C 语言知识点总结（上）

第 1 章　程序设计概述

1、计算机语言的发展

低级语言	机器语言	直接识别	二进制
	汇编语言	不能直接识别	需汇编
高级语言	——	不能直接识别	编译：先翻译再执行 解释：边翻译边执行

2、程序与程序设计

⑴程序是指用计算机语言对所要解决问题中的数据以及处理步骤做出的完整而准确的描述，而得到这个描述的过程就称为程序设计。

⑵著名的计算机科学家 Nikiklaus Wirth 给出了一个公式：数据结构+算法=程序。

⑶程序设计步骤：分析问题，建立数学模型；确定数据结构和算法；编制程序；调试程序。

3、结构化程序设计

基本思想是规定几种基本控制结构，然后由这些基本控制结构按一定规律组成程序。特征是自顶向下，逐步求精。三种基本控制结构：顺序结构、选择结构和循环结构。

4、算法

⑴概念：为解决一个问题而采取的方法和步骤，就称为算法。

⑵算法的特征：有穷性；确定性；有效性；有 0 个或多个输入；有 1 个或多个输出。

⑶算法的描述：自然语言、程序流程图、N-S 图、PAD 图、伪代码、计算机语言等。

第 2 章　C 语言概述

1、C 语言特点

⑴语言简练，使用灵活方便；运算符丰富；数据类型丰富；C 语言是结构化的程序设计语言；语法限制不太严格，程序设计自由度大；C 语言允许直接访问物理地址，能进行位（bit）运算，可实现汇编语言的大部分功能，可以直接对硬件进行操作。

⑵与其他高级语言相比，C 语言生成的目标代码质量好，程序执行效率高；与汇编语言相比，用 C 语言编写的程序可移植性好，基本不作修改就可以在各种计算机系统中运行。

2、C 语言的构成

⑴C 语言是由函数构成的。一个 C 程序可以只包含一个主函数（main()），也可以包含一个

主函数和若干个其他函数。用户可以根据需要设计自己的函数，此外，C 语言提供了丰富的标准库函数。

⑵函数由函数首部和函数体两部分组成。函数首部包括函数名、函数返回值的类型、函数参数（形式参数）的名字及类型。函数体包括变量定义部分和执行部分，二者均可缺少。

⑶每个语句后面都要有一个分号，这个分号必不可少。

⑷主函数可以放在程序的任意位置。但无论主函数放在哪，C 程序总是从主函数开始执行，一直执行到主函数结束，程序终止运行。

⑸C 程序没有提供专门的输入输出语句，是通过调用标准库函数实现的。

3、C 语言的书写格式

⑴为了提高程序的可读性，最好一行只写一个语句。

⑵锯齿状编程，即低一层次的语句或定义应该比高一层次的语句或定义向右缩进若干格。

⑶/*　*/注释；// 单行注释；注意注释不能嵌套使用

⑷C 语言中的逗号、分号、单引号和双引号等符号，除非出现在字符串中，否则一定要在英文状态下输入。

4、基本字符集

⑴在 C 源程序中允许出现的所有字符的集合，称为 C 字符集，它是 ASCII 码字符集的一个子集，由字母、数字和特殊字符组成。

⑵字母：包括大写字母 A~Z，小写字母 a~z，区分大小写。

数字：0~9 共 10 个。

特殊字符：如控制字符，分隔符和运算符等。

注意区分一些字形上容易混淆的字符，如 O 和 0，1 和 l。

5、标识符

⑴程序中使用的变量名、数组名、函数名、文件名等，统称为标识符。标识符只能由字母（A~Z、a~z）、数字（0~9）和下划线（_）组成，并且第一个字符必须是字母或下划线。

⑵在标识符中，区分大小写。

⑶C 语言对标识符的字符个数，即长度，没有统一规定。它受各种版本的编译系统的限制，同时也受具体机器的限制。

⑷在满足标识符命名规则的前提下，标识符命名最好“见名知意”。

6、关键字

⑴关键字，又称保留字，是 C 语言规定的具有特定意义的一串字符。用户定义的标识符不能与系统的关键字相同。

⑵数据类型：用于定义变量、函数值或其他数据结构的类型。char、short、int、unsigned、long、float、double、struct、union、void、enum、signed。

⑶存储类型：用于表示变量的存储方式。auto、register、static、extern。

⑷控制类型：用于控制程序的执行流程。break、case、continue、default、do、else、for、

goto、if、return、switch、while。

⑸其他关键字：sizeof、typedef、const、volatile。

7、语句

⑴语句是组成程序的基本单位，能够完成一定的操作。将语句有机地组合起来就能实现所需的计算处理功能。

⑵分类：选择语句（if、switch）；循环语句（while、do while、for）；转移语句（break、continue、return、goto）；表达式语句；复合语句；函数调用语句；空语句。

8、标准库函数

⑴标准库函数不是 C 语言本身的组成部分，它是由 C 编译系统提供的可以直接使用的函数。常见的标准库函数有数学函数、字符处理函数、输入输出函数、动态存储分配函数、图形函数和接口函数等。

⑵标准库函数存放在库文件中，同时关于这些函数的声明和宏定义等信息存放在头文件中。使用标准库函数时，需要使用预处理命令“#include”将相应的头文件包含到用户的源文件中，形式如下：#include <头文件名> 或 #include “头文件名”。

⑶调用输入输出等函数时，需要将标准输入输出头文件 stdio.h 包含进来；

调用三角函数等数学函数时，需要将头文件 math.h 包含进来；

调用字符串处理函数时，需要将头文件 string.h 包含进来；

调用动态存储分配函数时，需要将头文件 malloc.h 包含进来。

9、C 程序从编辑到运行的基本过程

⑴编辑：把语言源程序输入到计算机中，并以文件的形式存放在磁盘上的过程。源程序文件的主名由用户指定，扩展名为.c。

⑵编译：把 C 语言源程序译成二进制形式的目标源程序，扩展名是.obj。这一过程由 C 编译系统提供的编译程序来完成，对源程序进行句法和语法检查。

⑶连接：编译得到的目标程序仍然不能直接执行，还需要将它与库函数或其他目标程序连接装配成可执行程序后才能执行。程序连接由系统提供的连接程序来完成，扩展名为.exe。

