第4章 结构化程序设计

4.1 结构化程序设计方法

4.2 结构化程序三种基本结构

4.3 顺序结构程序设计

4.4 选择结构程序设计

4.5 循环结构程序设计

本章内容导读

- 本章主要介绍结构化程序设计方法以及与三种基本结构相关的语句。通过学习本章，读者应掌握以下内容：
  - 了解结构化程序的三种基本结构；
  - 掌握字符和格式输入/输出函数的调用格式和功能；
  - 掌握单、双和多分支选择语句的格式和功能；
  - 掌握while语句、do-while语句、for语句、break语句和continue语句的格式和功能。

4.1 结构化程序设计方法

- **结构化程序设计（Structured Programming）**

以模块功能和处理过程设计为主的详细设计的基本原则。

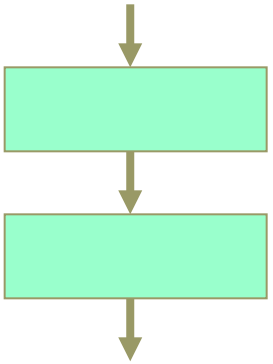
- 1965 年由E.W.Dijkstra 提出。
- **基本思路**

把一个复杂问题的求解过程分阶段进行，每一阶段处理的问题都控制在人们容易理解和处理的范围内。

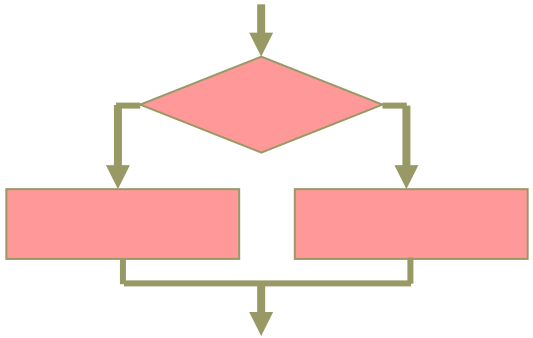
- **具体方法**
  - 自顶向下，逐步求精
  - 模块化设计
  - 结构化编码

