21世纪计算机辅导系列丛书

编译原理课程辅导与习题解析

胡元义　李长河　吕林涛　谈姝辰　编著

人民邮电出版社
POSTS & TELECOMMUNICATIONS PRESS

21世纪计算机辅导系列丛书

编译原理课程辅导与习题解析

● 胡元义　李长河　吕林涛　谈姝辰　编著

人民邮电出版社

图书在版编目（CIP）数据

编译原理课程辅导与习题解析/胡元义等编著. —北京：人民邮电出版社，2002.7
ISBN 7-115-10196-5
Ⅰ. 编… Ⅱ. 胡… Ⅲ. 编译程序—程序设计—高等学校—教学参考资料 Ⅳ. TP314
中国版本图书馆 CIP 数据核字（2002）第 042658 号

内容提要

编译原理课程具有较强的理论性，学习起来难度较大。本书配合教学内容，从学生“学”的角度提供了全面的辅导。全书共分 8 章，基本覆盖了编译原理课程的全部内容，每章包括“重点内容讲解”、“典型例题解析”、“习题及答案”三大部分，带领读者经历从“学习理论”到“结合实际理解理论”再到“自己亲自动手解决问题”的学习过程，意在帮助读者深刻理解本课程涉及的原理和概念，掌握基本的编译方法，从而透彻地领悟编译原理的精髓。

书中精选的例题与习题大多选自本科生和研究生的考试试题，也包括作者结合多年教学实践经验设计出来的典型范例，具有一定的知识水平和代表性。本书对例题进行了深入、细致的分析和解答，力求帮助读者抓住重点、突破难点。另外，每章后给出的习题和参考答案可供读者检验对本章知识的掌握程度，进一步巩固所学知识。

本书可作为计算机专业学生的学习辅导书，也可作为研究生入学考试的复习参考书，还可供计算机软件开发人员参考阅读。

21 世纪计算机辅导系列丛书

编译原理课程辅导与习题解析

◆ 编　　著　胡元义　李长河　吕林涛　谈姝辰
责任编辑　王文娟
执行编辑　苗　颖

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
邮编　100061　电子函件　315@ptpress.com.cn
网址　http://www. ptpress.com.cn
读者热线　010-67180876
北京汉魂图文设计有限公司制作
北京朝阳展望印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：787×1092　1/16
印张：21.5
字数：521 千字　　2002 年 7 月第 1 版
印数：1-5 000 册　　2002 年 7 月北京第 1 次印刷

ISBN 7-115-10196-5/TP・2828

定价：29.80 元

前　言

计算机语言之所以能由单一的机器语言发展到现今的数千种高级语言，就是因为有了编译技术。编译技术是计算机科学中发展得最迅速、最成熟的一个分支，它集中体现了计算机发展的成果与精华。

编译原理是计算机专业的一门核心课程，在计算机本科教学中占有十分重要的地位。由于编译原理课程具有很强的理论性与实践性，学生在学习时普遍感到内容抽象、不易理解，掌握起来难度较大。本书通过课程辅导与习题解析的方式来帮助读者理解编译技术的原理和概念，掌握编译原理的相关方法，提高分析与解决问题的能力。

本书共分 8 章。第 1 章简要介绍了高级语言的特点与编译的基本概念；第 2 章介绍了词法分析的内容，主要涉及正规式与有限自动机；第 3 章主要介绍语法分析的相关知识，概括介绍了文法的定义及上下文无关文法，并介绍了两种主要的语法分析方法——算符优先分析法和 LL(1)分析法；第 4 章重点介绍了有关 LR 分析器的知识及各类 LR 分析表的构造方法；第 5 章介绍了语法制导翻译与中间代码生成的有关内容，给出了如何在语法分析的同时加工产生出中间代码的方法；第 6 章重点介绍程序运行时存储空间组织的相关内容，从编译角度考虑如何为程序的运行管理分配好存储空间；第 7 章介绍代码优化与目标代码生成的内容，讲解了如何对中间代码优化以及如何产生出最终的目标代码；第 8 章“符号表与错误处理”简要地介绍了符号表的组织与错误处理的方法。

