信息与电气工程学院

C程序设计案例教程

第2章 语言程序的构成与运行环境

本章主要内容有

- C语言程序设计的发展
- C程序设计特点
- C程序设计组成结构
- C语言编辑调试与编译运行步骤
- C程序程序例子
- C语言程序设计语义规范

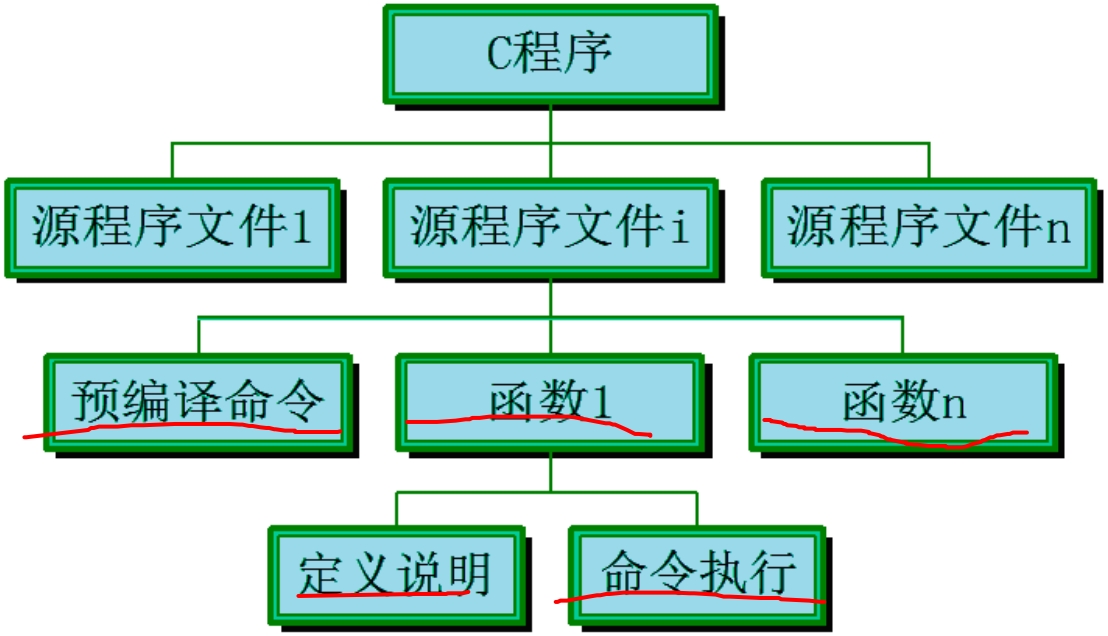
2.1 C语言程序设计的发展

C语言是70年代贝尔实验室为描述UNIX操作系统和C编译程序而开发的一种系统地过程描述语言。

20世纪80年代提出了面向对象的程序设计（Object-Oriented programming)的概念，在C语言面向过程基础上，增加了面向对象的机制，C++应运而生，既可用于结构化程序设计也可用于面向对象的程序设计，与C语言兼容。

2.2 C程序设计组成结构

一个完整的C程序设计基本结构如图2.1所示：



2.3 C程序设计的编译与运行

计算机程序是由计算机语言命令组成的指令序列的集合。任何计算机程序设计语言源程序（source program）均需要使用计算机事先安装好的“翻译”软件，即编译程序或解释程序，翻译成系统可直接识别的机器指令才能实际运行。

