计导基础

//表达式

表达式（expression）是由运算符、操作数和括号组成的、计算求值的基本单位。

操作数可以是：常量、变量、函数调用和表达式；

表达式的定义是一个递归的形式，由下面四条规则构成:  
（1）常量、变量、函数调用是表达式。  
（2）如果Ω是表达式，则(Ω)是表达式；如果θ是单目运算符，θ允许前缀形式，则θΩ是表达式；如果θ允许后缀形式，Ωθ是表达式。  
（3）若Ωl、Ω2、Ω3是表达式，θ是双目运算符，则Ωl θ Ω2是表达式；θ1和θ2构成三目运算符，则Ωl θ1 Ω2 θ2 Ω3是表达式。  
（4）只有有限次的使用上述规则得到的有限长度的字符串式子，才是表达式。

//典型的几类运算符和表达式 ：

算术运算符和算术表达式；

关系运算符和关系表达式； 逻辑运算符和逻辑表达式； 赋值运算符和赋值表达式；

//算术运算符和算术表达式

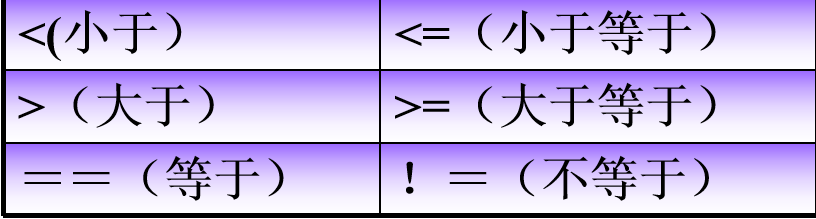
算术运算符：

算术表达式：由算术运算符、操作数和括号构成的表达式。

在C语言中，算术运算的操作数可以是整数、字符或者浮点数。如果表达式中的操作数全是字符或者整数，则表达式的值为整数；如果操作数至少有一个是浮点数，则表达式的值为浮点数。提示：字符数据总是先转换成整型数据，再参与运算的。

//关系运算符和关系表达式

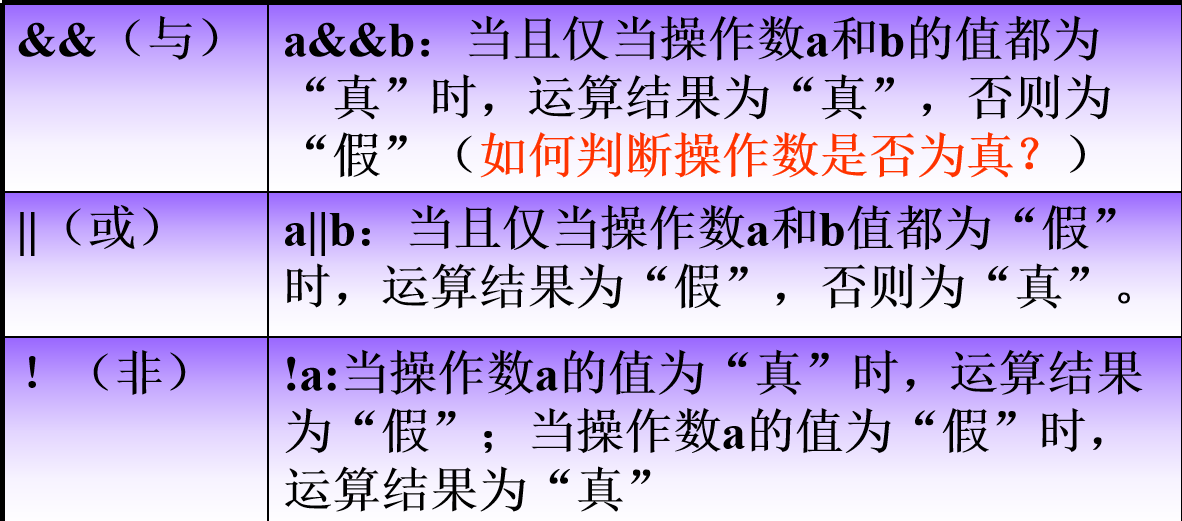
关系运算符：用于比较两个数据的大小关系。



关系表达式：用关系运算符将两个表达式连接起来构成的表达式，如(2＋3)<10 ；运算结果是逻辑“真”或者“假”。在C语言中，若运算结果是逻辑“真”，则关系表达式的值为1；若为逻辑“假”，则值为0。

//逻辑运算符和逻辑表达式

逻辑运算符：与、或、非

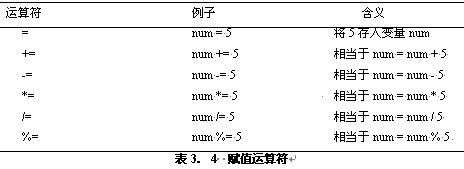


//C语言中，只要操作数值为非0，即为真。值为0，即为假

逻辑表达式：用逻辑运算符将表达式（一般为关系表达式）连接起来构成的表达式 ；运算结果是逻辑“真”或者“假”； 在C语言中，若运算结果是逻辑“真”，则逻辑表达式的值为1；若为逻辑“假”，则值为0。逻辑表达式求值的两个要点：判断操作数值是“真”还是“假”；当逻辑表达式求值结束后，将运算结果的逻辑“真”或“假”转换为整型值1或0。