4.2　结构化程序三种基本结构

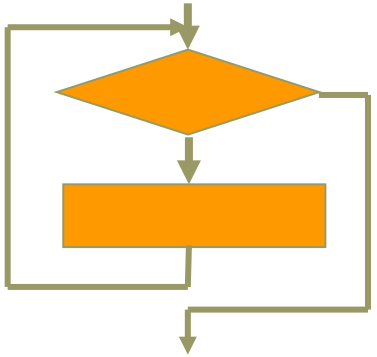
一、顺序结构



二、选择结构



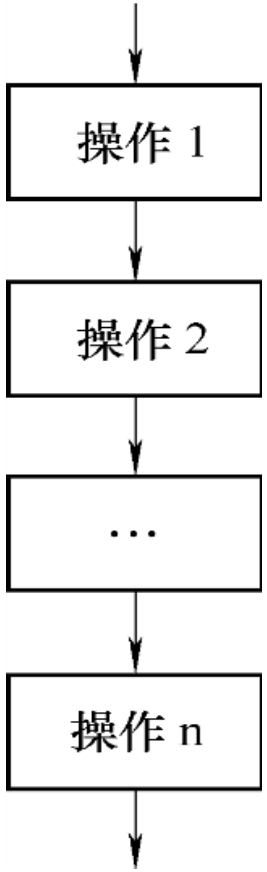
三、 循环结构



4.2　结构化程序三种基本结构

1、顺序结构

顺序结构是按书写顺序依次执行语句操作。

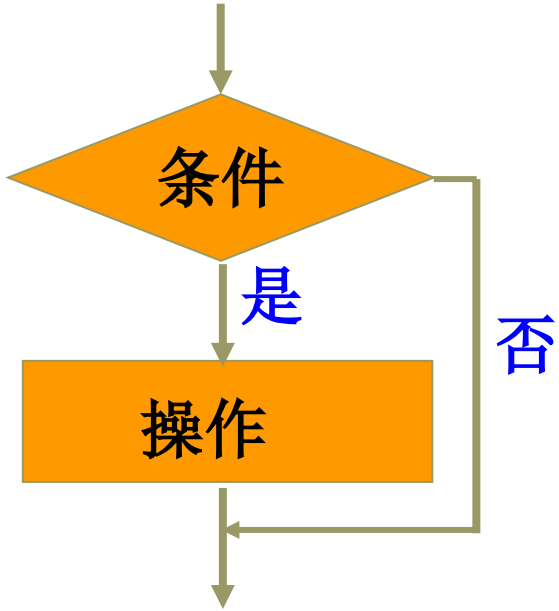


4.2　结构化程序三种基本结构

2、选择结构

➢ 单分支选择结构

仅包含一个条件，其功能是根据条件是否成立，决定是否执行某个操作。



