- 主动回答问题, 每次奖励0.5分, 5分封顶(考前公布)

– 实践部分(1.5学分) : 上机实践(50机时)

• 第7~8周开始, 第18周期末考核

要求：

提前预习，上课认真听讲；
课后及时复习，独立认真完成作业。

课后辅导：

课程助教：

王 鑫 13051571125

郑嘉腾 15652909585

地点：新主楼G305

网址：<http://judge.buaa.edu.cn>

<http://www.icourses.cn/mooc/>（爱课程）

作业提交：

每周一、周四交作业

- 1、班长或者课代表负责收发作业，不要个人单独交
- 2、在第一节课前，将收好的作业放到讲台
- 3、助教会在第一节课后取本次作业，并将上次批好的作业放回讲台



编译技术

1、理论和实践相结合、基础性和前沿性相结合；2、强调编译过程的完整性，因材施教，要求每一个同学独立完成一个不同难度的完整编译系统。通过对每个学生进行面试答辩，严把质量关。3、依托北航计算机软件与理论的学术优势，科研和教学紧密互动，研制了编译课程教学辅助软件，开发了特色教学网站。

[▶ 开始学习](#)[▶ 参与课堂互动](#)

课程名称：编译技术

所属学校：北京航空航天大学

负责人：张莉

课程类型：理论课（含实践/实验）

课程属性：专业基础课/技术基础课

课程学时：48

学科门类：工学

专业类：计算机类

专业：计算机科学与技术

适用专业：计算机软件与理论、软件工程、计算机系统结构

国家级

♥ 收藏课程

★ 站内分享

★ 分享到：



0

课程介绍

课程指导思想及定位 “编译技术”是计算机软件学科的一门核心专业必修课程，是学生了解高级程序设计语言原理和编译相关技术的基础课，也是学生接触到的第一个完整的软件系统，在教学中具有重要地位。根据北航计算机学院的培养定位，我们要求学生既要掌握有关编译的经典基础理论，又要学会运用先进的软件开发技术构造实际编译系统的方法，因此这是一门理论和实践要求很高的课程，也是学生在本科学习阶段培养动手能力的一个非常重要...

[查看更多](#)

课程大纲



教学日历



考评方式与标准



学习指南

第一章 概论

(介绍名词术语、了解编译系统的结构和编译过程)

- 编译的起源：程序设计语言的发展
- 基本概念
- 编译过程和编译程序构造
- 编译技术的应用



1.1 程序设计语言的发展

机器语言
(机器指令)

汇编语言

面向用户
的语言

面向问题
的语言

C7 06 0000 0002 MOV x, 2

x = 2

低级语言

高级语言



- 低级语言 (Low level Language)
 - 字位码、机器语言、汇编语言
 - 特点：与特定的机器有关，功效高，但使用复杂、繁琐、费时、易出错
- 高级语言
 - Fortran、Pascal、C、Java 语言等
 - 特点：不依赖具体机器，移植性好、对用户要求低、易使用、易维护等。