C程序设计编译运行步骤工作流程如图2.2所示。

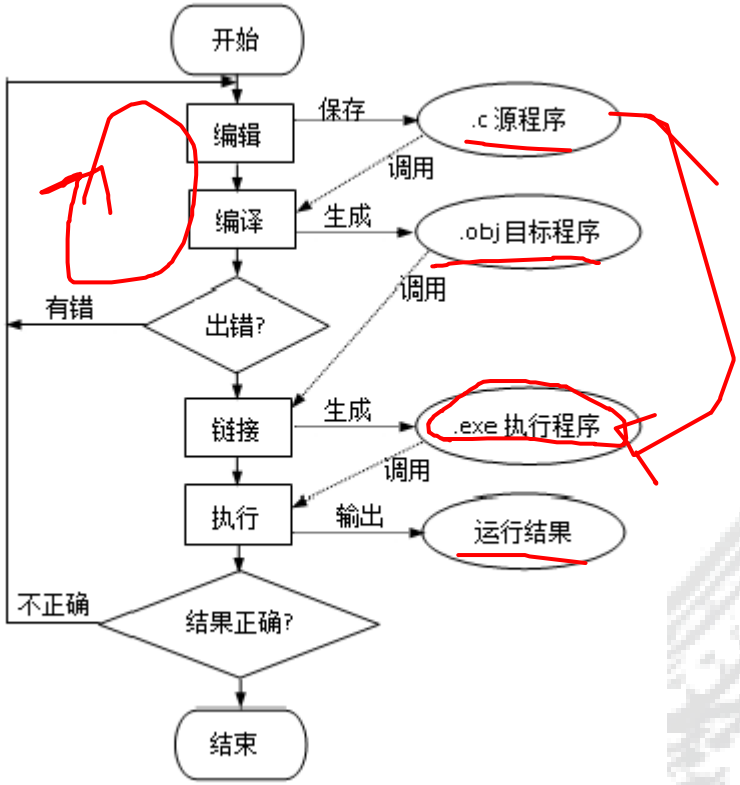


图2.2 C程序设计编译运行步骤及工作流程

2.4 简单的C程序介绍

下面先介绍几个简单的C程序，然后从中分析C程序的特性。

例 1 第一个程序This is a c program .

```
/* example1.1  The  first  C  Program*/
#include <stdio.h>
int main()
{
    printf("This is a C program.\n");
    return 0;
}
```

注释

编译预处理

主函数

语句

输出：
This is a C program.

注 printf语句中的“\n”是换行符

（1）本程序的作用是输出以下一行信息:

This is a C program.

（2） **main** 表示“主函数”。每一个C程序都必须有一个 **main** 函数。函数体由大括弧{}括起来。本例中主函数内只有一个输出语句，

（3）**printf**是C语言中的输出函数(详见第4章)。双引号（双括号）内的字符串原样输出。“\n”是换行符，即在输出“**This is a C program.**”后回车换行。

（4）语句最后有一分号。

例2
求两个数的和

```
/*     example2   calculate the sum of a and b*/
#include <stdio.h>
/*  This is the main program  */
int main()
{    int a,b,sum;     /*定义变量*/
     a=10;
     b=24;
     sum=a+b;
     printf("sum= %d\n",sum);
     return 0;
}
```

预处理命令

函数

注释

语句

注

printf语句中的“ %d ”是表示“十进制整数类型”

运行结果：
sum=34

（1）本程序的作用是求两个整数a和b之和sum。

（2）/*……*/表示注释部分，为便于理解，我们用汉字表示注释，当然也可以用英语或汉字拼音作注释。注释只是给人看的，对编译和运行不起作用。注释可以加在程序中任何位置。

（3）第5行是声明部分， 定义变量a和b，指定 a和b为整型(int)变量。

（4）第6,7行是两个赋值语句，使a和b的值分别为10和24。

（5）第8行使sum的值为a+b，

（6）第9行中“%d”是输入输出的“格式字符串”，用来指定输入输出时的数据类型和格式(详见第4章)，“%d”表示“以十进制整数形式输出”。

在执行输出时，此位置上代以一个十进制整数值。printf函数中括弧内最右端sum是要输出的变量，现在它的值为579(即123+456之值)。因此输出一行信息为sum is 579

例2
求两个数的和

```
/*     example2   calculate the sum of a and b*/
#include <stdio.h>
/* This function calculates the sum of x and y   */
int add(int x,int y)
{   int  z;
    z=x+y;
    return(z);
}

/*  This is the main program  */
int main()
{    int a,b,sum;     /*定义变量*/
     a=10;
     b=24;
     sum=add(a,b);
     printf("sum= %d\n",sum);
}
```

预处理命令

注释

函数

语句

注 printf语句中的“ %d ”是表示“十进制整数类型”