⑷运行。

10、上机说明及技巧

⑴C 语言的编译错误有以下两种：

警告（warning）：非致命性错误，一般不影响程序的运行；

错误（error）：必须要改正的错误，否则无法进行下一步操作。

⑵实际的 C 应用程序一般包括多个文件，一般将一个应用程序作为一个工程来处理，所有文件形成一个有机的整体。将工程置于工作区的管理之下，一个工作区可以管理多个工程。同一个工作区中的工程之间相互独立，但公用一个工作区的环境设置。开发应用程序之前，应该首先创建工作区和工程，再创建文件。学习 C 语言时，为简单方便，通常在系统自动创建的默认工作区和工程的环境下直接创建文件即可。

⑶调试：见 PPT。

第 3 章　数据类型、运算符和表达式

1、C 的数据类型

⑴在程序设计语言中，用数据类型来描述不同数据的特点。数据类型规定了数据的取值范围、存储方式以及允许进行的运算。每个数据都属于某种数据类型。

⑵C 语言具有丰富的数据类型，将其分为基本数据类型（整型、浮点型、字符型）、构造数据类型（数组、结构体、共用体、枚举）、指针类型和空类型四大类。

⑶基本数据类型的值是不可分解的。

2、常量

⑴在程序运行过程中，值不能被改变的量称为常量。

⑵常量分类：整型常量、实型常量、字符型常量、字符串常量。

⑶直接常量（字面常量）：由数值或字符本身直接表示的，从字面形式可判别出值和类型。

符号常量：用标识符来表示一个常量，在程序中使用该常量时可以直接引用这个标识符。

⑷符号常量需要先定义，才能使用，定义形式：#define 标识符 常量。

⑸符号常量本质上是一个常量，它的值在其作用域内不能改变，不能再被赋值。程序中修改其值，会“一改全改”。

3、变量

⑴在程序运行过程中，值可以改变的量称为变量。一个变量应该有一个名字，变量的名字用标识符表示。

⑵变量在内存中占据一定的存储单元，在该存储单元中存放数据，这个数据称为变量的值，因为数据有不同的类型，所以变量也分为不同的类型。

⑶变量的三方面含义：变量都有名称，即变量名；变量都要占据一定大小的内存单元（存储单元）；存储变量值的存储单元地址就是变量的地址。

⑷变量的类型必须是 C 语言中有效的数据类型，每一种类型都有相应的类型说明符，如整型为 int，单精度浮点型为 float 等等。

⑸变量的定义：数据类型　变量列表；

变量列表的形式是：变量名 1,变量名 2,…，变量名 n，即逗号分隔的变量名的集合。

①允许在一个变量定义语句中，定义一个或多个相同类型的变量。各变量名之间用逗号分隔。类型说明符与变量名之间至少有一个空格。

②变量定义必须放在变量使用之前，一般放在函数体的开头部分。

⑹变量必须先定义后使用（强制定义规则）。

好处：可以使程序中的变量在使用时不发生错误；将变量指定为某一类型，在编译时就能为其分配存储单元，提高运行时的效率；确定了变量的类型后，实际上就确定了这个变量所能进行的操作。

⑺变量的使用：给变量赋值；读取变量的值。

⑻变量的初始化：在定义变量的同时对变量赋初值，其等价于一个变量定义和一个赋值。

一般形式为：数据类型 变量 1=值 1,变量 2=值 2,…;

变量初始化时，可以给所有变量赋初值，也可以给部分变量赋初值。

4、整型数据

⑴整型数据的存储：数据在计算机内存中都是以二进制形式存放的。但不同的数据类型，其存放格式不同，整型数据在内存中是以补码形式存放的。有符号整数的原码中最左边的一位是符号位，0 表示是正数，1 表示是负数。正数的补码与原码形式相同，负数的补码是将该数的原码除符号位以外其余各位按位取反后再加 1。

⑵整型常量：有十进制、八进制和十六进制三种。

①十进制整数：以非 0 开头的，由数字 0~9 组成的数。

②八进制整数：以 0 开头的，由数字 0~7 组成的数。

③十六进制整数：以 0X 或 0x 开头的，由数字 0~9 和字母 A~F（a~f）组成的数。

⑶整型变量：Visual C++下

类型	类型说明符	字节	取值范围
基本整型	[signed] int	4	-2^31 ~ (2^31-1)
短整型	[signed] short [int]	2	-2^15 ~ (2^15-1)
长整型	[signed] long [int]	4	-2^31 ~ (2^31-1)
无符号基本整型	unsigned [int]	4	0 ~ (2^32-1)
无符号短整型	Unsigned short [int]	2	0 ~ (2^16-1)
无符号长整型	Unsigned long [int]	4	0 ~ (2^32-1)

⑷整型数据的上溢和下溢：两个整数运算结果超出定义类型的取值范围。

⑸整型常数的范围

①系统通常会把程序中出现的整型常量作为 int 型来处理。

②在一个整型常量的后面加上字母 l 或 L，表示它是 long 型常量。

③在一个整型常量的后面加上字母 u 或 U，表示它是无符号整数。

5、实型数据

⑴实型数据的存储：将实数分为数符（表示实数的符号）、尾数、指数三部分分别表示，指数部分是一个有符号整数，由阶符和阶码组成。尾数采用规范化的数据形式，即小数点放在第一个有效数字前面。由于受机器存储位数的限制，实数大多是近似值。