各章首先介绍了本章包含的内容和需要重点掌握的知识点，然后对本章的典型例题进行了深入、细致的分析和解答。为了帮助读者抓住编译原理这门课程的精髓，提高解题技能，我们将每章的典型例题分为三类：第一类是概念题，题目都是关于本章知识包含的概念；第二类为基本题，这些题目体现了本章的基本内容和重点内容；第三类为综合题，主要是为开拓读者视野，题目多为综合题和典型应用题。

为了便于读者正确理解概念，掌握解题技巧，各章的典型例题大多给出了详尽的解题过程，以及引用到的概念、原理和公式的出处。对有代表性的习题和疑难习题，也给出了详细的分析和说明。此外，针对某些难题，书中还给出了一些新的解题思路和方法。

最后，每章还给出了供读者演练解题能力的习题，并附有习题答案。

希望读者通过对本书的学习，能更全面、透彻地理解和掌握编译原理这门课程。

由于编者水平有限，书中难免存在差错，敬请广大读者批评指正。本书责任编辑的电子邮件地址为 wangwenjuan@ptpress.com.cn，衷心希望与各位读者进行交流。

编者

2002 年 4 月

目　录

第1章 高级语言与编译程序概述

1.1 重点内容讲解

1.1.1 高级程序语言概述

高级程序语言是用来达到交流算法并在计算机中实现这两种目的的。高级语言一般较接近数学语言和工程语言，因此比较直观、自然和易于理解。并且，高级语言通常都是独立于计算机的。

1．程序语言的定义

（1）程序语言：一个程序语言是一个记号系统。如同自然语言一样，程序语言也是由语法和语义两方面定义的。任何语言程序都可看成是一定字符集（称为字母表）上的一个字符串，合乎语法的字符串才算是一个合式的程序。所谓一个语言的语法是指这样一组规则，用它可以形成和产生一个合式的程序，这些规则一部分称为词法规则，另一部分称为语法规则（或产生规则）。

（2）词法规则：指单词符号的形成规则。

（3）语法规则：语言的语法规则规定了如何从单词符号形成更大的结构（即语法单位）。换言之，语法规则是语法单位的形成规则，一般程序语言的语法单位有表达式、语句、分程序、函数、过程和程序等。

语言的词法规则和语法规则定义了程序的形式结构，是判断输入的字符串是否构成一个形式上正确（即合式）的程序的依据。

（4）语义规则：对于一个语言，不仅要给出它的词法、语法规则，而且要定义它的单词符号和语法单位的意义，这就是语义问题。离开了语义，语言只不过是一堆符号的集合。所谓一个语言的语义是指这样一组规则，使用它可以定义一个程序的意义。

一个程序语言的基本功能是描述数据和进行数据运算。所谓程序，从本质上来说是描述一定数据的处理过程。一个程序大体上可视为图 1.1 所示的层次结构。

2．类型、数组、名字、表达式与语句

（1）初等类型数据

一个程序语言必须提供一定的初等类型数据成分，并定义对于这些数据成分的运算。有

些语言（如 Pascal）还提供了由初等数据构造复杂数据的手段。常见的初等数据类型有：数值数据、逻辑数据、字符数据、指示器（它的值指向另一些数据）。

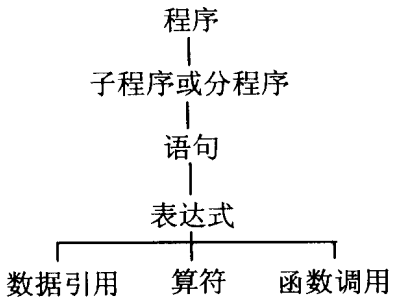


图 1.1　程序的层次结构

（2）数组

数组元素的地址计算和存储方式密切相关。对 n 维数组 $A[l_1:u_1,l_2:u_2,\cdots,l_n:u_n]$，其中，$l_i$、$u_i$ 分别代表数组 A 各维的下界和上界。设 a 为数组 A 的首地址，每个数组元素占 k 个机器字节（存储器按字节编址），则按行排列时元素 $A[i_1,i_2,\cdots,i_n]$的地址 D 计算如下：