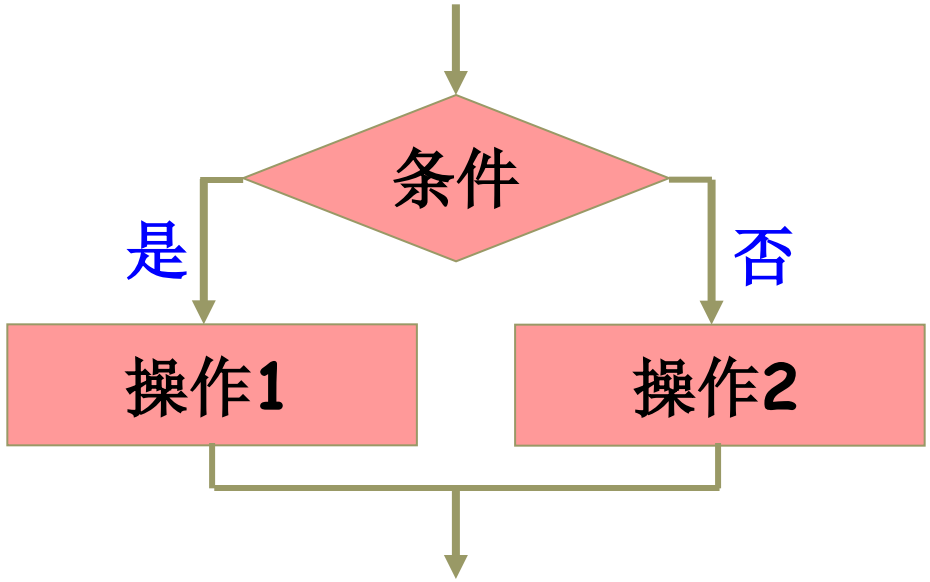
注：*表达式可以是任意表达式，若“非0”则执行操作。*

4.2 结构化程序三种基本结构

2、选择结构

- **双分支选择结构**

  仅包含一个条件，其功能是根据条件是否成立，决定从两个操作中选取一个操作执行。

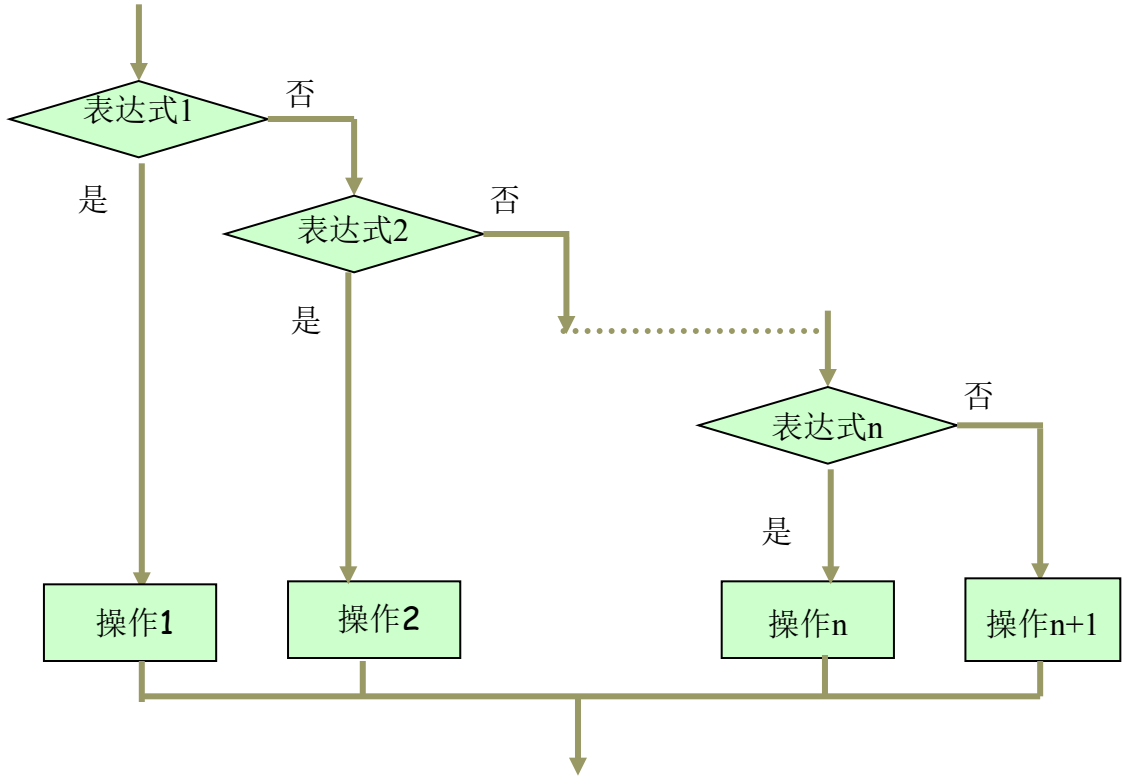


4.2　结构化程序三种基本结构

2、选择结构

- **多分支选择结构**

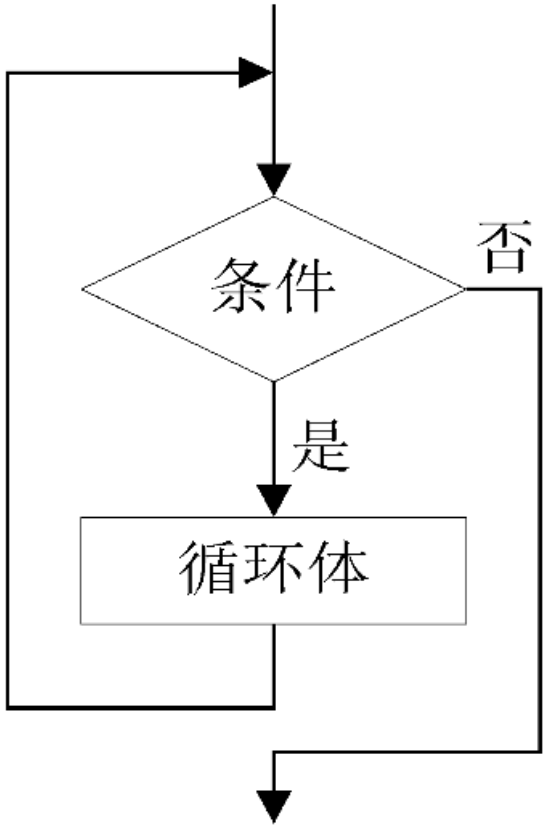
包含n（n>1）个条件，其功能是根据某个条件是否成立，从n+1个操作中选取一个操作执行。