⑵实型常数：只采用十进制。

①小数形式：由正负号、整数部分、小数点和小数部分组成，小数点不可省。

②指数形式：由小数部分、字母 E 或 e、指数部分三部分组成，用来表示一些比较大或比较小的数值。E 或 e 前必须有数字，之后的数字（指数）必须是整数。

⑶实型变量

类型	类型说明符	字节	有效位数	数量级范围
单精度类型	float	4	7	10^-38 ~ 10^38
双精度类型	double	8	15 ~ 16	10^-308 ~ 10^308

在编程时，要注意实型变量的有效位数，避免将一个很大的数和一个很小的数直接相加减，否则就会丢失很小的数。

⑷实型常量的类型

①将程序中的实型常量默认看作是双精度数。

②如果要将实型常量指定为单精度型，在数字后面加字母 f 或 F，注意有效数字只能 7 位。

③对于超出有效位的数字，系统存储时自动舍去。

6、字符型数据

⑴字符数据的存储：一个字符占一个字节，存储的是字符的 ASCII 码。一个汉字占两个字节。ASCII 码的取值范围是 0~255，其中 0~127 为标准 ASCII 码，128~255 称为扩展 ASCII 码，因为 ASCII 码形式上是 0 到 255 间的整数，所以 C 语言中字符型数据和整型数据在一定范围内是通用的。

⑵字符常量：用单引号括起来的一个字符。

转义字符常量：以“\”开头，后跟一个或几个规定的字符，赋予特殊含义。

字符形式	转义字符的意义
\n	回车换行
\t	水平制表（横向跳到下一制表位置）
\a	响铃
\b	退格（将当前位置移到前一列）
\r	回车不换行（将当前位置移到本行开头）
\f	换页（将当前位置移到下页开头）
\\	反斜杠字符“\”
\’	单引号
\”	双引号
\0	空字符，表示字符串结束
\ddd	1~3 位八进制数所代表的字符
\xhh	1~2 位十六进制数所代表的字符（x 不能大写）

⑶字符变量

①字符变量用于存放字符常量，即一个字符变量可存放一个字符，实际存放的是 ASCII 码。

②在一定范围内，字符型数据和整型数据可以通用。一个字符数据既可以以字符形式输出，也可以以整数形式输出。以整数形式输出时，输出的是对应的 ASCII 码。当然，整型数据也可以用整数和字符两种形式输出，以字符形式输出时，系统首先求该数与 256（ASCII 码中的字符个数）的余数，然后用余数作为 ASCII 码，转换成相应的字符输出。

③在一定范围内，字符型数据与整型数据还可以互相赋值，字符数据也可以参加算数运算。

④前面说的字符变量称为有符号字符变量，系统将它们在内存中的值看成是带符号的整数，也就是会将最高位作为符号位。如果不想按有符号处理，可以将字符变量定义为无符号字符变量：unsigned char。

⑷字符串常量

①字符串常量是一对双引号括起来的字符序列。其中字符串两边的双引号是字符串常量的边界符，它不是字符串常量的组成部分。如果字符串常量中本身包含双引号，则要用转义字符表示。

②字符串通常在内存中存储时，除了一个字符占用一个字节外，还自动在其尾部增加一个转义字符'\0'，作为字符串结束的标志。在写字符串时系统会自动加上。

⑸字符常量和字符串常量区别

①字符常量用单引号括起来，字符串常量用双引号括起来。

②字符常量只能是单个字符，字符串常量可以包含多个字符。

③字符串常量占用一个字节的内存空间，字符串常量占用的字节数为字符个数加 1。

④可以把一个字符常量赋给一个字符变量，但不能把一个字符串常量赋给一个字符变量。

⑤C 语言中没有字符串变量，必须使用字符数组来存放字符串变量。

7、运算符

⑴运算符是描述对数据进行特定运算的符号；参与运算的数据，被称为运算对象。

⑵运算符分类：单目运算符：一个运算符连接一个运算对象；双目运算符：一个运算符连接两个运算对象；三目运算符：一个运算符连接三个运算对象。

⑶运算符的优先级：一个表达式中不同运算符在运算中的先后顺序（先高后低）。C 语言中优先级分为 15 级，1 级最高，15 级最低。

优先级	类型	运算符
1		圆括号( )、下标[]、指向->、成员.
2	单目	逻辑非!、按位取反~、自增++、自减--、负-、正+、强制类型转换(类型)、取目标*、取地址&、长度 sizeof
3	算术（双目）	乘*、除/、求余%
4		加+、减-
5	位运算（双目）	左移<<、右移>>
6	关系（双目）	大于>、大于或等于>=、小于<、小于或等于<=
7		等于==、不等于!=
8	按位（双目）	按位与&
9		按位异或^
10		按位或|

11	逻辑（双目）	逻辑与&&
12		逻辑或||
13	条件（三目）	条件?:
14	赋值（双目）	赋值或复合赋值=,+=,-=,*=,/=,%=,<<=,>>=,&=,^=,|=
15	顺序	逗号,

⑷运算符的结合性

①左结合性：当优先级相同时，从左到右进行运算；

②右结合性：当优先级相同时，从右到左进行运算。

③多数运算都是左结合性，只有单目、三目和赋值运算符为右结合性。

⑸任何类型的表达式都有一个确定的运算结果，即表达式的值。对表达式求值时，编译系统一般从左至右对表达式进行扫描，在扫描过程中遇到运算对象时，根据运算符的优先级和结合性来决定先处理哪个运算符。

8、算术运算符和算术表达式

⑴算术运算符：加+、减或负-、乘*、除/、模或求余%

⑵除法说明

①当两个整数（字符）相除时，其结果是整数，如果不能整除，则商只保留整数部分。

②若是两个实数相除或一个实数和一个整数相除，所得的商为实数。

③整数相除不想取整，可将其中一个数或两个数转化为实数后相除。

⑶取余运算：要求两个运算对象都为整数，运算结果是两数相除后的余数，余数的符号与被除数的符号相同。

⑷一般来说，双目运算符的两侧运算对象的类型如果相同，所得结果的类型与运算对象的类型一致。如果类型不同，系统会自动按转换规律进行类型转换，然后再做相应的运算。

⑸用括号可以改变表达式的运算顺序，左右括号必须配对，多层括号都用圆括号（）表示。

⑹算术表达式的运算对象可以是常量、变量和函数调用。sqrt()是求平方根的函数，pow 是乘方的函数，如 sqrt(30)是求 30 的平方根，pow(4.5,3.8)是求 4.5 的 3.8 次方。