用高级语言编制的程序，计算机不能立即执行，必须通过一个“翻译程序”加工，转化为与其等价的机器语言程序，机器才能执行。
这种翻译程序，称之为“编译程序”。

1.2 基本概念

- 源程序

用汇编语言或高级语言编写的程序称为源程序。

- 目标程序

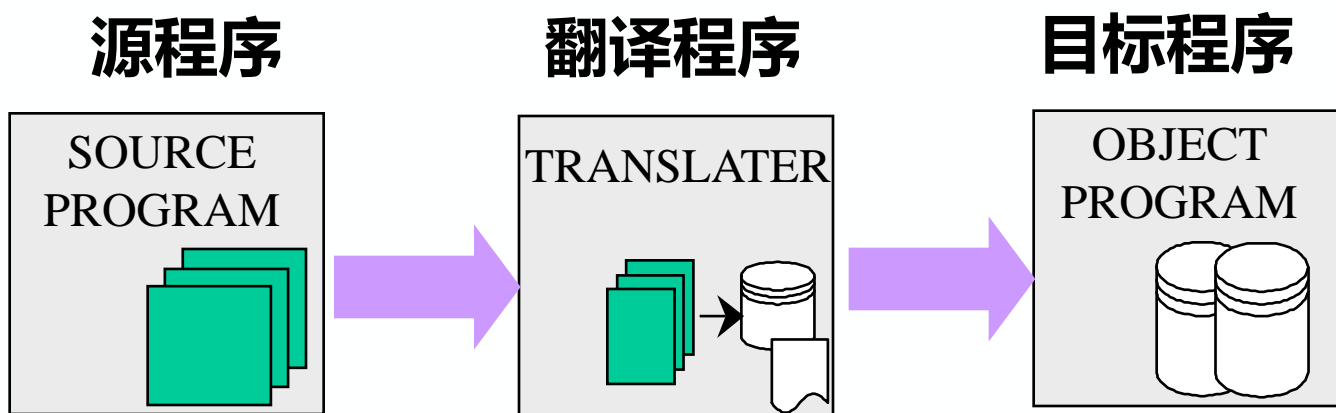
用**目标语言**所表示的程序。

目标语言：可以是介于源语言和机器语言之间的“中间语言”，可以是某种机器的机器语言，也可以是某机器的汇编语言。

- 翻译程序

将**源程序**转换为**目标程序**的程序称为翻译程序。它是指各种语言的翻译器，包括汇编程序和编译程序，是汇编程序、编译程序以及各种变换程序的总称。

源程序、翻译程序、目标程序 三者关系：



即源程序是翻译程序的输入，目标程序是翻译程序的输出

源程序

汇编语言
高级语言

翻译程序

汇编程序
编译程序

目标程序

机器语言
目标程序

• 汇编程序

若源程序用汇编语言书写，经过翻译程序得到用机器语言表示的程序，这时的翻译程序就称之为汇编程序，这种翻译过程称为“汇编”（Assemble）

• 编译程序

若源程序是用高级语言书写，经加工后得到目标程序，这种翻译过程称“编译”（Compile）

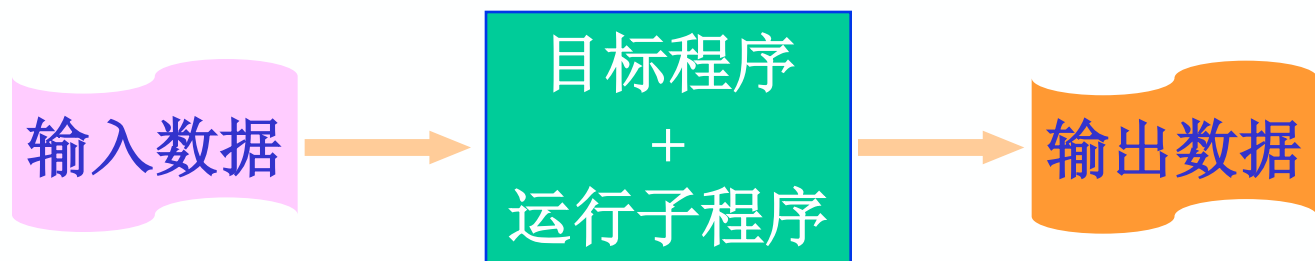
汇编程序与编译程序都是**翻译程序**，主要区别是加工对象的不同。由于汇编语言格式简单，常与机器语言之间有一一对应的关系，汇编程序所要做的翻译工作比编译程序简单得多。