运行结果:
sum=34

例3 输出两个数中较大的数

```
#include <stdio.h>
int max(int x,int y)
{   int  z;
    if(x>y) z=x;
        else z=y;
    return(z);
}
int main()
{
      int a,b,c;
      scanf("%d,%d",&a,&b);
      c=max(a,b);
      printf("max = %d",c);
}
```

输入：10,20 ↵
输出：max = 20

定义max子函数，函数值、形参x、y为整型

声明部分，定义变量

调用max函数，返回值赋给c

注 scanf语句中“&a” 的含义是“取地址”

（1）本程序包括两个函数:主函数main和被调用的函数max。

max函数的作用是将x和y中较大者的值赋给变量z。

return语句将z的值返回给主调函数main。

返回值是通过函数名max带回到main函数的调用处。

（2）main函数中的scanf是“输入函数”的名字(scanf和printf都是C系统提供的标准输入输出函数)。

程序中scanf函数的作用是输入a和b的值。

&a和&b中的“&”的含义是“取地址”，此scanf函数的作用是将两个数值分别输入到变量a和b的地址所标志的单元中，也就是输入给变量a和b。

这种形式是与其他语言不同的。&a和&b前面的“%d，%d”的含义与前相同，只是现在用于“输入”。它指定输入的两个数据按十进制整数形式输入。关于scanf函数详见第4章。

（3）main函数中第4行为调用max函数，在调用时将实际参数a和b的值分别传送给max函数中的形式参数x和y。经过执行max函数得到一个返回值(即max函数中变量z的值)，把这个值赋给变量c。然后输出c的值。

（4）printf函数中双引号内的“max=%d”，在输出时，其中“%d”将由c的值取代之，“max =”原样输出。

程序运行情况如下:

10，20　(输入10和20给a和b)

max=20　　(输出c的值)

2.5 C语言程序设计语义规范

- 通过以上几个例子，可以看到:

(1)函数与主函数

- C程序是由函数构成的:
  - 程序中的全部工作都是由各个函数分别完成的;
  - 编写C程序就是编写一个个函数;
  - C的这种特点使得容易实现程序的模块化。
- 必须有且只能有一个主函数main():
  - 一个C程序总是从main函数(简称主函数)开始执行的;可
  - 以放在程序中任一位置
- 被调用的函数可以是系统提供的库函数(例如printf和scanf函数)，也可以是用户根据需要自己编制设计的函数(例如，例1.3中的max函数)。

(2) 一个函数由两部分组成:

① 函数的首部，即函数的第一行。

包括函数名、函数类型、函数属性、函数参数(形参)名、参数类型。

例如，例1.3中的max函数的首部

```
int      max     (int          x ,      int          y )
 ↓        ↓       ↓            ↓        ↓            ↓
函数类型 函数名  函数参数类型  函数参数名 函数参数类型 函数参数名
```

一个函数名后面必须跟一对圆括弧， 函数参数可以没有，如main( )。

② 函数体，即函数首部下面的大括弧{……}内的部分。如果一个函数内有多个大括弧，则最外层的一对{ }为函数体的范围。

函数体一般包括:

i）*声明部分*：在这部分中定义所用到的变量。

例1.3中main函数中的“int a,b,c;”。

ii）*执行部分*：由若干个语句组成。

(3)函数体的语义规范

i）C程序书写格式自由，一行内可以写几个语句， 一个语句可以分写在多行上。

ii）每个语句和数据定义的最后必须有一个分号。分号是C语句的必要组成部分。即使是程序中最后一个语句也应包含分号。

例如:

```
c=a+b;
```

iii）可以用/*……*/对C程序中的任何部分作注释。一个好的、有使用价值的源程序都应当加上必要的注释，以增加程序的可读性。

iv)标识符：是用户程序设计过程中需要使用的变量、函数等。
由用户自己定义字符标识，以便调用。
在定义函数或说明变量时，必须用到标识符命名。

例如：

```
float  p,s,v,r;    /*定义4个浮点类型变量*/
float s_volume(float x)   /* 定义圆球体积函数*/
{ return(4.0/3.0*PI*pow(x,3); }
```