9、赋值运算符和赋值表达式

⑴赋值运算符=的作用是将一个数据赋给一个变量。由赋值运算符将一个变量和一个表达式连接起来的式子称为赋值表达式，其一般形式为：变量=表达式。

⑵赋值运算符和数学中的等号不同，其表示右边的值赋给左边，具有方向性。

⑶如果赋值运算符两边的数据类型不一致，赋值时要进行类型转换，转换工作由 C 编译系统自动进行。转换原则是将右边的表达式的类型转换成左边变量的类型，然后再进行赋值。当赋值号右边的类型比左边高时，要降级转换，导致精度降低或运算错误。

⑷复合赋值运算符：+=，-=，*=，/=，%=，<<=，>>=，&=，^=，|=

形式：变量><双目运算符=><表达式>，其等价于<变量>=<变量><双目运算符><表达式>

⑸多级赋值：变量=变量=……=表达式，结合方式从右往左。

10、自增自减运算符

⑴自增运算符++是使变量的值增 1，自减运算符—是使变量的值减 1。

⑵增量运算符可以放在变量之前，称为前置运算；也可以放在变量之后，称为后置运算。

①前置运算：先将变量加 1 或减 1，然后将该变量的新值用于表达式中。

②后置运算：先将变量的值用于表达式，然后再将该变量的值增 1 或减 1。

⑶增量运算符只能用于变量，不能用于常量或表达式。

⑷增量运算符是单目运算符，右结合，优先级高于所有双目运算符。

⑸i+++j，将其理解为(i++)+j，而不是 i+(++j)。

11、不同数据类型间的转换

⑴自动类型转换：运算前，如果两个运算对象类型不同，先将其转换成同一类型，即将“较低类型”转换为“较高类型”（数值不变）后，再进行运算。运算结果是“较高类型”。

⑵强制类型转换：将一个表达式的值转换成所需类型，形式为（数据类型）（表达式），类型说明符和表达式（单个变量除外）都必须加括号。

⑶无论是强制转换还是自动转换，都只是为了本次运算的需要进行的临时性转换，原来变量的类型和值并没有改变。

12、关系运算符和关系表达式

⑴关系运算符：>大于、<小于、>=大于或等于、<=小于或等于、==等于、!=不等于。用关系运算符将两个表达式连接起来的式子，称为关系表达式。

⑵关系表达式的运算结果只有两种：真（成立）或假（不成立）。在 C 语言中，用 1 代表真，用 0 代表假。也就是说，关系表达式的值只有两种可能：0 或 1。

⑶关系表达式的优先级，低于算术运算符，高于赋值运算符。

⑷由于实数存储时有舍入误差，可能导致两个逻辑上应该相等的数不相等，因此应该避免对实数做相等比较。如比较 x 和 y，要避免使用 x==y，而用 fabs(x-y)<1e-6。

⑸关系表达式的值（0 或 1）可以参与其他的运算。

13、逻辑运算符和逻辑表达式

⑴逻辑运算符：&&逻辑与、||逻辑或、!逻辑非。

⑵逻辑运算的运算对象可以是任意类型的量。不管具体值是多少，只要不等于 0，就被视为真；只有当其值为 0 时，才被看作假。

⑶运算规则

a	b	!a	!b	a&&b	a||b
真	真	假	假	真	真
真	假	假	真	假	真
假	真	真	假	假	真

假	假	真	真	假	假

⑷优先级：!最高，&&次之，||最低

⑸不等式的程序表达：错误：1<x<10　正确：x>1 && x<10

⑹逻辑运算的短路特性：在含有逻辑运算符&&或||的表达式中，并不是所有的逻辑运算符都要被执行，只有在必须执行这个逻辑运算符才能求出表达式的值时，才执行该运算符。对&&运算符来说，只有左边的表达式的值为非 0 的情况下，才计算右边表达式的值。而对||运算符来说，只有左边的表达式的值为 0 的情况下，才计算右边表达式的值。

14、条件运算符和条件表达式

⑴条件运算符是唯一的三目运算符，要求有三个运算量。

⑵一般形式：表达式 1?表达式 2：表达式 3；

求值方式：条件表达式在执行时，首先计算表达式 1，如果它的值为真（非 0），则计算表达式 2，并以表达式 2 的值作为整个条件表达式的值；若表达式 1 的值为假（0），则计算表达式 3，并以表达式 3 的值作为整个条件表达式的值。

⑶优先级仅高于赋值运算符和逗号运算符，右结合。

⑷条件运算符?:是一个运算符，不能分开单独使用。

⑸条件运算符通常用于赋值语句中，即变量名=表达式 1?表达式 2：表达式 3。

15、逗号运算符和逗号表达式

⑴逗号运算符功能是把两个或多个表达式连起来组成一个表达式。

⑵一般形式：表达式 1,表达式 2,…，表达式 n

求值方式：从左到右顺次求解每个表达式的值，并以最后一个表达式的值作为整个逗号表达式的值。

⑶逗号表达式的优先级最低，左结合。

⑷程序中使用逗号表达式，通常是想分别计算各个表达式。

⑸逗号运算主要用于解决只能出现一个表达式的地方却要出现多个表达式的问题，如 for 语句。

⑹并不是所有出现逗号的地方都组成逗号表达式。

第 4 章　顺序结构程序设计

1、C 程序的组成结构

一个 C 程序可以由多个源程序文件（分别编译的文件模块）组成，而一个源文件可以由若干个函数和预处理命令以及全局变量定义部分组成。

2、C 语句种类

表达式语句：由一个表达式加分号组成，即表达式;