//赋值运算符和赋值表达式

赋值运算符：用于将值存储到变量中；赋值表达式：带有赋值运算符的表达式 ；赋值表达式的值为赋值后赋值号左边变量的值（左值）；



优先级：算术运算符 高于 关系运算符 高于 逻辑运算符 高于 赋值运算符

//语句

语句（statement）用于向计算机发出操作指令。一条语句经编译后产生一条或多条机器指令。一个实际的程序可能包括若干条语句。C语言规定了6类语句： 表达式语句、复合语句、选择语句、循环语句、标记语句、跳转语句

//表达式语句

表达式：算术表达式、关系表达式、逻辑表达式、赋值表达式。

表达式语句：在一个表达式末尾加上分号“；”（分号是C语言中语句的结束符）。

例：

i<j;（一个表达式语句，但无意义）i = i + 10 \* j;（赋值表达式语句） printf(“age=%d”,getage());（表达式语句，表达式为函数调用）//复合语句 复合语句是包含零个或多个语句的代码单元，使得一组语句成为一个整体，也被称为语句块。

在C语言中，复合语句由一个左大括号、可选语句段、一个右大括号组成。如： { i = 1; j = 2 \* i; }//选择语句

在写程序时，有时往往需要测试某一个条件 是否成立，然后根据测试结果来控制程序后续执行路径。此时要用到选择语句。 C语言中选择语句有： 1）if-else 语句 2）switch语句

if-else C语句:

if (表达式) 语句1 else 语句2

if-else语句中的“表达式”必须用“(”和“)”括起来；表达式可以是任一种表达式，关系表达式和逻辑表达式较常见；表达式的值为非零时，为“真”，否则为“假”。语句可以是任何语句：表达式语句、复合语句、选择语句和循环语句。else子句是if语句的一部分，必须与if配对使用，不能单独使用。可以没有else子句（但必须有if子句），即： if(表达式) 语句//switch语句

实现多路选择常量表达式的值必须是C语言中的整型、字符型或者枚举类型。

switch (表达式){ case 常量表达式1: 语句1[break;] case 常量表达式2: 语句2[break;] ······ case 常量表达式n: 语句n[break;] default： 语句n＋1}//方括号表示其内的语句可选

switch语句实例：

switch (gender){ case 0:printf(“hello, boy!");break; case 1:printf(" hello, girl ");break; default:printf(“invalid gender”);}

//循环结构语句

有时程序中需要多次运行同一段代码（重复做相同的事情）。这种控制结构称为循环结构。在循环结构中，需要刻画出重复执行的是哪些动作（循环体），以及什么条件下需要重复（循环条件）。 C语言提供了三类用于实现循环结构的语句：1）while 2）do-while3）forwhile C语句：

while (表达式) 语句

此处的表达式可以为任意一种表达式,语句为任意类语句；如果表达式值为真(非0），则执行语句；执行完后再测试表达式，如果仍为真（非0），则再次执行语句；直到表达式为假（为0）。

循环体：每次循环做相同的操作，只是操作的是不同的数据。

循环初始化；while（循环条件）{ 本次循环本职工作； 修改相关变量，为下一次循环做准备；}do-while C语句：

do 语句 while (表达式);// 注意此处有语句结束符;

语句组至少会执行一次。

for语句:

for (表达式1；表达式2；表达式3) 语句

先处理表达式1；然后判断表达式2是否成立，若成立，则运行语句，然后执行表达式3，完成一次循环；然后再判断表达式2是否成立，若成立，则执行语句，然后执行表达式3 ……若表达式2不成立，退出循环;

相当于：

表达式1；//循环初始化While（表达式2）{ 语句 表达式3;}

表达式1和表达式3可以是用逗号格开的表达式列表。如：for(i=1, j=50;i<=10&&j>=20;i++, j-= 5 )在for结构中，表达式1和表达式3部分应该只放置包含控制变量的表达式。对其他变量的操作应该放在循环体之前或循环体之后；for结构中的三个表达式不是必须具备的。如果在程序的其他地方初始化了控制变量，则可以省去表达式1；如果省略了表达式2，则假定条件为真，建立了一个“无限循环”；如果在for结构体中计算了递增（递减）表达式或者不需要递增（递减）表达式，则可以省去表达式3。

//函数

函数是封装并给以命名的一段程序代码，完成明确的功能，可供调用。函数是C语言程序设计的基本单位。一个C语言程序是由一个主函数（main）和其他若干个子函数组成的。程序的运行都从主函数开始，也是由主函数结束。主函数可以调用其他函数（子函数）来完成一些特定任务，而子函数还可以调用其他子函数。 主函数由操作系统调用，其他函数由主函数或另外的函数调用。当主函数执行结束，控制便交还给操作系统。

