编译原理

邹阳

auroras@163.com

勤学楼4509

概 论

- 课程简介
- 什么是编译程序
- 为什么学习编译原理
- 编译过程概述
- 编译程序结构
- 编译程序生成

概 论

- 课程简介
- 什么是编译程序
- 为什么学习编译原理
- 编译过程概述
- 编译程序结构
- 编译程序生成

课程性质与任务

- 是计算机专业的一门**核心专业课**，旨在介绍编译程序构造的一般原理和基本方法
- 既是一门**理论性**、**技术性**与**实验性**很强的课程，又是理论与实践紧密结合的课程
- 本课程主要任务是介绍程序设计语言编译程序构造的基本原理和设计方法。通过本课程学习：
  - 掌握和理解编译一般过程、编译各个阶段功能、以及一些常用的编译设计方法和技巧
  - 在编译理论、方法和技术上得到系统且有效的训练
  - 掌握一些基础软件理论与软件分析方法，有助于提高软件人员的素质和能力

课程定位——计算机专业核心课程

分类	课程名称	课程定位	备注
计算机基础	计算机导论	入门	
	算法和数据结构	基础	
	高级语言程序设计（1，2）	必备工具	
计算机理论（离散数学1,2,3）	数理逻辑	计算机数学	
	集合论和图论		
	组合数学		
计算机硬件类课程	数子电路和数字逻辑	硬件基础课程	含实验
	计算机原理和汇编语言	部件原理	含实验
	计算机接口与通讯	部件间通讯	含实验
	计算机体系结构	体系结构	含实验
	计算机网络		
计算机软件类课程	编译技术	系统软件层	含课程设计
	操作系统		含课程设计
	数据库系统原理		含课程设计
	软件工程		
	人工智能	应用类	
	计算机图形学/数字图象处理	应用类	

课程主要内容

- 概述
- 形式语言与自动机
- 词法分析
- 语法分析
- 语法制导的语义计算
- 中间代码生成（符号表）
- 存储管理
- 代码优化
- 代码生成
- 编译程序构造方法

教 材

- 王生原等，《编译原理》 (第3版)，清华大学出版社，2015年6月
- 张素琴等，《编译原理》 (第2版)，清华大学出版社

参考书目

- **陈火旺，程序设计语言编译原理(第3版)，国防工业出版社，2000**
- **陈意云、张昱，编译原理(第3版)，高等教育出版社，2014**
- **Alfred V. Aho / Monica S.Lam / Ravi Sethi / Jeffrey D. Ullman. Compilers: Principles, Techniques, and Tools. Addison-Wesley, 2006**
- **Alfred/赵建华等译，编译原理(第2版)，机械工业出版社，2009**
- 钱焕延，编译技术(第2版)，东南大学出版社
- K. C. Louden，编译原理及实践，冯博琴译，机械工业出版社

相关课程

- 编译原理：3学分，理论、方法与技术
- 编译原理课程实践：1学分，实践，单独一门课

概 论

- 课程简介
- 什么是编译程序
- 为什么学习编译原理
- 编译过程概述
- 编译程序结构
- 编译程序生成

什么是编译程序

- 机器语言machine language
- 汇编语言assembly language
- 高级语言high-level language
- 翻译程序translator
  - 汇编程序assembler
  - 编译程序compiler
  - 解释程序interpreter

58 20 1020 (1000)	LD R2, x
59 20 1024 (1004)	CP R2, y
47 C0 1014 (1008)	BNG LESS
50 20 1028 (100C)	ST R2, max
47 F0 101C (1010)	JMP EXIT
58 20 1024 (1014)	LESS:
50 20 1028 (1018)	LD R2, y
	ST R2, max
	EXIT:

if (x>y) max=x; else max=y;

Definition

- Simply stated, a compiler is a program that reads a program written in one language-the source language-and translates it into an equivalent program in another language-the target language. (Excerpted from "*Compilers: principles, techniques and tools*", by *A.V. Aho, R. Sethi, and J.D. Ullman*)

定义

简单地，一个编译程序就是一个语言翻译程序，它把一种语言(称作源语言)书写的程序翻译成另一种语言(称作目标语言)书写的等价程序。

其他与编译相关的程序

- Linker
- Loader
- Preprocessor
- Editor
- Debugger
- Profiler：描述器，对执行中目标程序进行行为统计
- Project manager