函数调用语句：由函数名、括号、实参、分号组成，即函数名(实际参数表);

控制语句：条件判断语句、循环语句、辅助控制语句;

空语句：只有 1 个分号的语句

复合语句

3、数据的输入输出

⑴输出：从计算机向外部输出设备（显示屏、打印机、磁盘等）输出数据。

输入：从外部设备（键盘、磁盘、光盘、扫描仪等）向计算机输入数据。

⑵不把输入输出作为 C 语言提供的语句的原因：使 C 语言编译系统简单，因为将语句翻译成二进制的指令是在编译阶段完成的，没有输入输出语句就可以避免在编译阶段处理时出现与硬件有关的问题，可以使编译系统简化，而且通用性强，可移植性好，对各种型号的计算机都适用。

⑶标准输入输出函数：putchar()、getchar()、printf()、scanf()、puts()、gets()。调用时需先引人 stdio.h 的头文件。

4、格式化输出函数——printf()

⑴printf()用于输出各种类型数据，称为格式化输出函数。它的作用是向终端（或系统隐含指定的输出设备）输出若干个任意类型的数据。

⑵调用格式：printf(“格式控制符”,输出列表);

①格式控制：用双引号括起来的字符串，包括三部分信息：

格式说明：总是由%和一个规定字符组成，将输出的数据以指定的格式输出。

转义字符：是由\和特殊字符组成的，在输出数据时控制光标的位置。

普通字符：即需要原样输出的字符。

②输出列表：需要输出的数据，可以是常量、变量或表达式，逗号隔开，其个数应与格式说明个数一样多，且顺序一一对应。

⑶如果输出参数的个数多于格式说明，将按格式字符串中的个数输出参数；如果输出参数的个数少于格式字符串所说明的个数，也会按格式字符串中的个数输出参数，但没有对应输出参数的格式字符串，输出内容不确定。

⑷格式说明的一般形式：%[修正符][输出宽度 m][.精度 n][l/h]格式字符，其作用是指定输出时的数据转换形式。

⑸格式字符

符号	说明
%d	用于十进制有符号整数的输出
%u	用于十进制无符号整数的输出
%f	用于小数形式的浮点数的输出
%s	用于字符串（字符序列）的输出
%c	用于单个字符的输出
%p	用于指针的值的输出

%e,%E	用于指数形式的浮点数的输出
%x,%X	用于以十六进制表示的无符号整数的输出
%o	用于以八进制表示的无符号整数的输出
%g,%G	按照数值的特点，自动选择合适的浮点数形式输出

当需要输出字符%时，需要连用两个%；输出\时，需要连用两个\。

⑹输出宽度 m 和精度 n

①对整型数据，m 包括了符号和其整数本身的位数，n 起不到作用。若给出的 m 大于实际数据的位数，则可用空格补足 m 位。如果 m 小于实际数据位数以及默认时，都原样输出数据。

②对字符数据，m 表示数据的总位宽，在%c 格式中，使用 n 不起作用；在使用%s 格式时，n 表示要输出的字符个数，实际字符多于 n 时，将被截断输出；若 n 为默认时，则按 m 的要求处理整个序列。

③对浮点数来说，m 包括了符号、整数、小数点和小数，n 只是小数数据的位宽，通常 m 大于 n。如果给出 n 小于实际数据的小数位数，则四舍五入截去超过的部分；若 n 大于实际数据的小数位数，则末尾补 0 满足 n 位。如用%f，不指定输出位宽时，整数部分原样输出，小数部分输出六位。

⑺修正符：在%和格式字符间插入不同的修正符，则输出的形式有所不同。

修正符	意义
0	输出结果不足位宽时，高位补 0，默认时补空
+	输出结果包括+及-符号，默认时，正号不输出，负号输出
-	输出结果左对齐，默认为右对齐
L 或 l	在 Visual C++中，双精度型数据的输入要用此修正
H 或 h	用于短整型数据的格式控制
*	表示输入项在读入后不赋给相应的变量

5、格式化输入函数——scanf()

⑴scanf()用于从键盘上输入各种类型数据，称为格式化输入函数。

⑵调用函数 scanf()的格式：scanf(“格式控制”,地址列表);

①功能：按指定的格式，从标准输入设备（键盘）输入数据，分别存放到地址列表中的各个变量的存储空间中。

②函数 scanf()是从缓冲区中按指定的格式读取数据的，如果缓冲区为空，则等待用户从键盘输入数据，一旦用户从键盘上输入了回车，则从缓冲区中按照排队的先后顺序读取数据，存放在相应的变量存储空间。

③格式控制的含义同 printf 类似，地址列表是由若干个地址组成的列表，可以是变量的地址，或字符串的首地址。各个地址之间用逗号隔开。

⑶用%d 读取一个 int 型的整数

①缓冲区有数据。如果排在队头的数据是空格或者回车符，则取走丢弃；如果遇到整数，则取走存放到对应的 int 型变量空间；如果遇到其他类型的数据，读取失败。

②缓冲区无数据。等待用户从键盘读取数据，回车后再按有数据处理。

⑷用%c 读取一个字符

①缓冲区有数据。取走排在队头的第一个字符，存放到对应的 char 型变量空间。

②缓冲区无数据。等待用户从键盘输入数据，回车后按有数据处理。

⑸其他说明

①用格式控制符读取数据时，一定要分析输入相应的数据之后，在缓冲区中剩余的数据是否对后续的读取数据有影响，必要时，需要用读数据的函数将产生影响的数据取走，以保后续读数据的函数读取的数据是正确的。