$$D=a+[(i_1-l_1)d_2d_3\cdots d_n+(i_2-l_2)d_3d_4\cdots d_n+\cdots+(i_{n-1}-l_{n-1})d_n+(i_n-l_n)]\times k$$

（3）标识符和名字

在程序语言中各种名字都是用标识符表示的。所谓标识符是指由字母或数字组成的，但以字母开头的字母数字串。注意，标识符是一个没有意义的字符序列，而名字却有明确的意义和属性。每个名字可看成是代表一个抽象的存储单元，这个单元的内容认为是此名字（在某一时刻）的值。如果不指出名字的属性，它的值就无法理解。一个名字的属性包括类型和作用域。名字的类型决定了它能具有什么样的值、值在计算机内的表示方式以及对它能施加什么运算，名字的作用域则规定了它的值的存在范围。

（4）表达式

一个表达式是由运算量（亦称操作数，即数据引用或函数调用）和算符组成的。对多数程序语言来说，表达式的形成规则可概括为：

① 变量（包括下标变量），常数是表达式；

② 若 E_1、E_2 为表达式，θ 是一个二元算符，则 $E_1\ \theta\ E_2$ 是表达式；

③ 若 E 是表达式，θ 为一元算符，则 θE（或 Eθ）是表达式；

④ 若 E 是表达式，则（E）是表达式。

表达式中算符的运算顺序和结合性的约定大多和通常的数学习惯一致。

（5）语句

从功能上说，语句大体可分为说明语句和执行语句两大类。说明语句旨在定义各种不同数据类型的变量或运算；执行语句旨在描述程序的动作。

从编译角度说，说明语句旨在定义名字的性质，许多说明语句没有相应的目标代码，编译程序把这些性质登记在符号表中，并检查程序中名字的引用和说明是否相一致。

我们知道，每个名字有两方面的特征，一方面它代表一定的存储单元，另一方面它又以该单元的内容作为值。而对执行语句中的赋值语句 A:=B，其意义是“把变量 B 的值送入变量 A 所代表的存储单元”；即对赋值号右边的 B 我们需要的是它的值，对左边的 A 我们需要

的是它所代表的那个存储单元（地址）。为了区分一个名字的这两种特征，我们把一个名字所代表的那个单元（地址）称为该名的左值；把一个名字的值称为该名的右值；也就是说，名字的左值指它所代表的存贮单元的地址，右值指该单元的内容。变量（包括下标变量）既持有左值又持有右值。常数和带有算符的表达式一般认为只持有右值。但对指示器变量，如 P，它的右值 P↑既持有左值又持有右值。

执行语句又分为简单句和复合句。简单句是指那些不包含其他语句成分的基本语句，复合句则指那些句中有句的语句。

3．程序段

一个语言的最高级结构包括分程序、过程和程序。对于分程序结构的语言来说，其分程序的结构可以是嵌套的，也就是说，分程序内可以含有别的分程序。过程也可以看成是一个分程序，这个分程序可以在别的分程序中被调用。

分程序结构允许同一标识符在不同的分程序中表示不同的名字。关于名字的作用域的规定是：

① 一个在分程序 B_1 中说明的名字 X 只在 B_1 中有效（局部于 B_1）；

② 如果 B_2 是 B_1 的一个内层分程序且 B_2 中对标识符 X 没有新的说明，则原来的名字 X 在 B_2 中仍然有效；如果 B_2 对 X 重新作了说明，那么，B_2 中对 X 的任何引用都是指重新说明过的这个 X。