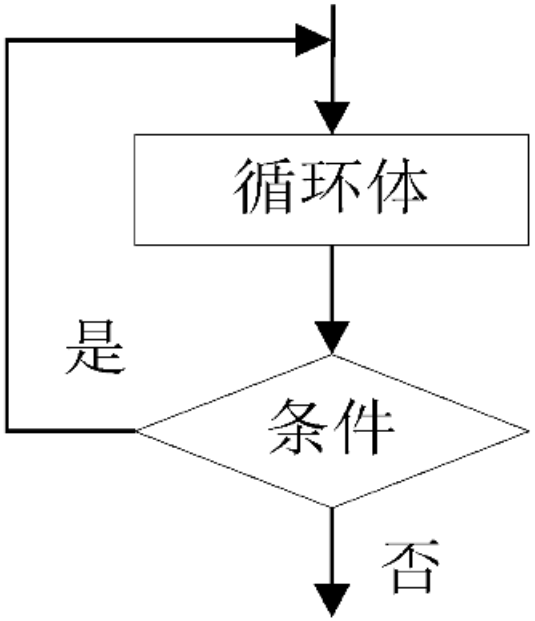
4.2　结构化程序三种基本结构

3、循环结构

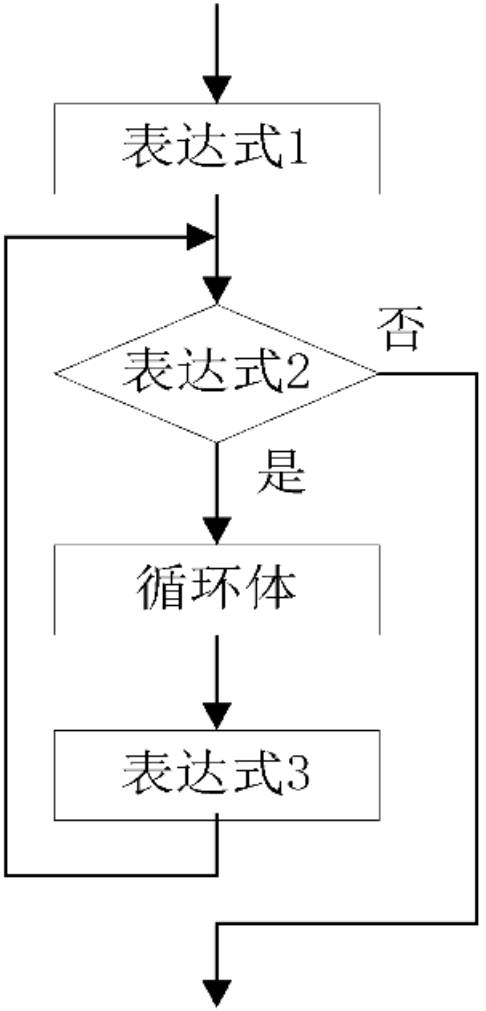
当型循环结构



直到型循环结构



次数型循环结构



4.3 顺序结构程序设计

➢ 顺序结构的程序组成

（1）程序开头的编译预处理命令。

程序中要使用的库函数（又称系统函数），除了print( )和scanf( )函数外，其它的都必须使用编译预处理命令将相应的头文件包含进来。

（2）顺序结构的程序由完成具体功能的各个语句和运算组成。主要包括：

- ✓ 变量类型的说明
- ✓ 提供数据的语句
- ✓ 运算部分
- ✓ 输出部分

4.3 顺序结构程序设计

1、赋值语句

- 格式一　**变量=表达式;**
- 功能：计算表达式的值，然后赋予变量。
- 格式二　**变量 复合赋值运算符 表达式;**
- 功能：将变量和表达式进行指定的算术或位运算后，再将运算结果赋予变量。

2、函数调用语句

C语言中事先编好的函数（称为库函数或系统函数）或用户自定义函数（用户函数）都是通过函数调用实现其功能的，并获得函数的返回值。

- 格式　**函数名(参数1, 参数2, …, 参数n);**
- 功能：调用指定的库函数或用户自定义函数，对圆括号中的参数进行该函数指定的运算，运算结果作为函数的返回值返回。

4.3 顺序结构程序设计

3、表达式语句

- 格式 表达式;
- 功能：计算表达式的值。

4、复合语句

格式
```
{
    语句1;
    语句2;
    …
    语句n;
}
```

- 功能：依次执行语句1，语句2，…，语句n。
- 说明（1）复合语句可以包含多条语句，但整体上是作为一条语句看待。(2)复合语句中若有数据定义语句，则应放在复合语句中其它语句的前面。

4.3 顺序结构程序设计

5、字符输入/输出函数

C语言中的输入和输出操作，均是利用C语言编译系统提供的库函数实现的，C语言本身并没有提供这两类语句。

➢ **字符输出函数putchar( )**

- 调用格式：putchar(ch)
- 功能：把一个字符输出到标准输出设备（显示器）。
- 参数：ch可以是一个整型变量、字符型变量、整型常量或字符型常量，也可以是一个转义字符或整型表达式，但不能是字符串。
- 返回值：输出ch对应的单个字符。
- 说明：源程序（文件）开头需要加入编译预处理命令#include <stdio.h>或#include "stdio.h"。