②多个数据输入时，可用空格、Tab 键、回车键分隔，不因分开书写语句影响数据分隔。多个字符输入时，连续输入，不因分开书写语句影响字符连续输入。

遇不同数据类型数据时，自动分隔数据。

③输入数值数据时，可以规定位宽，用以指定输入数据所占的总位宽 m，系统自动按它截取所需数据。在格式控制符%f 不能规定精度，即修正符中的 n 不起作用。

④如果格式控制中含有除格式说明以外的字符，运行时必须原样输入。

⑤使用%*附加说明符，表示跳过它对应的数据。

⑥Visual C++中双精度数据输入时，格式输入函数中必须使用%lf 格式符。

⑦当输入的数据超过该语言和环境所允许的数据范围时，系统接收的数据也会不同。

6、字符输出函数——putchar()

⑴调用形式：putchar(ch);

⑵功能：输出 ch 对应的字符。Ch 可以是字符型变量、整型变量、字符常量表达式或控制字符。其作用等同于 printf("%c",ch);该函数的返回值是其参数 ch 的 ASCII 码值。

⑶printf()与 putchar()使用区别

①前者的输出参数可以是任何类型数据，而后者的输出参数是字符类型（如果是其他类型的数据不能保证正确显示），可以理解它是一种字符显示的函数。

②输出单个字符时，printf 函数使用双引号，如 printf("\n");，而 putchar 用的是单引号，如 putchar('\n');

③printf 函数需要设定输出格式，而 putchar 函数无格式控制。

7、字符输入函数——getchar()

⑴函数 getchar()没有参数，称为无参函数。调用形式：getchar()。

⑵功能：从标准输入设备（键盘）输入一个字符，并将读到的字符作为函数值返回。

⑶说明：函数 getchar()是从缓冲区中读取字符的。如果缓冲区不为空，则从排队等待的数据中读取第一个字符；如果是空，则等待用户从键盘输入数据，回车后再从缓冲区读取排在队头的第一个字符，并显示在屏幕上。

8、输入输出缓冲流的概念

由于计算机系统中各种设备运行速度不尽相同，为了匹配快速设备和慢速设备间的通信步伐，计算机中大量使用硬件缓冲区，用以弥补不同硬件之间运行速度的差距。

第 5 章　选择结构程序设计

1、if 语句的 3 种形式

⑴单分支 if 语句

①格式：if(表达式) 内嵌语句

②语义：先计算表达式的值，如果表达式的值为非 0（真）时，则执行其后的内嵌语句，然后执行 if 语句的后续语句；否则，即当表达式值为 0（假）时，不执行该内嵌语句，直接执行 if 语句的后续语句。

③说明：在 if 语句中的“内嵌语句”是在满足条件时，所执行的操作语句，可以是一条语句，以分号结束；也可以是由多个语句用｛｝括起来构成的复合语句。在单分支 if 语句中，给出了条件满足时执行的语句，当条件不满足时，单分支 if 语句本身不作处理。

⑵双分支 if 语句

①格式：if(表达式) 内嵌语句 1　else　内嵌语句 2

②语义：先计算表达式的值，如果表达式的值为非 0（真）时，则执行内嵌语句 1，否则执行内嵌语句 2。执行内嵌语句后，都将执行 if 语句的后续语句。

③说明：else 隐含的是与 if 之后的表达式相反的条件。在双分支 if 语句中，既给出了条件满足时执行的语句，也给出了当条件不满足时执行的语句。

⑶多分支 if 语句

①格式：if(表达式 1) 内嵌语句 1

else if(表达式 2) 内嵌语句 2 … else if(表达式 n) 内嵌语句 n

else 内嵌语句 n+1

②语义：依次判断表达式的值，当某个值为非 0（真）时，则执行其后对应的内嵌语句。然后跳到整个 if 语句的后续语句继续执行。如果所有的表达式值均为 0（假），则执行内嵌语句 n+1。然后继续执行后续语句。

③说明：if 语句中 else 隐含的条件，是否定在它之上所有 if 后面的表达式条件。

④多分支 if 语句与多个单分支语句是不同的，前者的效率高于后者。

2、if 语句的注意点

⑴三种形式的 if 语句在 if 后面都有表达式，可以为任意表达式。一般为逻辑表达式或关系表达式，也可以是其他表达式，如赋值表达式，甚至也可以是一个常量、变量，但是必须用括号括起来。

if(a)：只要 a 的值非 0，条件是真，否则是假。

if(!a)：只有 a 的值为 0，条件为真，其余均为假。

if(a=b)：把 b 的值赋给 a，如果 a 为非 0，则为真，否则为假。

if(a==b)：如果 a、b 相等，条件为真，否则为假。

⑵无论是哪种形式的 if 语句，系统将其视为一个整体，即一条语句。if 语句中的 else 子句不能作为语句单独使用，它必须是 if 语句的一部分，与 if 配对使用。

⑶if 和 else 后面的内嵌语句，可以是任何语句。如果是｛｝括起来的复合语句，后面不需要加分号。

⑷建议：只要是内嵌语句，不管是几条语句，均用｛｝括起来。

⑸多分支 if 语句每次只能执行其中一个分支。

3、if 语句的嵌套

⑴if 语句的内嵌语句可以是另一个 if 语句，这就是 if 语句的嵌套。

⑵规定 else 总是与它前面最近的尚未与 if 配对的，没有被花括号隔开的 if 配对。

4、switch 语句的形式

⑴switch 语句处理多分支选择。形式为：

```
switch(表达式)
{ case  常量表达式 1：语句组 1; [break;]
  case  常量表达式 2：语句组 2;[break;]
  …
  case  常量表达式 n：语句组 n;[break;]
  default  语句组 n+1;}
```

⑵语义：先计算表达式的值，然后逐个与其后的常量表达式值相比较，当表达式的值与某个 case 之后的常量表达式的值相同时，即执行其后的语句组，之后顺序执行下面的各个分支的语句组，直至遇到 break 语句或者执行到右花括弧。如表达式的值与所有 case 后的常量表达式均不相同时，则执行 default 后面的语句。