(4) 包含文件定义

①包含文件定义又称头文件说明，其格式为：

#include <文件名>或

#include “文件名”

②宏定义

#define PI 3.1415926535

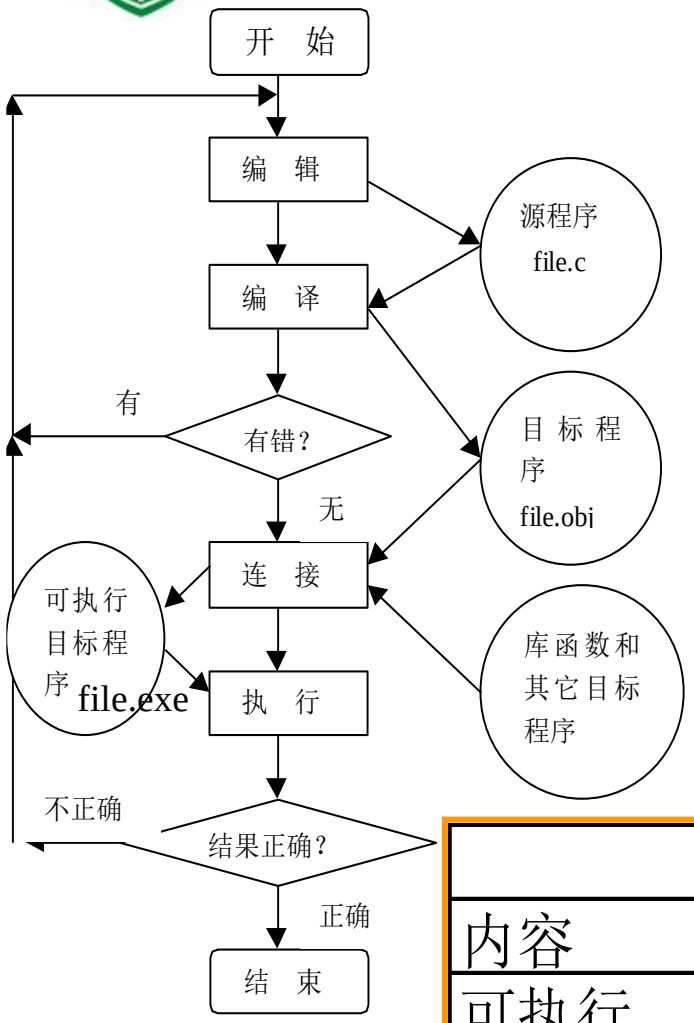
– C语言格式特点

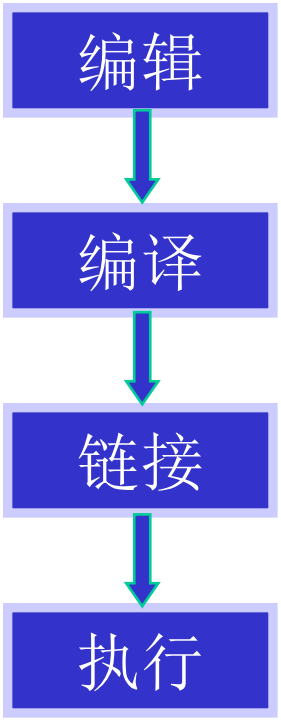
- 习惯用小写字母，大小写敏感
- 不使用行号，无程序行概念
- 可使用空行和空格
- 常用锯齿形书写格式

优秀程序员的素质之一：

- 使用TAB缩进
- {}对齐
- 有足够的注释
- 有合适的空行

```
int main()
{
   int  i, j, sum;
   sum = 0;
   for (i = 1; i < 10; i++)
   {
      for (j = 1; j < 10; j++)
      {
         sum += i * j;
      }
   }
   printf("%d",sum);
   return 0;
}
```





程序代码的录入，生成源程序*.c

语法分析查错，翻译生成目标程序*.obj

与其它目标程序或库链接装配,生成可执行程序*.exe

	源程序	目标程序	可执行程序
内容	程序设计语言	机器语言	机器语言
可执行	不可以	不可以	可以
文件名后缀	.c或.cpp	.obj	.exe

2.6 用程序设计语言描述

1. 机器语言 (Machine language)