换言之，标识符 X 的任一出现（除出现在说明句的名表中外）都意味着引用某一说明语句所说明的那个 X，此说明语句同所出现的 X 共处在一个最小分程序中，这个原则称为“最近嵌套原则”。

分程序结构的主要优点是：可以非常有效地使用存储器。因为一个分程序只有在执行时才需要数据空间，执行后所占用的空间被释放。由于分程序结构的嵌套性，因此可用一个栈作为整个程序运行的数据空间。

4．参数传递

在 Pascal 中，下面的过程段：

```
FUNCTION  MOD（X，Y）：real；
BEGIN
        MOD：=sqrt(x*x+y*y)
END；
```

定义了一个称为 MOD 的函数过程。其中 X、Y 称为形式参数，简称形参。下面这个语句表示了对这个函数的一次调用：

```
A:=MOD（B,C）；
```

其中，B、C 称为实在参数，简称实参。

下面，分别讨论参数传递的 4 种不同的途径，分别是传地址（Call by reference）、传值（Call by value）、传结果（Call by result）和传名（Call by name）。“传名”常常也称“换名”，“传结果”也称“得结果”。

① 传地址：所谓传地址是指把实在参数的地址传递给相应的形式参数。在过程段中每个形式参数都有一个对应单元，称为形式单元。形式单元将用来存放相应的实在参数的地址。当调用一个过程时，调用段必须预先把实在参数的地址传递到一个被调用段可以拿得到的地

方。如果实在参数是一个变量（包括下标变量），则直接传递它的地址；如果实在参数是常数或其他表达式（如 A+B），那就先把它的值计算出来并存放在某一临时单元之中，然后传递这个临时单元的地址。当程序控制转入被调用段之后，被调用段首先把实参地址抄进自己相应的形式单元中，过程体对形式参数的任何引用或赋值都被处理成对形式单元的间接访问（即通过形式单元所存放的实参地址转而对实参进行操作）。这样，当被调用段工作完毕返回时，实在参数单元已经持有所期望的值。

② 传值：传值是一种最简单的参数传递方法。调用段把实在参数的值计算出来并存放在一个被调用段可以拿得到的地方。被调用段开始工作时，首先把这些值抄进自己的形式单元中；此后，就像使用局部名一样使用这些形式单元，即形式参数如同是一种先从实参那里获得初值的局部变量，获得初值后就再不与实参发生任何联系了，这点与传地址不同，在传地址中，形式参数始终都与实参联系着（形参的值始终指向实参，而操作都据此而转向实参，即针对实参进行）。在传值方式下，如果实参不为指示器（即指针变量），那么，在被调用段里将永远无法改变实参的值。

③ 传结果：和传地址相似（但不等价）的另一种参数传递方法是传结果。这种方法的实质是，每个形式参数对应有两个单元，第一个单元存放实参的地址，第二个单元存放实参的值。在过程体中对形参的任何引用或赋值都看成是对它的第二个单元的直接访问。但在过程工作完毕返回前必须把第二个单元的内容存放到第一个单元所指的那个实参单元中。

④ 传名：传名是 ALGOL 语言所定义的一种特殊的形实参数结合方式。ALGOL 用“替换规则”解释“传名”参数的意义。过程调用的作用相当于把被调用段的过程体抄到调用出现的地方，但把其中任何一个出现的形式参数都替换成相应的实在参数（文字替换）。如果在替换时发现过程体中的局部名和实在参数的名字使用相同的标识符，则必须用不同的标识符来表示这些局部名。

1.1.2 编译程序概论

计算机执行一个高级语言程序一般要分为两步：第一步，用一个编译程序把高级语言翻译成机器语言；第二步，运行所得的机器语言程序求得运行结果。

通常所说的翻译程序是指这样的一个程序，它能够把某一种语言程序（称为源语言程序）改造成另一种语言程序（称为目标语言程序），后者与前者在逻辑上是等价的。例如源语言可以是 Fortran、Pascal、Cobol 或 C 这样的“高级语言”，而目标语言是汇编语言这类的“低级语言”，这样的一个翻译程序就称为编译程序。