概 论

- 课程简介
- 什么是编译程序
- 为什么学习编译原理
- 编译过程概述
- 编译程序结构
- 编译程序生成

为什么学习编译原理

- **计算机领域**
  - 程序设计语言
  - 软件技术
  - 思维方法

为什么学习编译原理

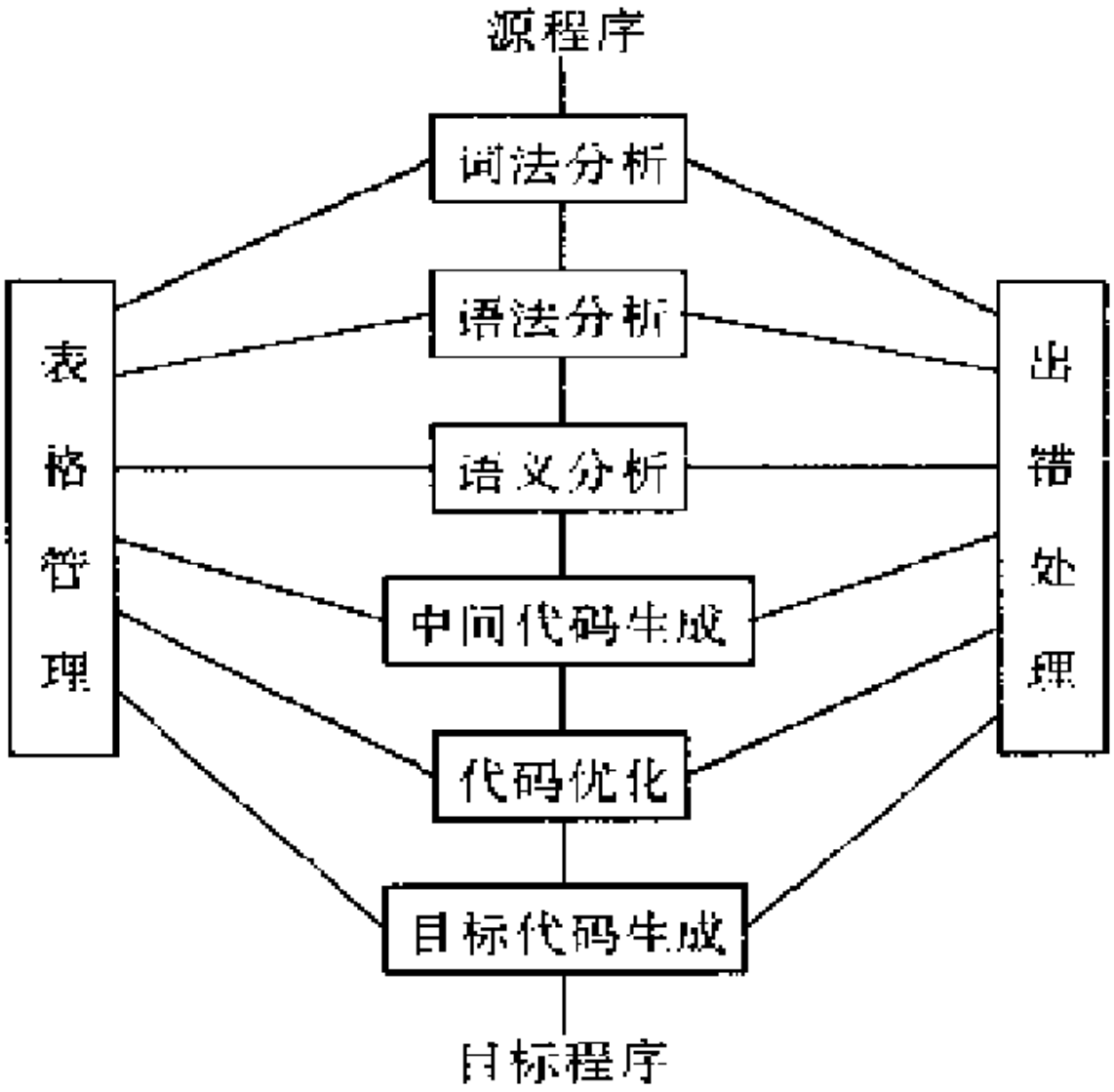
- 应用领域
  - 信息检索：文本描述
  - 人工智能：句法分析、自然语言理解
  - 网络语言：XML, SGML
  - 网络通信：通信协议描述与解释