4.3 顺序结构程序设计

5、字符输入/输出函数

➢ 字符输入函数getchar( )

- 调用格式：getchar( )
- 功能：从标准输入设备（键盘）上读入一个字符。
- 参数：无参数。
- 返回值：输入的单个字符。
- 说明

（1）getchar( )函数只能用于单个字符的输入，一次输入一个字符。输入的数字也是按字符处理。当输入多于一个字符时，只接收第一个字符。

（2）getchar( )函数输入的字符可以赋给一个字符变量或整型变量，也可以不赋给任何变量，而作为表达式的一部分。

（3）源程序（文件）开头需要加入编译预处理命令#include <stdio.h>或#include "stdio.h"。

4.3 顺序结构程序设计

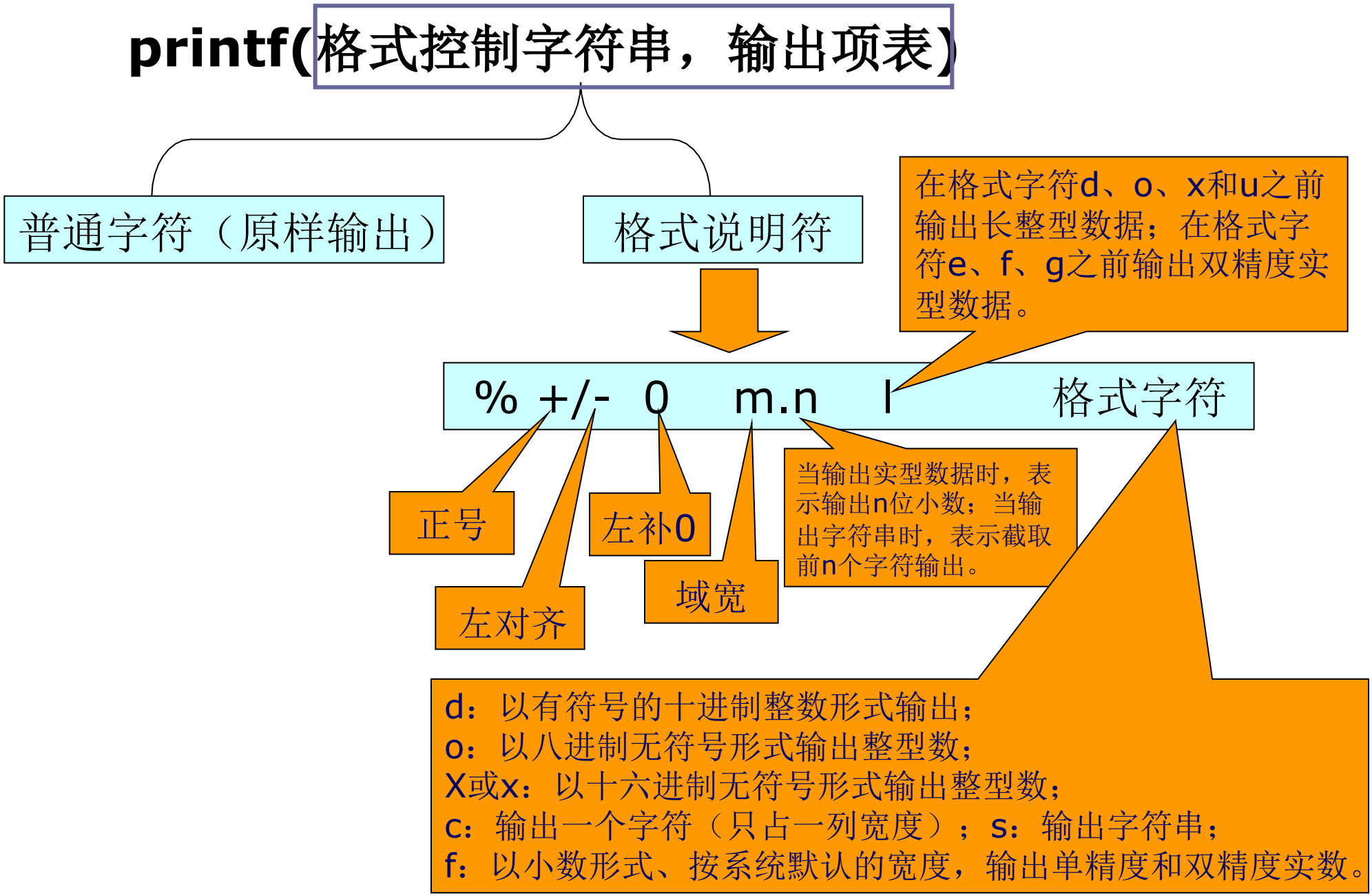
6、格式输入/输出函数

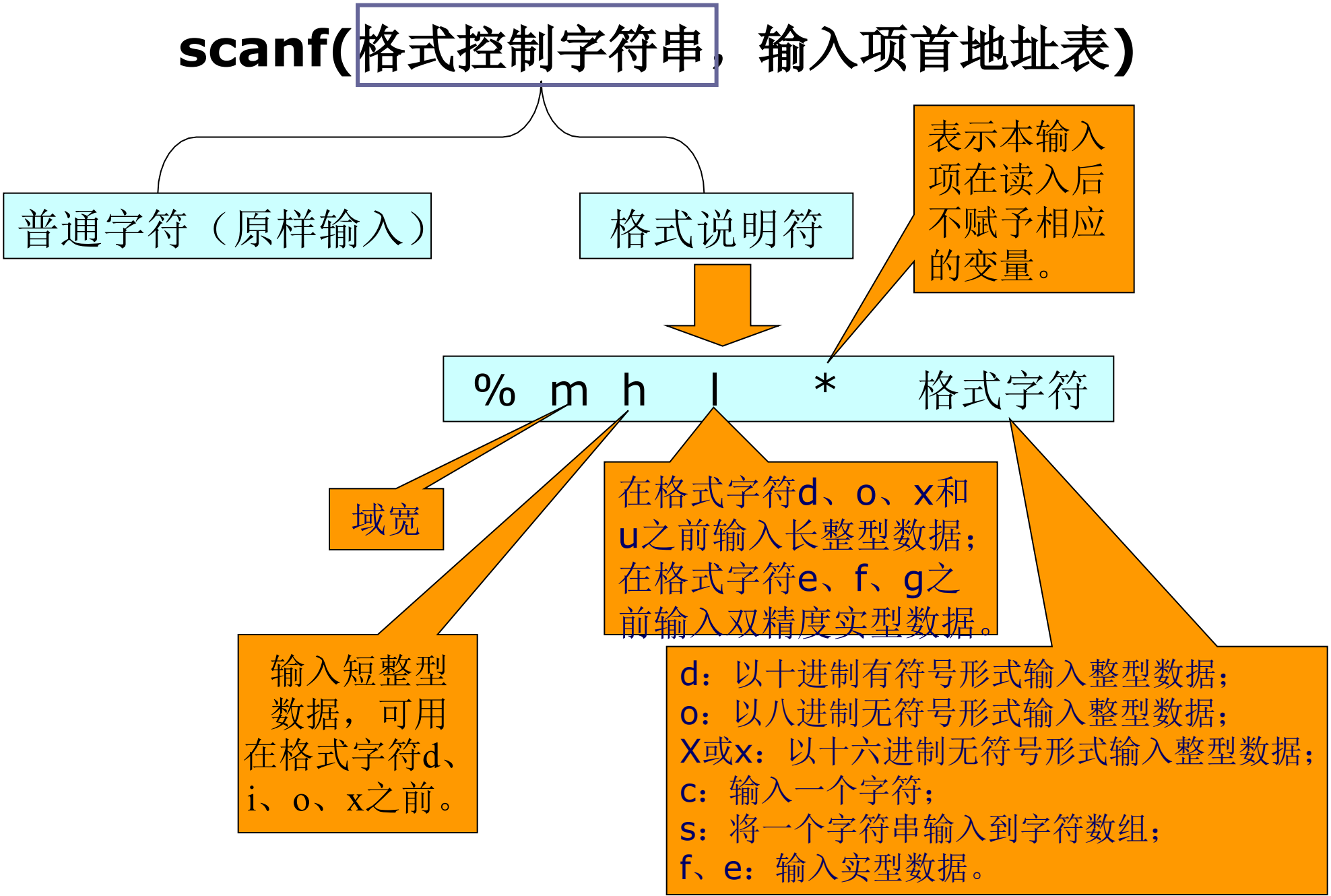
➢ 格式输出函数printf( )