源程序的编译和运行

- 编译或汇编阶段

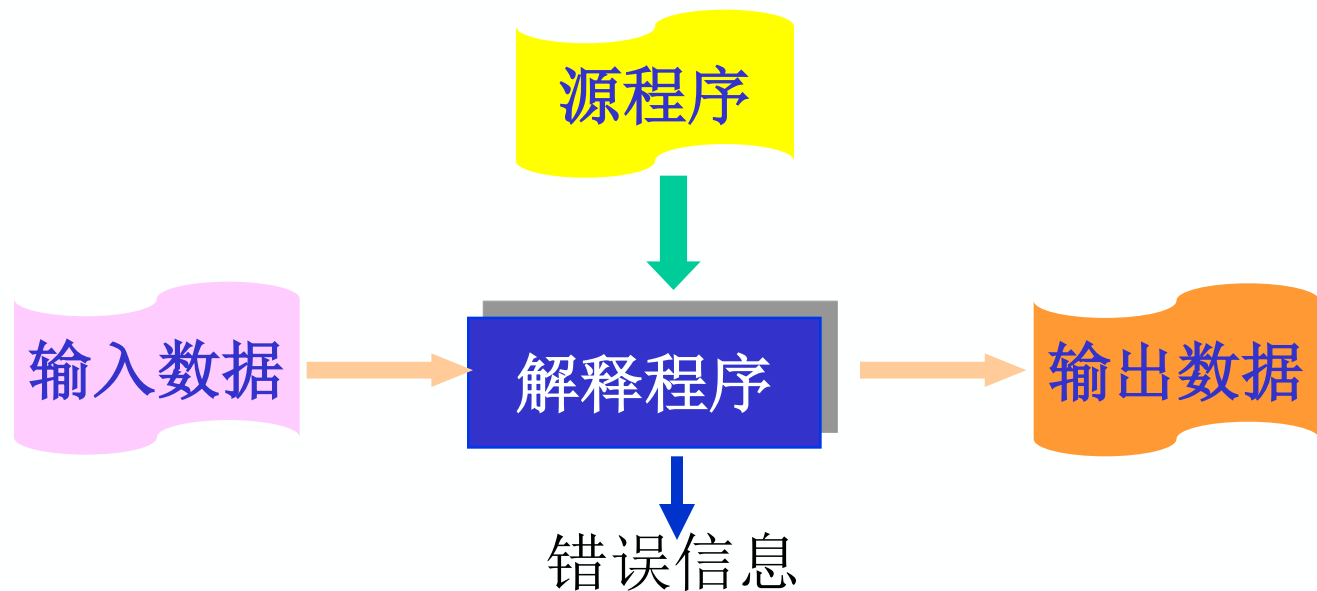


- 运行阶段



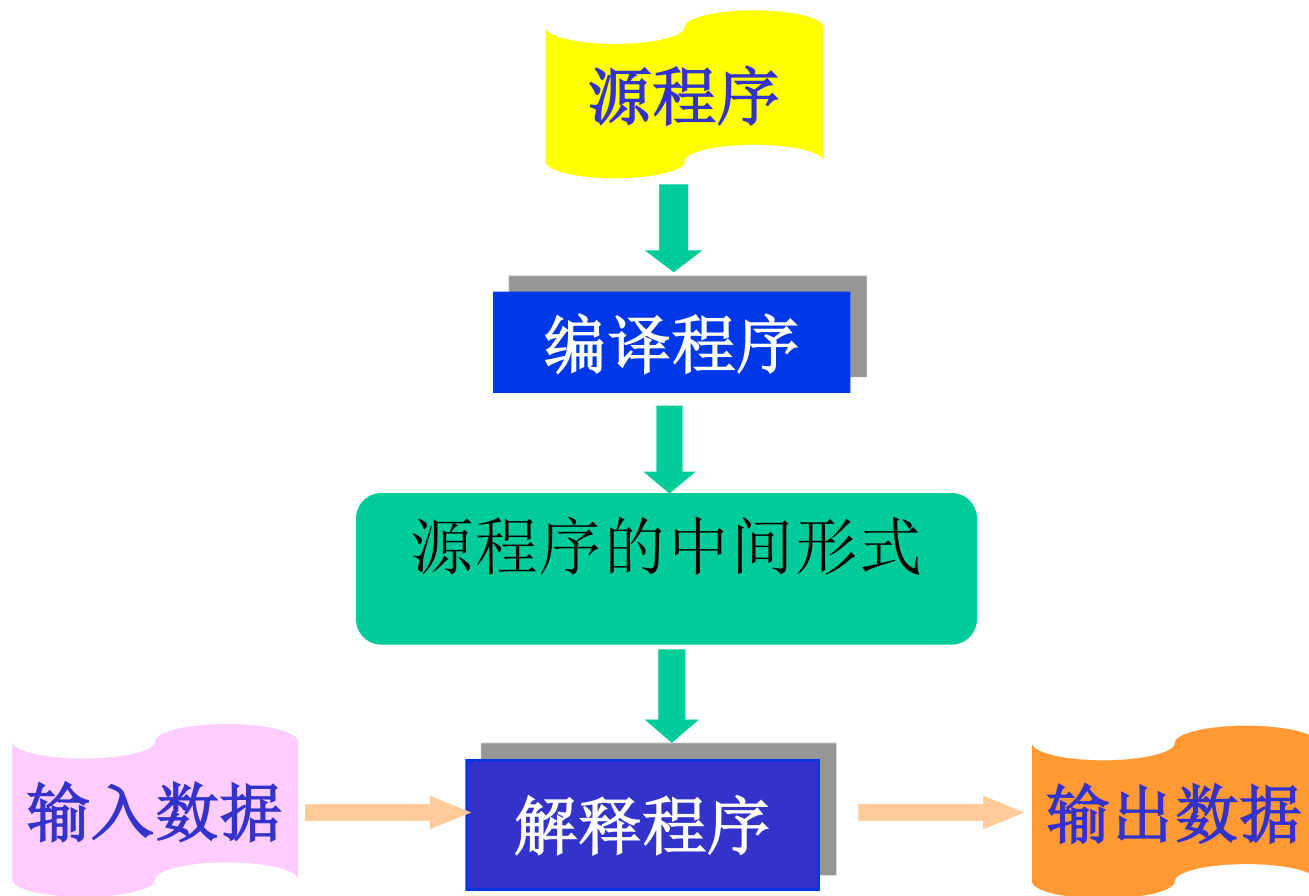
- **解释程序 (Interpreter)**
对源程序进行解释执行的程序。

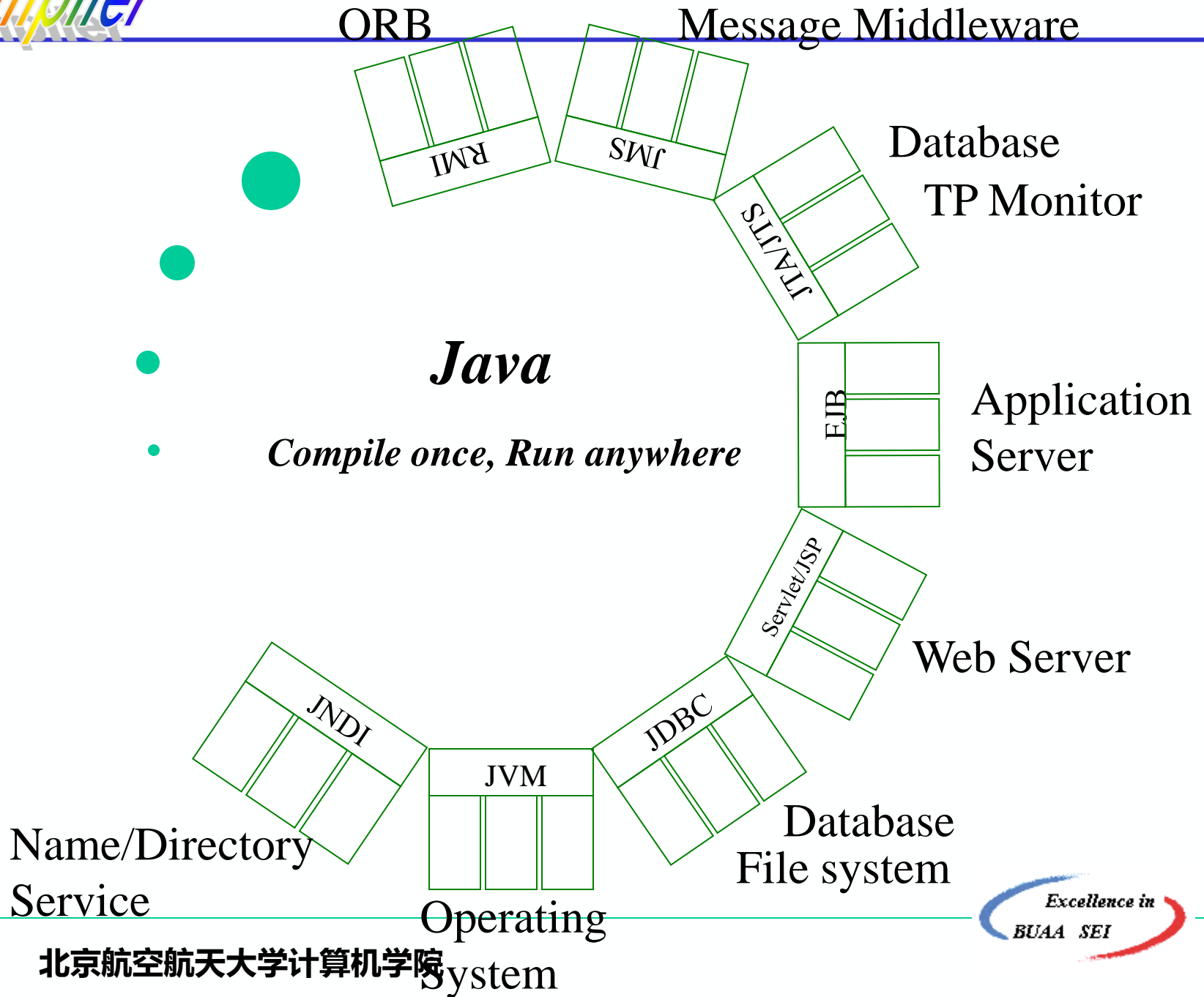
- **工作过程**



- **特点、与编译程序比较**

“编译-解释执行”系统





1.3.1 编译过程



编译过程是指将高级语言程序翻译为等价的目标程序的过程。

习惯上是将编译过程划分为5个基本阶段：



一、词法分析



任务：分析和识别单词。

源程序是由字符序列构成的，词法分析扫描源程序（字符串），根据语言的词法规则分析并识别单词，并以某种编码形式输出。

• **单词**：是语言的基本语法单位，一般语言有四大类单词

<1> **语言定义的关键字或保留字**（如BEGIN、END、IF）