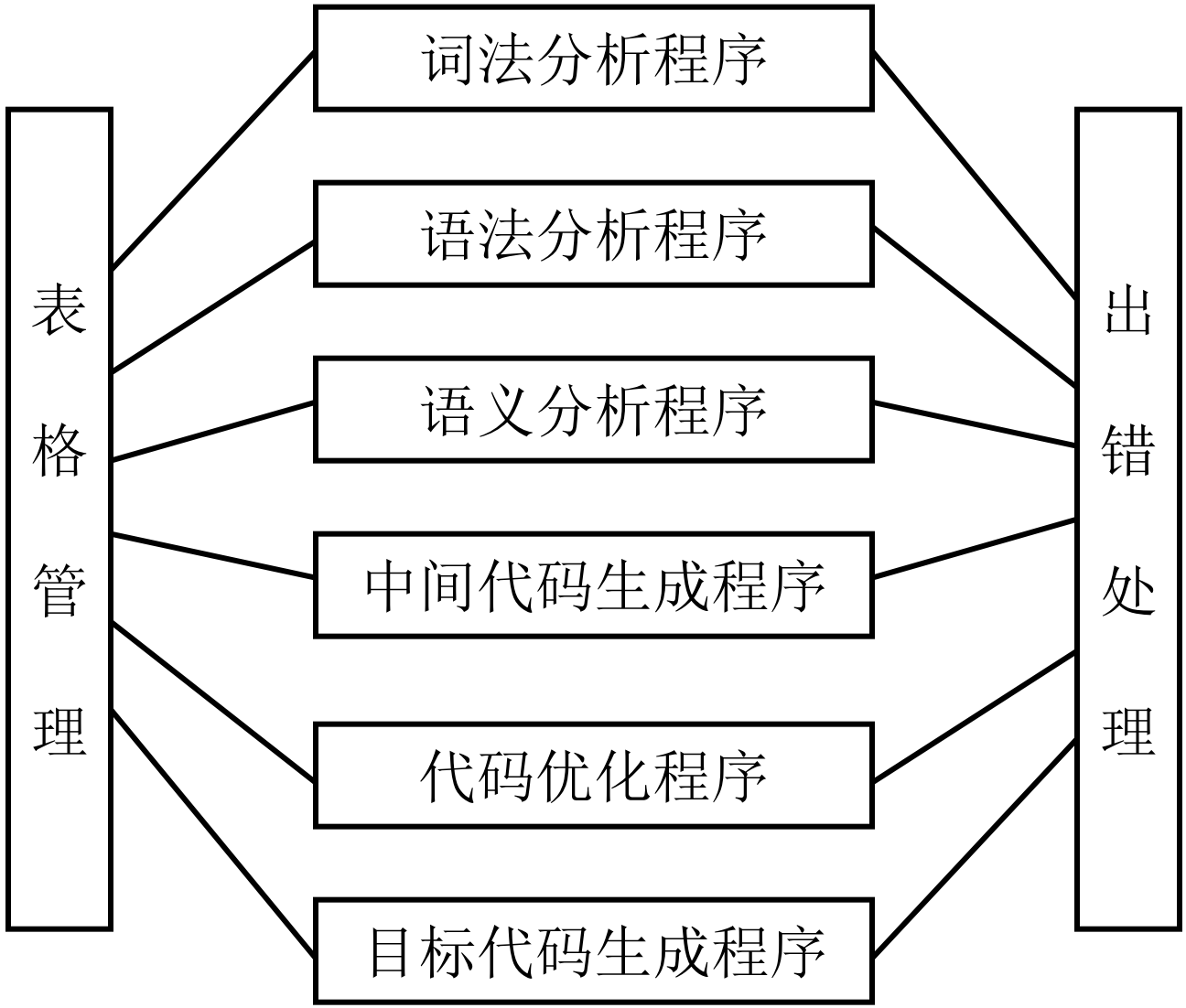
概 论

- 课程简介
- 什么是编译程序
- 为什么学习编译原理
- 编译过程概述
- 编译程序结构
- 编译程序生成

编译过程概述

- 词法分析 lexical analysis
- 语法分析 syntax analysis
  - 语法树 syntax tree
- 语义分析 semantic analysis
- 中间代码生成 intermediate code generation
- 代码优化 code optimization
- 目标代码生成 target code generation


源程序
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
目标程序
表格管理
出错处理


词法分析程序
语法分析程序
语义分析程序
中间代码生成程序
代码优化程序
目标代码生成程序
表格管理
出错处理

词法分析

从左至右读字符流的源程序、识别(拼)单词

例子：

position := initial + rate * 60;

position := initial + rate * 60;

单词类型	单词值
标识符1(id1)	position
算符(赋值)	:=
标识符2(id2)	initial
算符(加)	+
标识符3(id3)	rate
算符(乘)	*
整数	60
分号	;

又如, 一个C源程序片断： int a;

a = a + 2;

词法分析后可能返回:

单词类型	单词值
保留字	int
标识符(变量名)	a
界符	;
标识符(变量名)	a
算符(赋值)	=
标识符(变量名)	a
算符(加)	+
整数	2
界符	;

术 语

词法分析(lexical analysis or scanning)

---The stream of characters making up a source program is read from left to right and grouped into tokens, which are sequences of characters that have a collective meaning.

单　词---token

保留字---reserved word

标识符---identifier(user-defined name)

语法分析

功能：层次分析，**依据**源程序的**语法规则**把源程序的单词序列组成语法短语(表示成语法树)

position := initial + rate * 60 ;