编译程序的工作是指从输入源程序开始到输出目标程序为止的整个过程，是非常复杂的。一般来说，整个过程可以划分成 5 个阶段：词法分析、语法分析、中间代码生成、优化和目标代码生成。

第一阶段，词法分析。词法分析的任务是输入源程序，对构成源程序的字符串进行扫描和分解，识别出一个个单词符号，如基本字、标识符、常数、算符和界符等。在词法分析阶段的工作中遵循的是语言的构词规则。

第二阶段，语法分析。语法分析的任务是在词法分析的基础上，根据语言的语法规则（文法规则）把单词符号串分解成各类语法单位（语法范畴），如“短语”、“子句”、“句子（语句）”、

“程序段”和“程序”。通过语法分解确定整个输入串是否构成一个语法上正确的“程序”。语法分析所遵循的是语言的语法规则。

第三阶段，中间代码让成。这一阶段的任务是对各类不同语法范畴按语言的语义进行初步翻译的工作。把语法范畴翻译成中间代码所遵循的是语言的语义规则。一般而言，中间代码是一种独立于具体硬件的记号系统，它与现代计算机的指令形式有某种程度的接近，或者说能够比较容易地把它变换成现代计算机的机器指令。常用的中间代码有四元式、三元式、间接三元式和逆波兰记号等。

第四阶段，优化。优化的任务是对前阶段产生的中间代码进行加工变换，以便在最后阶段能产生出更为高效（节省时间和空间）的目标代码。优化的主要方面有：公共子表达式的提取、循环优化、算符归约等。优化所遵循的原则是程序的等价变换规则。

第五阶段，目标代码生成。这一阶段的任务是把中间代码（或经优化处理之后）变换成特定机器上的绝对指令代码或可重新定位的指令代码或汇编指令代码。这个阶段实现了最后的翻译，它的工作依赖于硬件系统结构和机器指令含义。

如果最终得到的目标代码是绝对指令代码，则这种目标代码可以立即运行；如果目标代码是汇编指令代码，则这种目标代码需经汇编之后方可运行。目前多数实用编译程序所产生的目标代码都是一种可重定位的指令代码，这种目标代码必须通过一个连接装配程序把各个目标模块连接、装配在一起，使之成为一个可以独立运行的绝对指令代码程序。