<2> **标识符**

<3> **常数**

<4> **分界符**（运算符）（如+、-、*、/、；、（、）.....）

对于如下的字符串,词法分析程序将分析和识别出9个单词：

$$\frac{X1}{1} \frac{:=}{2} \frac{(\frac{2.0}{3} \frac{+}{4} \frac{0.8}{5} \frac{)}{6}}{7} \frac{*}{8} \frac{C1}{9}$$

二、语法分析

任务：根据**语法规则**（即语言的文法），分析并识别出各种语法成分，如表达式、各种说明、各种语句、过程、函数等，并进行语法正确性检查。

$X1 := (2.0 + 0.8) * C1$

赋值语句的文法：

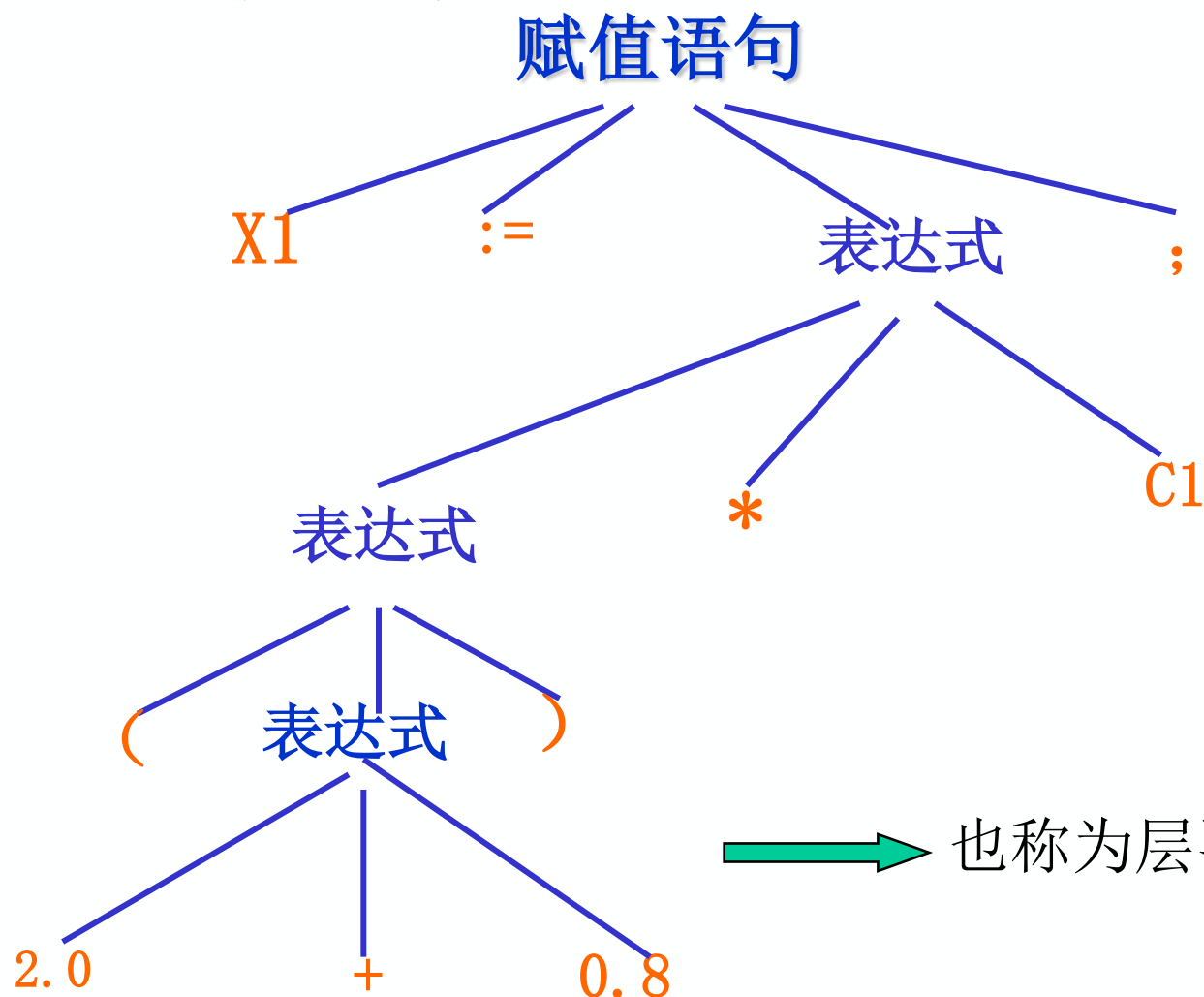
$\langle \text{赋值语句} \rangle \rightarrow \langle \text{变量} \rangle \langle \text{赋值操作符} \rangle \langle \text{表达式} \rangle$

$\langle \text{变量} \rangle \rightarrow \langle \text{简单标识符} \rangle$

$\langle \text{赋值操作符} \rangle \rightarrow :=$

$\langle \text{表达式} \rangle \rightarrow \dots\dots$

$X1 := (2.0 + 0.8) * C1;$



➡ 也称为层次分析。

三、语义分析、生成中间代码

任务：对识别出的各种语法成分进行语义分析，并产生相应的中间代码。

- 中间代码：一种介于源语言和目标语言之间的中间语言形式
- 生成中间代码的目的：
 - ＜1＞ 便于做优化处理；
 - ＜2＞ 便于编译程序的移植。
- 中间代码的形式：编译程序设计者可以自己设计，常用的有四元式、三元式、逆波兰表示等。