规则：

<赋值语句>::=<标识符>“:=”<表达式>

<表达式>::=<表达式>“+”<表达式>

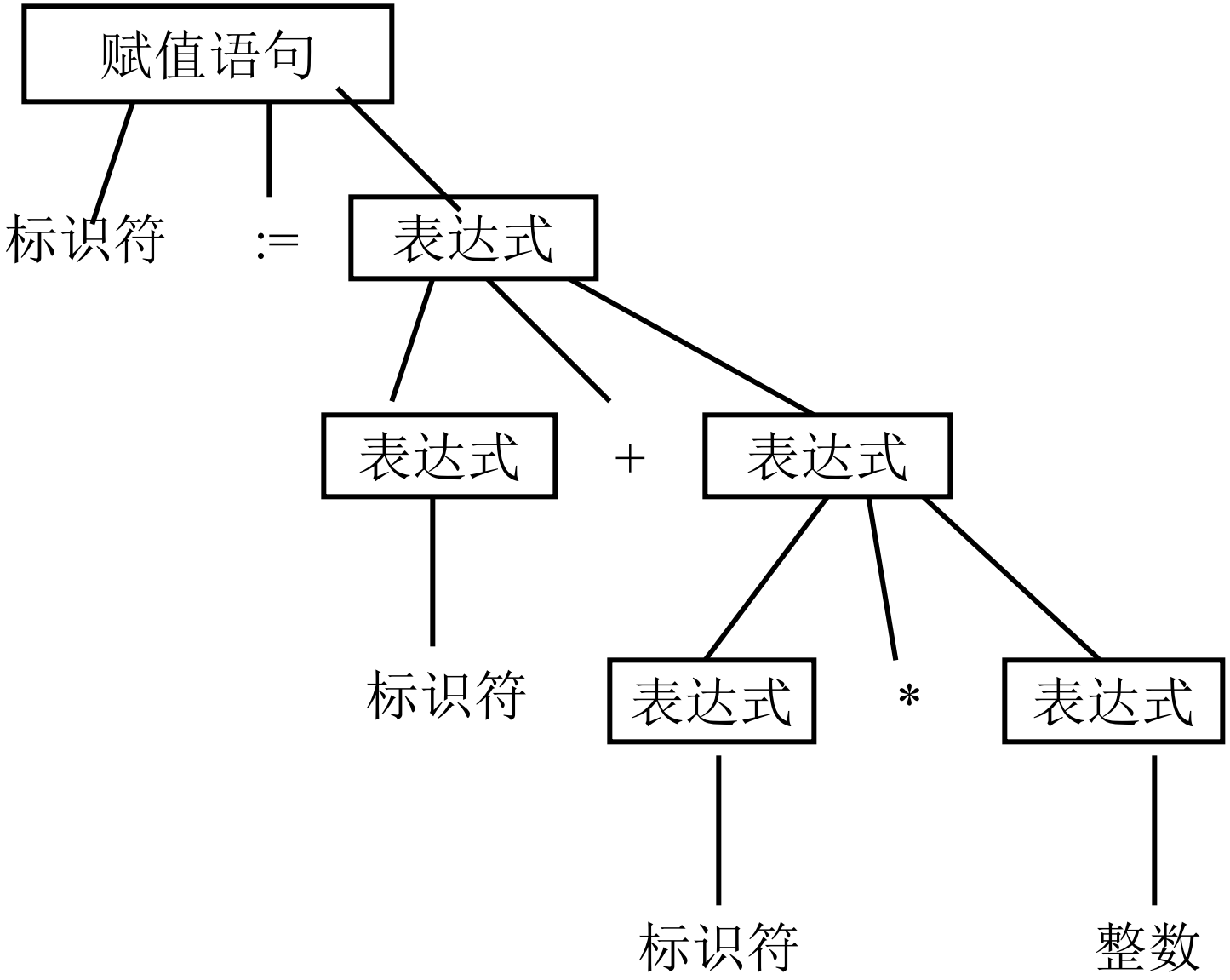
<表达式>::=<表达式>“*”<表达式>

<表达式>::=“(”<表达式>“)”

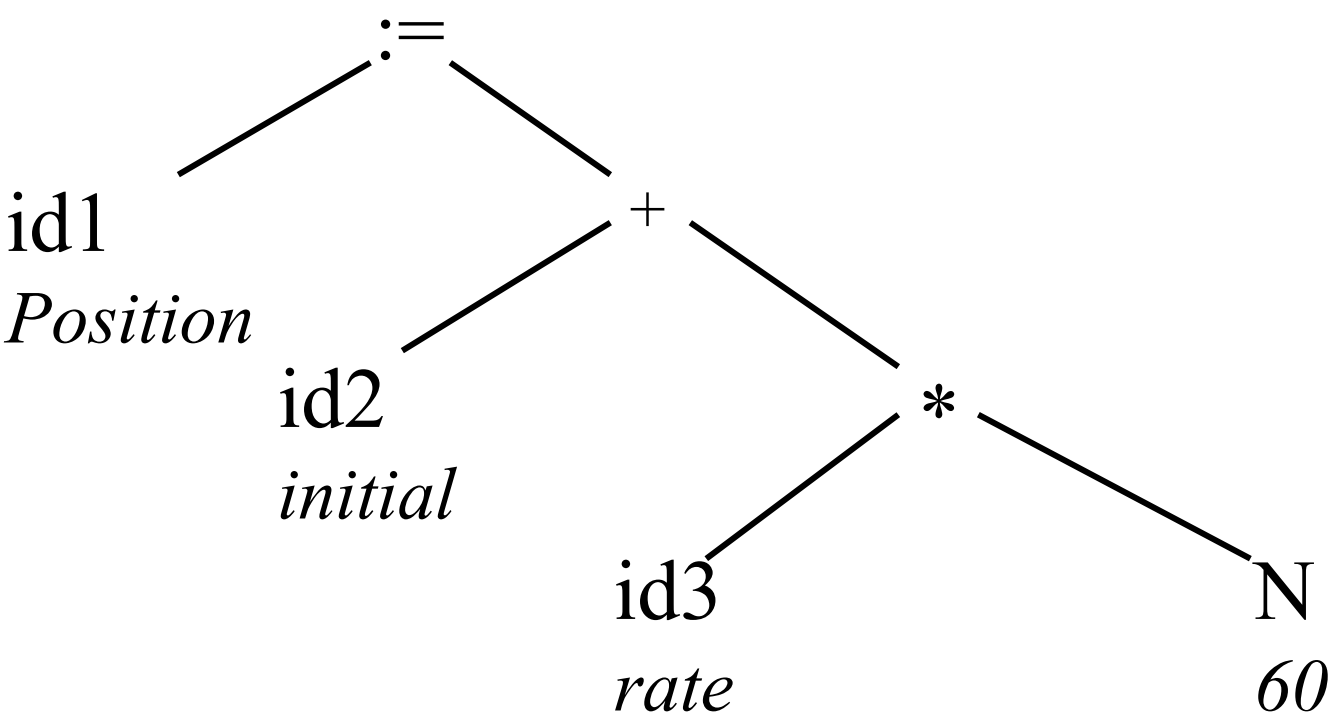
<表达式>::=<标识符>

<表达式>::=<整数>

<表达式>::=<实数>


赋值语句
标识符
:=
表达式
表达式
+
表达式
标识符
表达式
*
表达式
标识符
整数

id1:=id2+id3*N



术 语

语法分析(syntax analysis or parsing)

The purpose of syntax analysis is to determine the source program's phrase structure. This process is also called parsing. The source program is parsed to check whether it conforms to the source language's syntax, and to construct a suitable representation of its phrase structure.