5、switch 语句使用说明

⑴switch 后面括号里的表达式，ANSI 标准允许它为任何类型。Visual C++要求表达式的值必须是整型。

⑵每一个 case 的常量表达式的值可以是整型常量或字符常量，必须互不相同，否则就会出现互相矛盾的现象。

⑶如果 case 后面的语句组中，没有 break 语句，执行完成后，流程控制转移到它的下一个 case 继续执行。”case 常量表达式”只是起语句标号作用，并不是在该处进行条件判断。在执行 switch 语句时，根据 switch 后面表达式的值找到匹配的入口标号，就从此标号开始执行下去，不再进行判断。即 switch 语句一次可能执行多个分支。为了避免这种情况，可以使用 break 语句，跳出 switch 语句。

⑷各个 case 和 default 的出现次序不影响执行结果。

⑸多个 case 可以公用一组执行语句。如 switch(ch){case’A’: case’B’:　语句}

⑹在 case 后面，当包含一个以上的执行语句时，可以不用花括号括起来。

⑺default 子句是可选的，可以省略不用。

⑻switch 语句同 if 语句一样也允许嵌套。

⑼switch 语句的使用要注意表达式的值与 case 中常量值的对应。使用 switch 的困难在于表达式的构造，必须使表达式的值涵盖所有的 case 要求的常量值。

第 6 章　循环结构程序设计

1、while 语句

⑴while 语句用来实现“当型”循环结构，形式：while(表达式) 内嵌语句。

⑵执行过程：先计算出表达式的值，当表达式的值为 0 时，则不执行循环体，退出循环结构，转到循环结构的后续语句执行；当表达式的值为非 0 时，则执行循环体，然后再次计算表达式的值，重复上述过程，直到计算的表达式值为 0，退出循环结构。

⑶特点：先判断表达式，后执行循环体。

⑷使用说明：

①内嵌语句是构成 while 语句的一个部分，是被重复执行的语句。如果内嵌语句不止一句，需用复合语句。通常内嵌语句又称为循环体。

②while 语句中的表达式可以是任意类型的表达式。

③在循环结构中，一定要有使表达式的值趋于 0 的语句，否则循环将一直执行下去，即出现“死循环”。

④当控制循环的表达式等于 0 或不等于 0 时，经常使用省略形式：while(x!=0)等价于 while(x)，while(x==0)等价于 while(!x)。

⑤循环体语句要注意顺序，否则可能影响结果。

⑥在循环结构中，控制循环的变量，其步长的值可正可负，如果控制循环的变量是递增变化的，称为“正向”循环。反之称“逆向”循环。

2、do while 语句

⑴形式：do 内嵌语句 while(表达式);

内嵌语句为循环体，循环条件之后，一定要加分号。

⑵执行过程：先执行循环体，然后判断表达式，当表达式的值为非 0（真）时，重复执行循环体，直到表达式的值等于 0（假），循环结束。

⑶使用说明：

①do while 语句的控制循环是否执行的条件在内嵌语句即循环体之后，因此内嵌语句（循环体）至少会执行一次。

②do while 循环并不对应着直到型循环，因为这里的结束条件是 while 后面的表达式给出的，当这个表达式的值为 0 时，将结束循环。而直到型循环是一种 do until，它的结束条件刚好相反，是在为非 0 时结束的。

③同 while 语句一样在循环体语句中应有使循环趋向于结束的语句。

④在一般情况下，用 while 语言和用 do while 语句处理同义问题时，若二者的循环体部分是一样的，它们的结果也一样。但是如果 while 后面的表达式一开始就为假（值为 0）时，循环体一次也不执行；而 do while 语句至少执行一次循环体。

3、for 语句

⑴for 语句使用场景更多，灵活方便，该语句不仅可以用于循环次数已经确定的情况，而且还可以用于循环次数不确定，只给出由循环结束条件的情况，它完全可以替代 while 语句和 do while 语句。

⑵一般形式：for(表达式 1;表达式 2;表达式 3) 内嵌语句

⑶执行过程：

①先求解表达式 1；

②求解表达式 2，若其值为非 0（真），则执行循环体，然后执行③。若值为 0（假），则结束循环，转到⑤；

③求解表达式 3；

④转回到第②步继续执行；

⑤循环结束，执行 for 语句的后续语句。

⑷最常用形式：for(循环变量赋初值;循环终值;循环变量增量) 循环体

循环变量增量也称为步长。对于循环的处理，选择好循环的初始值、终值以及步长。

⑸使用说明：

①表达式 1 只执行一次。

②for 语句的一般形式中的表达式 1 可以省略，此时应在 for 语句之前给循环变量赋初值。注意省略表达式 1 时，其后的分号不能省略。

③如果表达式 2 省略，即不判断循环条件，循环无终止地进行下去，此种情形要应用其他跳转语句保证循环正常结束。

④表达式 3 也可以省略，此时缺少循环控制变量的增值变化，但可以用其他方法或将表达式 3 放在原循环体语句之后，保证循环正常结束。

⑤可以省略表达式 1 和表达式 3，只有表达式 2，即只给出循环条件，完全等价于 while。

⑥3 个表达式都可省略，相当于 while(1)。

⑦表达式 1 和表达式 3 用逗号表达式可以同时完成多个操作。

⑧表达式 2 一般是关系表达式或逻辑表达式，但也可以是其他表达式，只要其值为非 0，就执行循环体。

4、三种循环语句的比较

⑴三种循环语句都可以用来处理同一问题，一般情况下它们可以互相代替，但为了保证循环体正常运行，应该特别注意三个方面的处理：控制循环的初始状态（初始值）、循环控制的条件、循环体内部对控制条件的改变（增量变化）。这三个方面相互配合，相互影响，

共同完成循环控制。

⑵while 语句和 do while 语句中的控制循环的条件是 while 之后的表达式，for 语句中的控制循环的条件是表达式 2。

⑶控制循环的某些变量初值，对 while 和 do while 语句，必须在进入循环之前完成，对 for 语句可以在 for 语句之前完成，也可以在表达式 1 中完成。