例： $X1 := (2.0 + 0.8) * C1$

- 由语法分析识别出为赋值语句，**语义分析**首先要分析语义上的正确性，例如要检查表达式中和赋值号两边的类型是否一致。
- 根据赋值语句的语义，**生成中间代码**。即用一种语言形式来代替另一种语言形式，这是翻译的关键步骤。（翻译的实质：**语义的等价性**）



★ 四元式（三地址指令）

$X1 := (2.0 + 0.8) * C1$

| | 运算符 | 左运算对象 | 右运算对象 | 结果 |
|-----|-----|-------|-------|----|
| (1) | + | 2.0 | 0.8 | T1 |
| (2) | * | T1 | C1 | T2 |
| (3) | := | X1 | T2 | |

其中T1和T2为编译程序引入的工作单元

四元式的语义为：

$$2.0 + 0.8 \rightarrow T1$$

$$T1 * C1 \rightarrow T2$$

$$T2 \rightarrow X1$$

这样所生成的四元式与原来的赋值语句在语言的形式上不同，但语义上等价。



任务：目的是为了得到高质量的目标程序。

例如：前面的四元式中第一个四元式是计算常量表达式值，该值在**编译时**就可以算出并存放在工作单元中，不必生成目标指令来计算，这样四元式可优化为：

编译时： $2.0 + 0.8 \rightarrow T1$

(1) * T1 C1 T2

(2) := X1 T2

优化前的四元式：

(1) + 2.0 0.8 T1

(2) * T1 C1 T2

(3) := X1 T2

五、生成目标程序

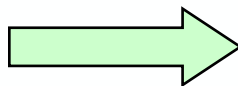


由中间代码很容易生成目标程序（地址指令序列）。这部分工作与机器关系密切，所以要根据机器进行。在做这部分工作时（要注意充分利用累加器），也可以进行优化处理。

$X1 := (2.0 + 0.8) * C1$

```
LOAD  2.0
ADD    0.8
STO    T1
LOAD   T1
MUL    C1
STO    T2
LOAD   T2
STO    X1
```

作利用累
加器的优化



```
LOAD  2.0
ADD    0.8
MUL    C1
STO    X1
```

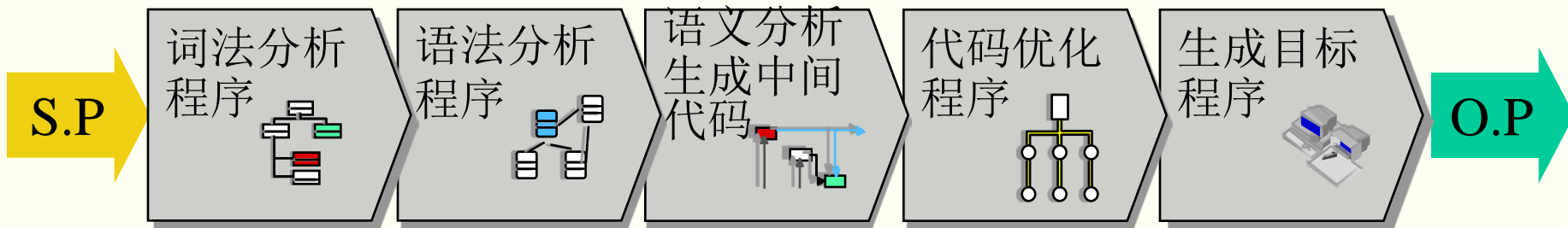
注意：在翻译成目标程序的过程中，
要切记保持语义的等价性。

1.3.2 编译程序构造



一、编译程序的逻辑结构

按逻辑功能不同，可将编译过程划分为五个基本阶段，与此相对应，我们将实现整个编译过程的编译程序划分为五个逻辑阶段（即五个逻辑子过程）。



在上列五个阶段中都要做两件事：

(1) 建表和查表； (2) 出错处理；

所以编译程序中都要包括符号表管理和出错处理两部分

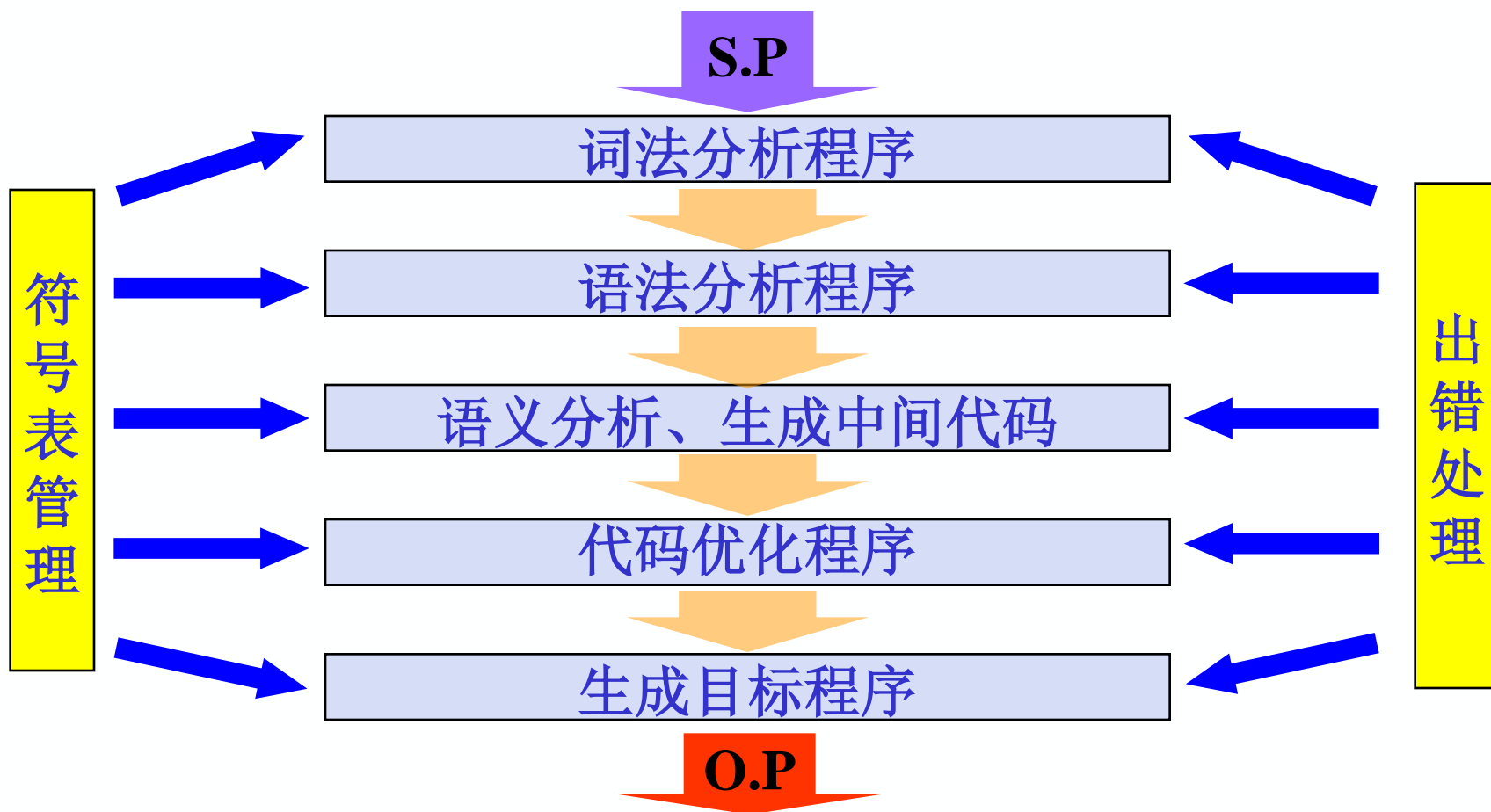
★ 符号表管理

在整个编译过程中始终都要贯穿着建表（填表）和查表的工作。即要及时地把源程序中的信息和编译过程中所产生的信息登记在表格中，而在随后的编译过程中同时又要不断地查找这些表格中的信息。

★ 出错处理

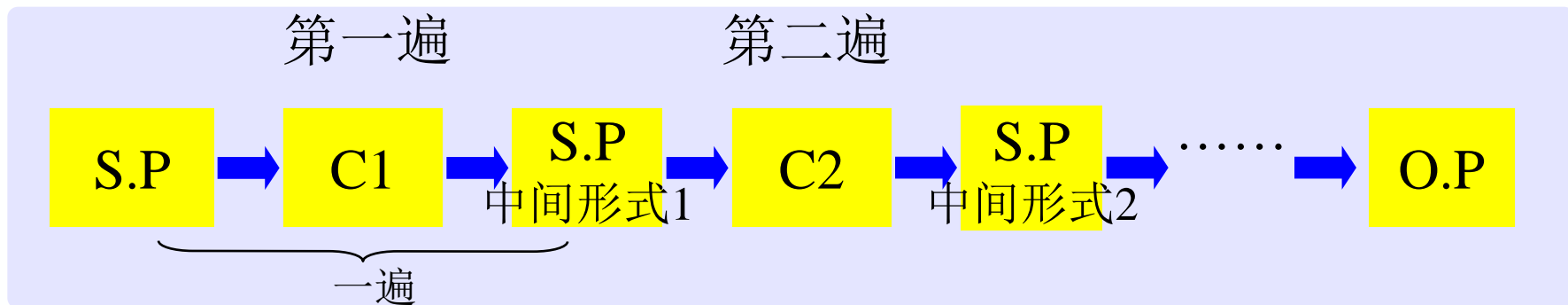
规模较大的源程序难免有多种错误，编译程序必须要有出错处理的功能。即能诊察出错误，并能报告用户错误的性质和位置，以使用户修改源程序。出错处理能力的大小是衡量编译程序质量好坏的一个重要指标。

典型的编译程序具有7个逻辑部分



二、遍 (PASS)