语法树(推导树) (parse tree or derivation tree)

语义分析

语义审查(静态语义)

上下文相关性

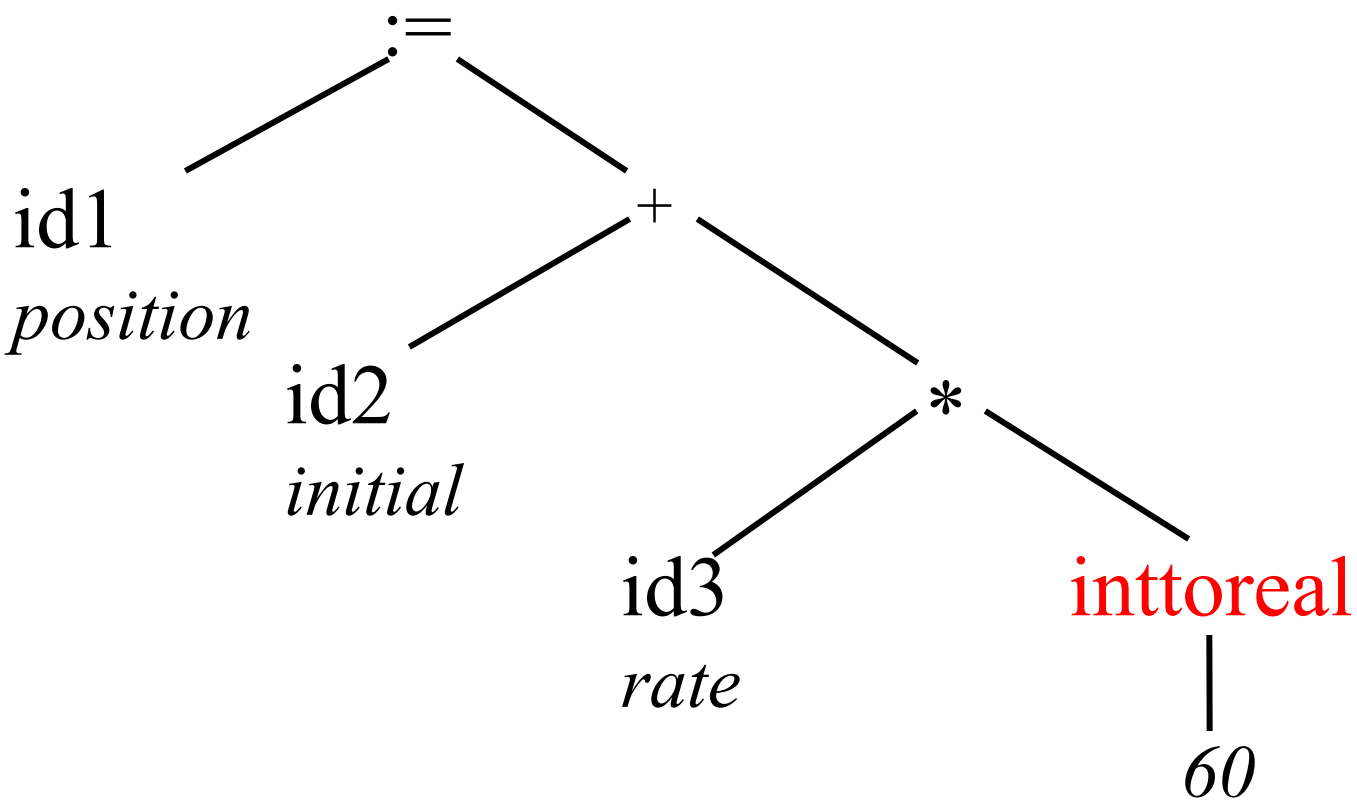
类型匹配

类型转换

例：

```
Program p();
var rate: real;
procedure initial;
…
position   :=   initial    +    rate * 60
/* error */  /* error */      /* warning */;
…
```

语义分析



术 语

语义分析(semantic analysis)

The parsed program is further analyzed to determine whether it conforms to the source language's contextual constraints: scope rules, type rules, e.g., to relate each applied occurrence of an identifier in the source program to the corresponding declaration.

中间代码生成

源程序的内部(中间)表示

三元式、四元式

(*　　id3　　t1　　t2)

t2 = id3 * t1

id1:= id2 + id3 * 60

(1)	(inttoreal,	60	-	t1	)
(2)	(* ,	id3	t1	t2	)
(3)	(+ ,	id2	t2	t3	)
(4)	(:= ,	t3	-	id1	)

中间代码生成
(intermediate code generation)

This is where the intermediate representation of the source program is created. We want this representation to be ***easy to generate***, and ***easy to translate into the target program***. The representation can have a variety of forms, but a common one is called three-address code or 4-tuple code.

代码优化

id1:= id2 + id3 * 60

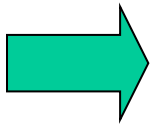
(1)	(inttoreal	60	-	t1)
(2)	(*	id3	t1	t2)
(3)	(+	id2	t2	t3)
(4)	(:=	t3	-	id1)

变换 ⇒

(1)	(*	id3	60.0	t1)
(2)	(+	id2	t1	id1)

代码优化

t1 = b* c
t2 = t1+ 0
t3 = b* c
t4 = t2 + t3
a = t4



t1 = b* c
t2 = t1 + t1
a = t2

代码优化(code optimization)

- **Intermediate code optimization**

  The optimizer accepts input in the intermediate representation and output a version still in the intermediate representation. In this phase, the compiler attempts to produce the smallest, fastest and most efficient running result by applying various techniques.