- 调用格式：printf(格式控制字符串，输出项表)
- 功能：按照用户指定的格式把指定的数据显示到标准输出设备（显示器）。
- 说明（1）格式说明符与输出项一一对应，若格式说明符的个数少于输出项个数时，则多余的输出项不输出；若格式说明符的个数多于输出项个数时，则对缺少的项输出不确定值。（2）不需要编译预处理命令。

➢ 格式输入函数scanf( )

- 调用格式：scanf(格式控制字符串，输入项首地址表)
- 功能：从键盘按照“格式控制字符串”中规定的格式读取若干个数据，然后按照“输入项首地址表”中的顺序，依次将数据存入相应的变量。
- 说明：不需要编译预处理命令。


printf(格式控制字符串，输出项表)
普通字符（原样输出）
格式说明符
% +/- 0 m.n l 格式字符
正号
左对齐
左补0
域宽
当输出实型数据时，表示输出n位小数；当输出字符串时，表示截取前n个字符输出。
在格式字符d、o、x和u之前输出长整型数据；在格式字符e、f、g之前输出双精度实型数据。
d：以有符号的十进制整数形式输出；
o：以八进制无符号形式输出整型数；
X或x：以十六进制无符号形式输出整型数；
c：输出一个字符（只占一列宽度）；s：输出字符串；
f：以小数形式、按系统默认的宽度，输出单精度和双精度实数。


scanf(格式控制字符串，输入项首地址表)
普通字符（原样输入）
格式说明符
表示本输入项在读入后不赋予相应的变量。
% m h l * 格式字符
域宽
在格式字符d、o、x和u之前输入长整型数据；在格式字符e、f、g之前输入双精度实型数据。
输入短整型数据，可用在格式字符d、i、o、x之前。
d：以十进制有符号形式输入整型数据；
o：以八进制无符号形式输入整型数据；
X或x：以十六进制无符号形式输入整型数据；
c：输入一个字符；
s：将一个字符串输入到字符数组；
f、e：输入实型数据。

注意

①如果相邻两个格式字符之间，没有指定数据分隔符（如逗号、冒号等），则相应的两个输入数据之间，至少用一个空格分开，或者用Tab键分开，或者输入一个数据后按回车，然后再输入下一个数据。

例如　int num1,num2;

　　scanf("%d%d",&num1,&num2);

若num1 输入12，num2 输入36，则正确的输入操作如下：

　　12□36←┘

或者 12←┘

　　36←┘

②输入实数时不能规定精度。

例如，float x;

　　scanf("%7.2f",&x);是错误的。

注意

③格式控制字符串”中出现的普通字符（包括转义字符），务必原样输入。

例如　int num1,num2;

　　scanf("%d,%d",&num1,&num2);

若num1 输入12，num2 输入36，则正确的输入操作如下：

12,36←┘

④在以%d、%f、%lf、%e 输入数值型数据时，遇到以下情况，系统认为数据输入结束。

- ✓ *遇到空格或回车键或者Tab键输入结束，可用它们作为数据之间的分隔符。*
- ✓ *遇到宽度结束。例如“%3d”，只取3列，即系统自动按域宽截取所需数据。*
- ✓ *遇到输入数据与格式说明符类型不一致，则输入结束。*
- ✓ *遇到非法输入结束。例如在输入数值型数据时，遇到字母等非数值符号（数值符号仅由数字字符0～9、小数点和正、负号构成）。*