遍：对源程序（包括源程序中间形式）从头到尾扫描一次，并做有关的加工处理，生成新的源程序中间形式或目标程序，通常称之为**一遍**。

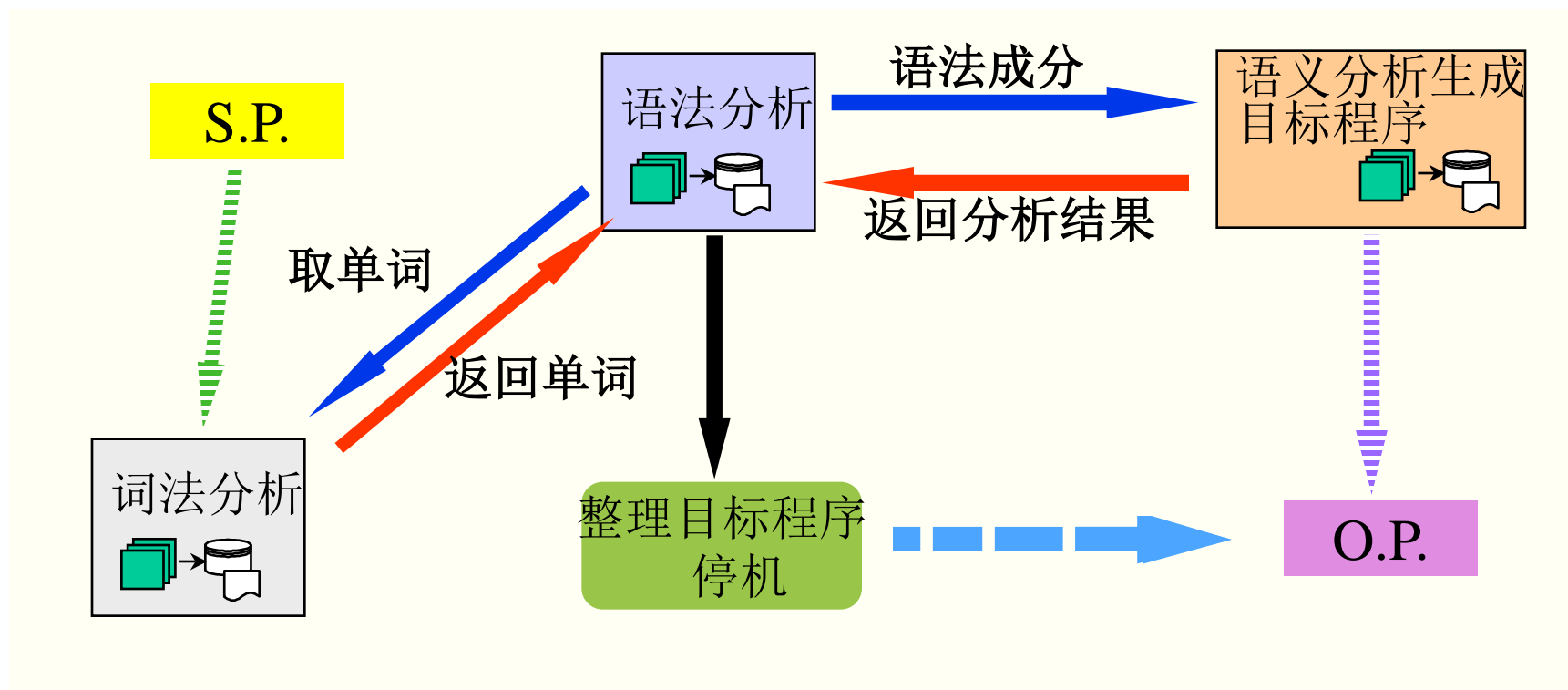


★ 要注意遍与基本阶段的区别

五个基本阶段：是将源程序翻译为目标程序在逻辑上要完成的工作。

遍：是指完成上述5个基本阶段的工作，要经过几次扫描处理。

一遍扫描即可完成整个编译工作的称为**一遍扫描编译程序**
其结构为：

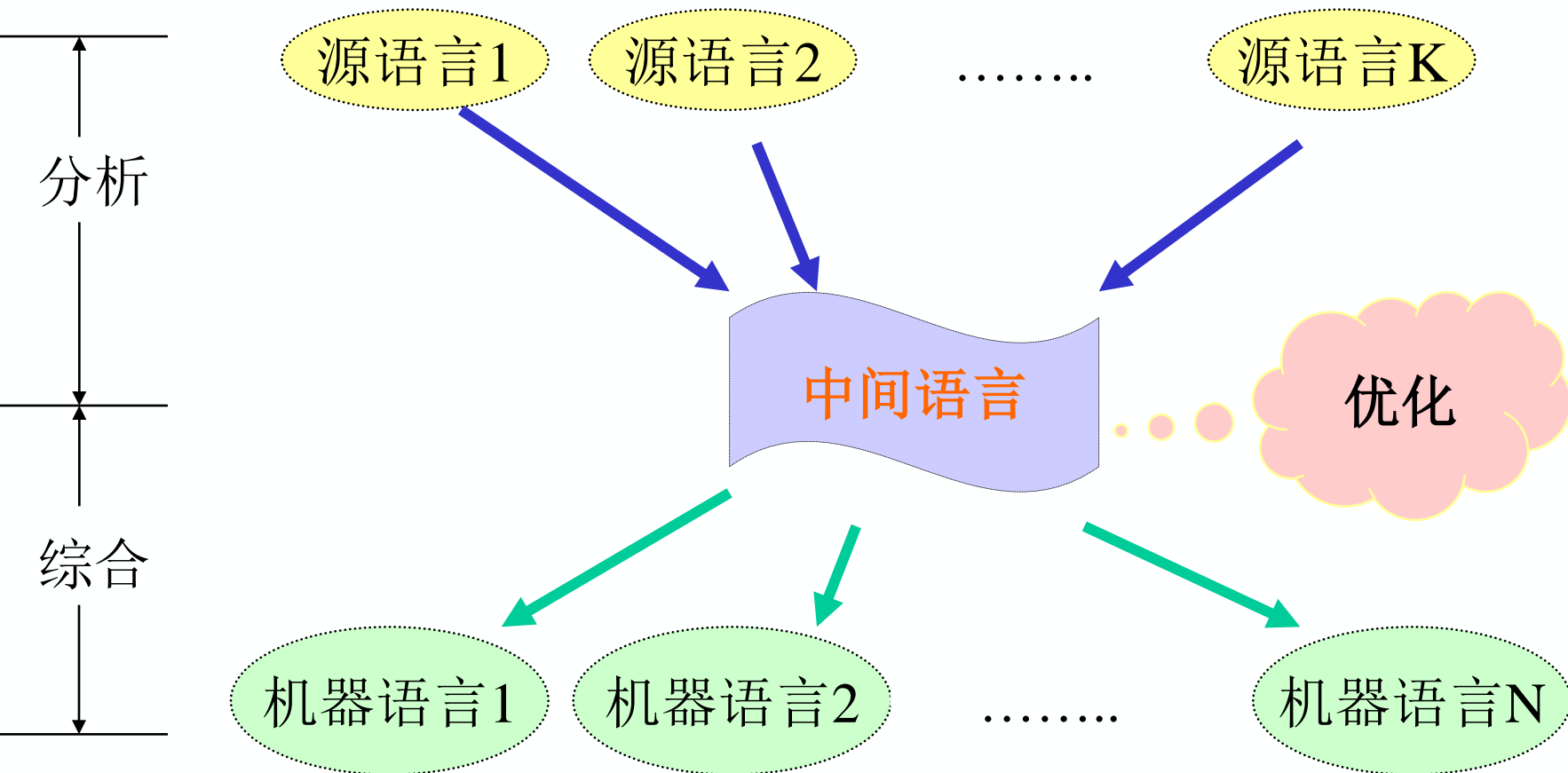


三、前端和后端

根据编译程序各部分功能，将编译程序分成前端和后端。

前端：通常将与源程序有关的编译部分称为前端。
词法分析、语法分析、语义分析、中间代码生成、
代码优化 ----- 分析部分
特点：与源语言有关

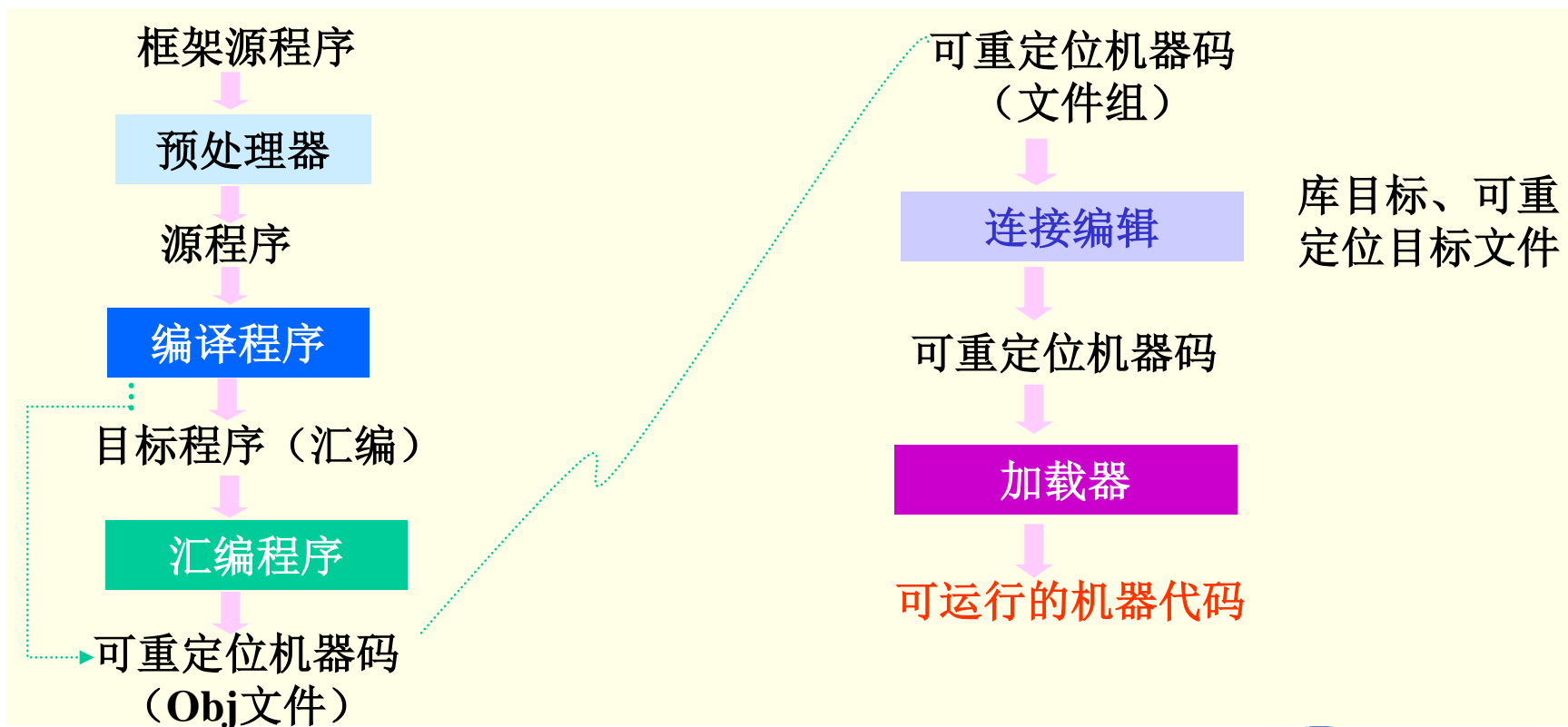
后端：与目标机有关的部分称为后端。
目标程序生成（与目标机有关的优化）
----- 综合部分
特点：与目标机有关



四、编译程序的前后处理器

源程序：多文件、宏定义和宏调用，包含文件

目标程序：一般为汇编程序或可重定位的机器代码



1.4 编译技术的应用

- ≈ 语法制导的结构化编辑器
- ≈ 程序格式化工具
- ≈ 软件测试工具
- ≈ 程序理解工具
- ≈ 高级语言的翻译工具
- ≈ 等等。

作业：P21：1,2,3,4,5

课程定位的理解

- **系统性**：编译是一个完整的系统，也是学生接触到的第一个系统
- **过程完整性**：编译不仅讲解了编译技术，其实质是讲解了模型从一种语言表达形式到另一种语言表达形式的等价转化方法，应该保证过程的完整性
- **实践性**：理论和实践相结合。无论是研究性大学、非研究性大学都应该注重实践。不同的在于具体要求的不同。

教学要求

- 理论学习

- 理解高级程序语言的工作原理
- 学习不同语言表示的程序之间的等价转化方法
- 编译程序的构造和工作原理
- 编译相关技术和常用优化技术

- 能力培养

- 理解系统的概念，完整设计一个不同难度的编译系统

回顾:

编译的起源

概念: 源程序 目标程序 翻译程序

汇编程序 编译程序

编译过程: 5个基本阶段

编译程序: 7个逻辑部分

遍 前端 后端 前后处理器