- **Object code optimization**

目标代码生成

(* , id3 60.0 t1)

(+ , id2 t1 id1)


```
movf   id3,R2
mulf   #60.0,R2
movf   id2,R1
addf   R2,R1
movf   R1,id1
```

符号表管理

记录源程序中使用的名字，收集每个名字的各种属性信息，如：类型、作用域、分配存储信息

Const1　　常量　　值：35

Var1　　变量　　类型：实　　层次：2

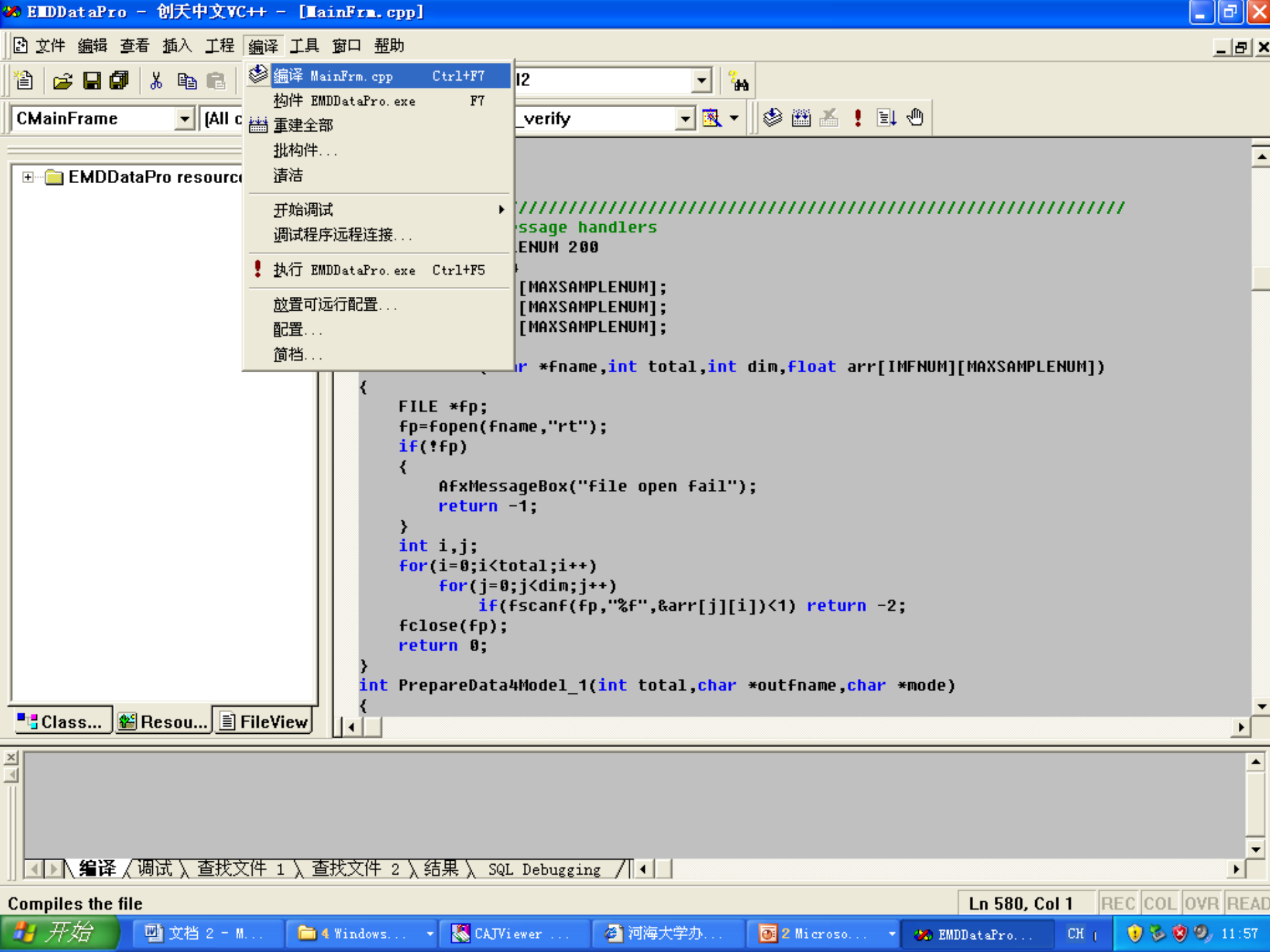
符号表(symbol table)

Symbol table is a data structure which is employed to associate identifiers with their attributes.

An identifier's attribute consists of information relevant to contextual analysis, and is obtained from the identifier's declaration.

出错处理

检查错误、报告出错信息、排错、恢复编译工作


EMDDataPro - 创天中文VC++ - [MainFrm.cpp]
文件 编辑 查看 插入 工程 编译 工具 窗口 帮助
编译 MainFrm.cpp Ctrl+F7
构件 EMDDataPro.exe F7
重建全部
批构件...
清洁
开始调试
调试程序远程连接...
执行 EMDDataPro.exe Ctrl+F5
放置可远行配置...
配置...
简档...
CMainFrame
EMDDataPro resourc
Class...
Resou...
FileView
ssage handlers
ENUM 200
[MAXSAMPLENUM];
[MAXSAMPLENUM];
[MAXSAMPLENUM];
*fname,int total,int dim,float arr[IMFNUM][MAXSAMPLENUM])
{
FILE *fp;
fp=fopen(fname,"rt");
if(!fp)
{
AfxMessageBox("file open fail");
return -1;
}
int i,j;
for(i=0;i<total;i++)
for(j=0;j<dim;j++)
if(fscanf(fp,"%f",&arr[j][i])<1) return -2;
fclose(fp);
return 0;
}
int PrepareData4Model_1(int total,char *outfname,char *mode)
{
编译 调试 查找文件 1 查找文件 2 结果 SQL Debugging
Compiles the file
Ln 580, Col 1
REC COL OVR READ
开始
文档 2 - M...
4 Windows...
CAJViewer ...
河海大学办...
2 Microso...
EMDDataPro...
CH
11:57

EMDDataPro - 创天中文VC++ - [MainFrm.cpp]