上述编译过程的 5 个阶段是编译程序工作时的动态特征，编译程序的结构可以按照这 5 个阶段的任务分模块进行设计。编译程序的结构示意如图 1.2 所示。

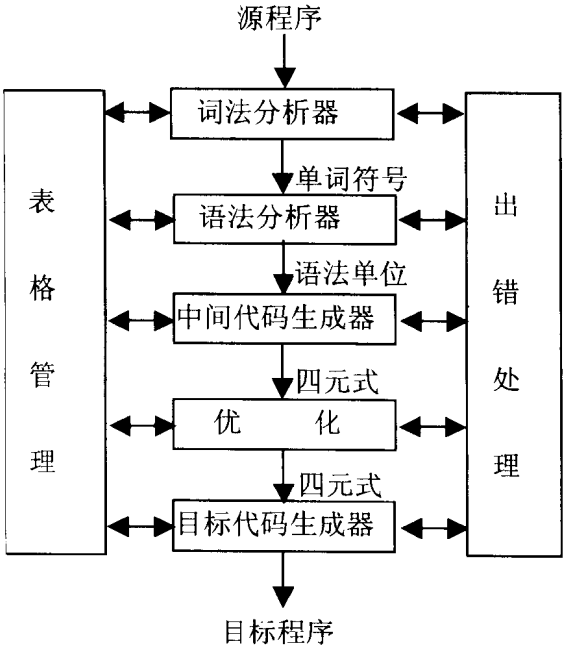


图 1.2　编译程序结构示意

编译过程中源程序的各种信息被保留在不同的表格里，编译各阶段的工作都涉及到构造、查找或更新有关的表格。因此，编译过程的绝大部分时间是用在造表、查表和更新表格的事

务上。出错处理与编译的各个阶段都有联系，与前三个阶段的联系尤为密切。

目前已经建立了各种编制部分的编译程序或整个编译程序的有效工具。有些能用于自动产生扫描器，有些可用于自动产生语法分析器，有些甚至可用来自动产生整个编译程序。此外，有些编译程序还可通过“移植”得到，即把某一计算机上的编译程序移植到另一计算机上，这需要寻找某种适当的“中间语言”。但是，由于无法实现建立通用中间语言，因此移植也只能在几种语言和几种类型的计算机之间进行。

1.1.3 过程与函数执行的分析方法

1．动态图描述规则

为了能够描述过程或函数调用执行的全过程，我们设计并采用动态图的方法来对程序的执行进行描述，即记录主程序和过程或函数的调用、运行及撤销各个阶段的状态，以及程序运行期间所有变量和过程、函数中值参（传值方式下的形式参数）与变参（传地址方式下的形式参数）的变化过程。动态图规则如下。

（1）动态图纵向描述主程序、过程或函数各层之间的调用关系；横向由左至右按执行的时间顺序记录主程序、过程或函数中各变量值的变化情况。

（2）过程或函数的值参均看作是带初值的局部变量（也可用箭头来表示实际参数传给值参的指向），其后，值参就作为局部变量参与过程或函数中的操作。对于变参，由于其作用就像指向实际参数的指针，故动态图中变参一律指向与其对应的实参变量（注意，两者位于动态图相邻的两层上；如果实参为表达式，则用一个临时变量代表这个表达式）。此后，所有对变参的操作都是根据变参箭头所指对实参变量进行的。

（3）函数名由于具有值的类型，也可看作为一个局部变量，但不同的是当此层函数调用结束时，需将这个函数名所具有的值返回到上一层的调用者（可用箭头来指向）。

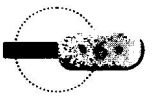
（4）主程序、过程或函数按运行中的调用关系由上向下分层，各层说明的变量（包括形式参数和函数）都依次列于该层首列，各变量值的变化情况按时间顺序记录在与该变量对应的同一行上。

注意：动态图描述规则仅能够描述参数传递中的传值与传地址两种。

2．过程与函数执行描述示例

示例 1 试分析下面程序输出的结果，其中过程 silly 中的形参 x 为传值方式，而形参 y 为传地址方式。

```
PROGRAM varment(output);
  VAR   x,y,z : integer;
  PROCEDURE   silly (x,y);
    VAR z : integer;
    BEGIN  {silly}
      X:=1; y:=12; z:=14
    END;  {silly}
  BEGIN
    x:=1; y:=2; z:=3;
```



```
        silly(y,x);
         write(x,y,z)
    END.
```

【分析】

过程 silly 的形式参数 x 是值参，在过程调用时它接受了全局变量 y 的值 2；而形式参数 y 是变参，过程调用中它存放的是实参变量 x 的地址（图 1.3 中用箭头表示 y→x 的指向）。过程 silly 中还定义了一个局部变量 z，它与全局变量 z 同名。由名字作用域的“最近嵌套原则”知：过程 silly 执行中是局部变量 z 起作用，而在过程 silly 之外是全局变量 z 起作用，局部变量的 z 并不存在。图 1.3 的动态图描述了整个程序的执行情况，其中包括过程 silly 因调用而创建、运行以及运行结束撤销的全过程，所有全局变量和局部变量的变化情况也在动态图中描述出来。动态图中带下划线 _ 的数值为主程序最后的输出结果，虚线用来按时间顺序分隔各个变量值的变化情况。从动态图上可以看出：过程 silly 中的变参 y 这一行上没有值出现，所有对变参 y 进行的操作都是根据 y 指针的指向对实参变量 x 进行的，它们分别位于相邻的两层上。由动态图可知程序的输出结果为：x=12;y=2;z=3。