⑷for 和 while 是先判断控制循环的条件后执行循环体，而 do while 是先执行循环体后判断控制循环的条件。

⑸while 语句多用于循环次数不定的情况，在不满足条件时不执行；for 语句多用于循环次数固定的情况；do while 语句多用于先执行循环体后判断条件。

⑹while 和 do while 语句能完成的操作，for 语句均能完成，功能更强大，形式更灵活。

5、循环的嵌套

⑴一个循环体内又包含另一个完整的循环语句，称为循环的嵌套。内嵌的循环中还可以嵌套循环，这就是多层循环。三种循环语句可以互相嵌套。

⑵使用说明：

①多层循环执行的过程：外层循环每执行一次，内层循环要完整地执行一遍。

②注意嵌套中循环体语句的构成。

③不允许出现交叉的循环结构。

④注意嵌套循环的控制变量不得混用。

6、循环的异常跳转

⑴break 语句

①形式：break; 功能：跳出当前的循环或 switch 结构。

②使用场景：通常用在循环体语句和 switch 语句中。当 break 用于 switch 语句中时，可使执行流程跳出 switch 语句，执行 switch 语句的后续语句；当 break 语句用于 do while、for、while 循环语句时，可使执行流程终止循环，执行循环结构后面的语句。通常 break 语句总是与 if 语句连在一起，放在循环体上，即满足条件时便跳出循环。

③break 语句不能用于循环语句和 switch 语句之外的任何其他语句。

④break 语句在循环体中，一般与 if 语句配合使用。

⑤break 在 3 种循环语句中，流程的转向是一样的。

⑥在多层循环中，一个 break 语句只向跳一层。用 break 跳转到最外层需设置多次。

⑵continue 语句

①形式：continue; 功能：跳过循环体中剩余的语句而强行执行下一次循环。

②使用场景：只用在 for、while、do while 等循环体中，常与 if 一起使用。Continue 在 3 中循环体中，流程的转向是不一样的。

③continue 与 break 的区别是：continue 语句只结束本次循环，继续进行下一次循环，而不是终止整个循环的执行。而 break 语句则是结束整个循环的执行。

7、循环结构及常用算法应用

⑴递推法

①顺推法：找到递推关系式，然后从已知的初始条件出发，一步步地按递推关系式递推，直至求出最终结果，往往使用正向循环。

例：求 n!；求 Fibnacci 数列等。

②逆推法：在不知道初始条件的情况下，经某种递推关系，从结果出发，利用已知条件倒着一步一步推理、分析，直到解决问题，往往采用逆向循环处理。

例：猴子吃桃子问题等。

⑵穷举法

又称枚举法，基本思想是，在问题要求的范围内，对所有可能的情况逐一测试，直到全部情况测试完毕。测试结构有两种：若某个情况符合问题要求，则为本题的一个答案；若测试完所有情况均不符合问题要求，则问题无解。

在穷举法中，每次的测试操作是循环体的核心。循环控制可以是确定的循环次数，也可以是对某一目标逐次测试：完成测试次数或达到某一目标后循环结束。

例：分硬币问题。

⑶迭代法

又称辗转法，是一种不断用新值取代变量的旧值，再用变量的旧值递推新值的过程。

例：求最大公约数（相减法、定义法、辗转相除法）等。

8、随机数的产生和应用

⑴背景介绍

①随机数就是在一定范围内随机产生的数，并且使得到这个范围内的每一个数的机会是一样的。随机数的特性是所产生的后面的那个数与前面的那个数毫无关系。

②不管用什么方法产生随机数，都必须给它提供一个名为“种子”的初始值。所有的随机数是在这个“种子”下产生的。种子的产生方法较多，最常用的是利用系统时钟产生。种子的个数可以用循环来控制。

⑵产生 0~32767 之间的随机整型数

①C 语言中提供了基于 ANSI 标准的伪随机数发生器函数 rand()和 srand()，用来生成随机数。

②工作过程：给 srand()提供一个随机数的种子；调用 rand()，它会根据提供给 srand()的种子值返回一个随机数（0~32767）；根据需要多次调用 rand()，从而不间断地得到新的随机数；无论何时，都可以给 srand()提供一个新的种子，从而进一步对 rand()的输出结果随机化。

③要调用库函数产生随机数的函数，需要在程序中加入库函数头文件#include <stdio.h>

④rand()是一个无参函数。其功能是产生一个 0~32767 的随机数整数。

⑤RAND_MAX 是一个系统预定义好的常量，它表示最大随机数整数。

⑥如果每一次执行程序要产生不同的随机数序列，则在调用函数 rand()之前先调用下面的

函数：srand((unsigned int) time(NULL))或 srand(time(0));其中，函数 time()的头文件是 #include <time.h>。该处理方式可以使计算机读取当前的时钟值，并把该值自动设置为随机数的种子。

⑦在调用 rand()函数产生随机数前，必须先利用 srand()设好随机数种子，如果未设随机数种子，rand()在调用时会自动设随机数种子为 1。若去掉该语句，则产生的若干个随机数每次都是一样的。

⑧程序实例：产生 10 个随机整数。

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void main()
{   int k = 0;
    srand((unsigned int) time(NULL));                    //设置随机数的种子
    for(k=0;k<10;k++)                                    //循环输出随机数
       printf("%d\t",rand());
    printf("\nmax rand_number is %d\n",RAND_MAX);}     //输出最大随机数
```

⑶其他形式产生的随机数

①产生一个 0 到整数 m-1 之间的随机数：rand()%m，m 称为比例因子。

如：随机产生 1~6 的整数代表掷 6 面的骰子的一种情况：1+rand()%6

②产生一个大于 0 且小于 1 的随机小数：(float)rand()/RAND_MAX。

注意：要将 rand()强制转换为 float 型之后，再与 RAND_MAX 作除法运算，否则由于整数除法的恒取整，产生结果永远为 0。