文件 编辑 查看 插入 工程 编译 工具 窗口 帮助

model2

(Globals) (All global members loaddata

EMDDataPro resources

Class... Resou... FileView

```
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
// CMainFrame message handlers
#define MAXSAMPLENUM 200
#define IMFNUM 4
float nj[IMFNUM][MAXSAMPLENUM];
float dt[IMFNUM][MAXSAMPLENUM];
float zj[IMFNUM][MAXSAMPLENUM];

int loaddata(char *fname,int total,int dim,float arr[IMFNUM][MAXSAMPLENUM])
{
    FILE *fp;
    fp=fopen(fname,"rt);
    if(!fp)
    {
        AfxMessageBox("file open fail");
        return -1;
    }
    int i,j;
    for(i=0;i<total;i++)
        for(j=0;j<dim;j++)
            if(fscanf(fp,"%f",&arr[j][i])<1) return -2;
    fclose(fp);
    return 0;
}
int PrepareData4Model_1(int total,char *outfname,char *mode)
{
```

MainFrm.cpp

```
D:\emd\EMDDataPro\MainFrm.cpp(121) : error C2001: newline in constant
D:\emd\EMDDataPro\MainFrm.cpp(122) : error C2143: syntax error : missing ')' before 'if'
D:\emd\EMDDataPro\MainFrm.cpp(123) : error C2143: syntax error : missing ';' before '{'
D:\emd\EMDDataPro\MainFrm.cpp(545) : warning C4305: 'initializing' : truncation from 'const double' to 'float'
```

编译 调试 查找文件 1 查找文件 2 结果 SQL Debugging

newline in constant

Ln 11, Col 1 REC COL OVR READ

开始 文档 2 - M... 4 Windows... CAJViewer ... 河海大学办... 2 Microso... EMDDataPro... CH 11:59

出错处理(error handling)
(error reporting and error recovery)

The compiler should report the location of each error, together with some explanation. The major categories of compile-time error: syntax error, scope error, type error.

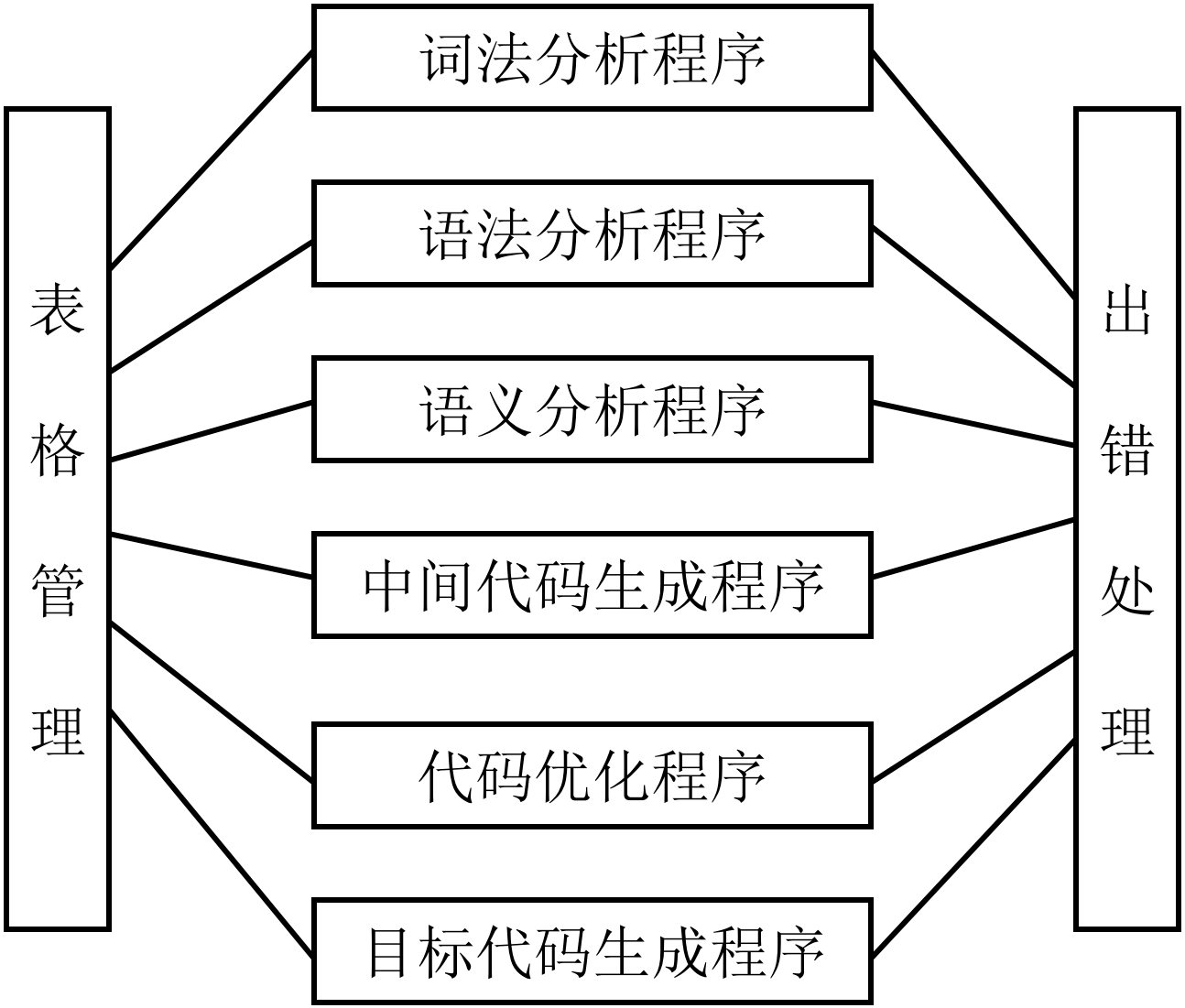
After detecting and reporting an error, the compiler should attempt error recovery, means that the compiler should try to get itself into a state where analysis of the source program can continue as normally as possible.