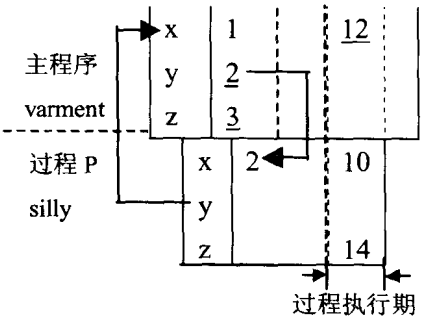


图 1.3　程序运行和过程调用动态图

示例 2　分析下面程序的输出结果，其中函数 f 中的形式参数 x 为传地址方式，n 为传值方式。

```
PROGRAM parament (output);
  VAR m,n:integer;
      x,y:real;
  FUNCTION f(x,n):real;
    BEGIN {f}
      x:=x/n;n:=n+1;m:=m-1;
      f:=x*y
    END; {f}
BEGIN
      n:=5;m:=2;y:=10;
      x:=f(y,n);
      write(n,m,x,y)
END.
```

【分析】

我们把函数名 f 看成一个局部变量，该程序执行的动态图如图 1.4 所示。从动态图中可以看出：函数 f 执行中，全局变量 m 和 y 的值都被改变。因为 x 指向实参 y，对 x 的赋值实际上是对实参 y 赋值；其次，函数 f 执行中对全局变量 m 进行了计算和赋值，故 m 值也被改变。由动态图可知，函数 f 执行结束时函数名 f 具有值 4，这个值在函数 f 结束时被返回给主程序中的全局变量 x（动态图中函数名 f 值 4 的箭头所指）。最终输出结果如下：

n=5;　m=1;　x=4;　y=2

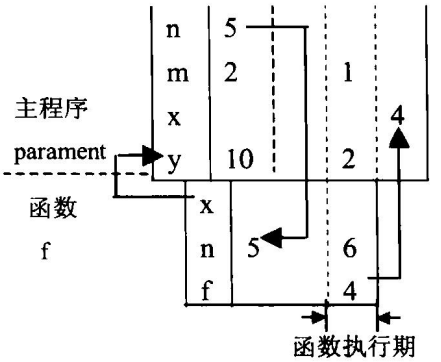


图 1.4　程序运行和函数调用动态图

1.2　典型例题解析

1.2.1　概念题

例题 1.1

单项选择题

1．将编译程序分成若干个“遍”是为了____。

a. 提高程序的执行效率

b. 使程序的结构更加清晰

c. 利用有限的机器内存并提高机器的执行效率

d. 利用有限的机器内存但降低了机器的执行效率

2．构造编译程序应掌握____。（陕西省 2000 年自考题）

a. 源程序　　b. 目标语言

c. 编译方法　　d. 以上三项都是

3．变量应当____。

a. 持有左值　　b. 持有右值

c. 既持有左值又持有右值　　d. 既不持有左值也不持有右值

4．编译程序绝大多数时间花在____上。（陕西省 1998 年自考题）

a. 出错处理　　b. 词法分析
c. 目标代码生成　　d. 管理表格

5. ____不可能是目标代码。　　（陕西省 1997 年自考题）
a. 汇编指令代码　　b. 可重定位指令代码
c. 绝对指令代码　　d. 中间代码

6. 数组 A[1…20,1…10]的首地址偏移量为 0，按列存储，每个元素占一个字节，存储器按字节编址，则 A[i,j]的偏移地址为____。
a. (i-1)×10+(j-1)　　b. (i-1)×20+(j-1)
c. (i-1)+(j-1)×10　　d. (i-1)+(j-1)×20