例如，计算表达式m / n-z的值，并把结果值存到10010000号内存单元。假设已知某计算机的取数操作码为1000，除法操作码为1010，减法操作码为1001，传送操作码为0100，另外也知m、n、z中的三个数已分别存放在11110110、10101101、01010110号内存单元。用机器语言可描述编写如下程序：

1000	11110110	取出放在11110110内存单元的值
1010	10101101	除法操作放在10101101内存单元的值
1001	01010110	把结果值减去放在10101101内存单元的值
0100	**10010000**	把最后结果值存到**10010000**号内存单元

2.汇编语言(Assembler language)

例如计算表达式m / n-z值的程序可以写成：

```
LDA     M
DIV     N
SUB     Z
MOV     Y
```

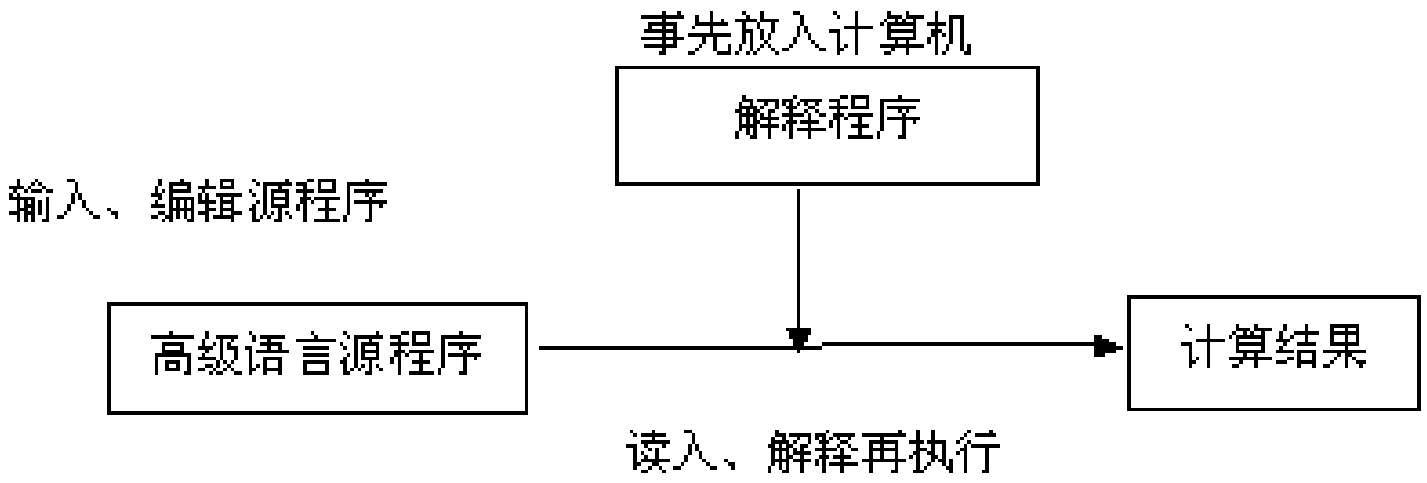
使用这种语言计算机CPU不能直接识别，必须用事先存放在存储器中的“翻译程序”，把汇编语言翻译成机器语言,计算机指令系统才能识别和执行，这个翻译程序称为汇编程序,翻译成机器语言描述的程序叫目标程序。

3. 高级语言(High-level language)

不管使用机器语言还是使用汇编语言描述算法和编写程序，都没有摆脱计算机指令系统的束缚。到了1954年，出现了一种与具体计算机指令系统无关的语言，即高级语言。它与人们习惯使用的自然语言与数学语言非常接近，例如，$y=2x^2-x+1$这样一个数学式子用高级语言来写，就写成y=2*x*x-x+1，基本上是原样表达，不需要再分步骤。

而且不同的计算机系统上所配置的高级语言基本上都是相同的，即高级语言具有很强的通用性，这样描述程序算法显然就得心应手的多。

高级语言的解释过程如图2.3所示。



高级语言的编译过程如图2.4所示。