编译程序中的主要数据结构

- 记号表 token(单词的内部表示)
- 语法树 syntax tree
- 符号表 symbol table
- 常数表 literal table
- 中间代码 intermediate code
- 临时文件 temporary file

概 论

- 课程简介
- 什么是编译程序
- 为什么学习编译原理
- 编译过程概述
- 编译程序结构
- 编译程序生成


词法分析程序
语法分析程序
语义分析程序
中间代码生成程序
代码优化程序
目标代码生成程序
表格管理
出错处理

几个重要概念

- 趟 (pass, 遍)
- 单趟扫描
- 多趟扫描
- 编译的前端 (front end)
- 编译的后端 (back end)

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**

前端与后端

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

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（以及与目标机有关的优化）

特点：与目标机有关

概 论

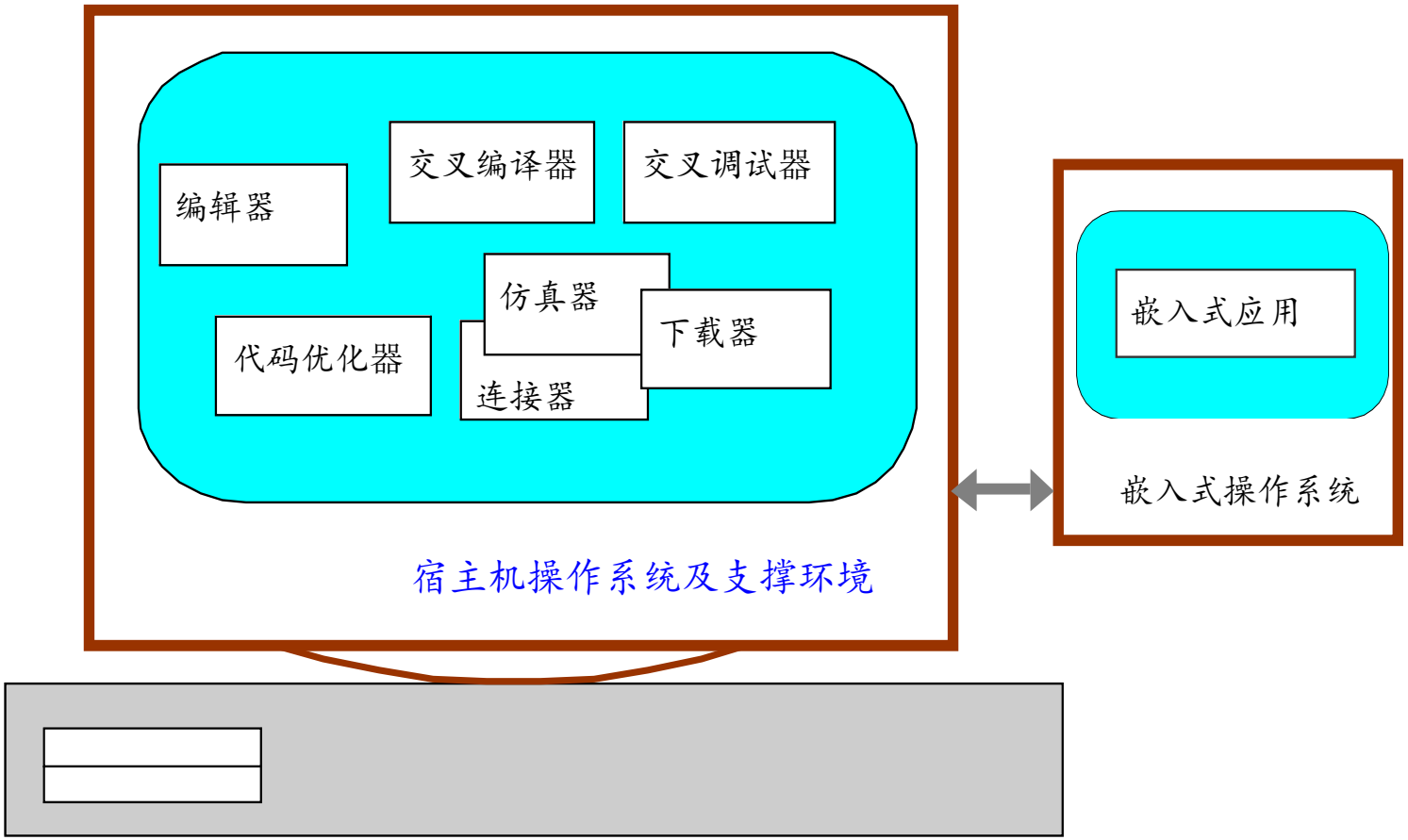
- 课程简介
- 什么是编译程序
- 为什么学习编译原理
- 编译过程概述
- 编译程序结构
- 编译程序生成

编译程序生成

- **交叉编译**
- 自展(自编译)
- 移植
- 自动生成程序

交叉编译器

由于目标机指令系统与宿主机的指令系统不同，编译时将应用程序的源程序在宿主机上生成目标机代码，称为交叉编译


编辑器
交叉编译器
交叉调试器
仿真器
下载器
代码优化器
连接器
宿主机操作系统及支撑环境
嵌入式应用
嵌入式操作系统

小 结

- 本章重点对编译的概念、编译程序的结构以及各个阶段功能作了概述
- 要求理解编译程序各个部分之间的逻辑关系以及它们怎样作为一个整体完成编译任务，还有若干重要概念：编译、遍、交叉编译等

作业

- 通读第1章
- 查阅编译技术应用方面的文献