7. 使用____可以定义一个程序的意义。
a. 语义规则　　b. 词法规则
c. 产生规则　　d. 左结合规则

【解答】

1. 将编译程序分成若干个“遍”是为了使编译程序的结构更加清晰，故选 b。

2. 构造编译程序应掌握源程序、目标语言及编译方法等三方面的知识，故选 d。

3. 对编译而言，变量既持有左值又持有右值，故选 c。

4. 编译程序打交道最多的就是各种表格，因此选 d。

5. 目标代码包括汇编指令代码、可重定位指令代码和绝对指令代码 3 种，因此不是目标代码的只能选 d。

6. 选 d。

7. 词法分析遵循的是构词规则，语法分析遵循的是语法规则，中间代码生成遵循的是语义规则，并且语义规则可以定义一个程序的意义。因此选 a。

例题 1.2

多项选择题

1. 编译程序各阶段的工作都涉及到____。　　（陕西省 1999 年自考题）
a. 语法分析　　b. 表格管理　　c. 出错处理
d. 语义分析　　e. 词法分析

2. 下列各项中，____与数组元素的地址有关。　　（陕西省 1999 年自考题）
a. 数组第一个元素的存储地址　　b. 数组的存储方式
c. 数组的维数　　d. 内存的编址方式　　e. 编译程序

3. 编译程序工作时，通常有____阶段。　　（陕西省 1998 年自考题）
a. 词法分析　　b. 语法分析　　c. 中间代码生成
d. 语义检查　　e. 目标代码生成

4. 过程调用的参数传递方式有____。
a. 传名　　b. 传值　　c. 传参数
d. 传地址　　e. 传结果

5. 编译过程中所遵循的规则有____。
a. 等价变换规则　　b. 短语规则　　c. 构词规则

d. 语义规则　　e. 语法规则

【解答】

1．编译程序各阶段的工作都涉及到表格管理与出错处理，故选 b、c。

2．数组元素的地址与其第一个元素的存储地址、数组的存储方式、数组的维数及内存的编址方式有关；故选 a、b、c、d。

3．编译程序工作通常分为词法分析、语法分析、中间代码生成、代码优化及目标代码生成等 5 个阶段；故应选 a、b、c、e。

4．参数传递方式分为传值、传名、传地址与传（得）结果 4 种，在此选 a、b、d、e。

5．代码优化的遵循的是等价变换规则，再由例题 1.1 题 7 可知应选 a、c、d、e。

例题 1.3

填空题

1．解释程序和编译程序的区别在于_____。

2．编译过程通常可分为 5 个阶段，分别是_____、语法分析、_____、代码优化和目标代码生成。（陕西省 1999 年自考题）

3．编译程序工作过程中，第一阶段输入是_____，最后阶段的输出为_____程序。（陕西省 1997 年自考题）

4．静态数组元素地址的计算公式由两部分确定，一部分是_____，它在_____时确定；另一部分是_____，它在_____时确定。

5．把语法范畴翻译成中间代码所依据的是语言的_____。（陕西省 2000 年自考题）

6．目标代码可以是_____指令代码或_____指令代码或绝对机器指令代码。（陕西省 2000 年自考题）

【解答】

1．是否生成目标程序

2．词法分析　中间代码生成

3．源程序　目标代码

4．常量部分　编译　变量部分　运行

5．语义规则

6．汇编　可重定位

例题 1.4

判断题

1．赋值号左边的变量只持有左值，不持有右值。（　）

2．二维数组 a[3…15,5…20]按行存放，并且每个元素占用一个存储单元，则数组元素 a[i,j]的地址为 base-85+i×16+j（base 是数组 a 存放的起始地址）。（　）

3．变量的名字用标识符来表示，同时名字代表一定的存储单元，有属性、值、作用域等特性。（陕西省 1997 年自考题）（　）

4．指示器变量的右值只持有右值，没有左值。（陕西省 2000 年自考题）（　）