1. 数组是一组有序数据的集合。数组中各数据的排列是有一定规律的，下标代表数据在数组中的序号。
2. 用一个数组名和下标来唯一地确定数组中的元素。
3. 数组中的每一个元素都属于同一个数据类型。不能把不同类型的数据放在同一个数组中。
4. 由于计算机键盘只能输入有限的单个字符而无法表示上下标，c语言就规定用方括号中的数字来表示下标，

定义一维数组：类型符 数组名[常量表达式]

1．在定义数组时，需要指定数组中元素的个数，方括号中的常量表达式用来表示元素的个数，即数组长度。例：指定a[3]，表示a数组有3个元素。注意下标是从0开始的，这三个元素是：a[0]，a[1]，a[2]。

2．常量表达式中可以包括常量和符号常量，如“int a[3+5];”是合法的。

3．常量表达式中不能包含变量，也就是说，C语言不允许对数组的大小作动态定义，即数组的大小不依赖于程序运行过程中变量的值。

如果在被调用的函数（不包括主函数）中定义数组，其长度可以是变量或非常量表达式。

Void func( int n )

{ //合法，n的值从实参传来

Int a[2\*n];

}

在调用func函数时，形参n从实参得到值。这种情况称为“可变长数组”，允许在每次调用func函数时，n有不同的值。但是在执行函数时，n的值是不变的，数组长度是固定的。

如果指定数组为静态（static）存储方式，则不能用“可变长数组”。

例： static int a[a\*n]; //不合法，a数组指定为static存储方式

在引用一维数组元素时，只能引用数组元素而不能一次整体调用整个数组全部元素的值。

数组名[下标]：可以是整型变量或整型表达式。

例：a[0]=a[5]+ a[7] +a[2\*3]

定义数组时用到的“数组名[常量表达式]”和引用数组元素时用的“数组名[下标]”形式相同，但含义不同。

例：int a[10]; // a[10]表示的是定义数组时指定数组包含10个元素

T=a[10]; // a[10]表示引用a数组中序号为10的元素

数组的初始化：为了使程序简洁，常在定义数组的同时，给各数组元素赋值。

1. 可以只给数组中的一部分元素赋值。

例：int a[10]={0,1,2,3,4};

定义a数组有10个元素，但花括号内只提供5个初值，这表示只给前面5个元素赋初值，系统自动给后5个元素赋初值为0。

即int a[10]={0,1,2,3,4,5,0,0,0,0,0};

1. 未赋值的部分元素自动设为0。
2. 在对全部数组元素赋初值时，由于数据的个数已经确定，因此可以不指定数组长度。

Int a[3]={1,2,3}; 等于 int a[]={1,2,3};

说明：如果在定义数值型数组时，指定了数组的长度并对之初始化，凡未被“初始化列表”指定初始化的数组元素，系统会自动把它们初始化为0（如果是字符型数组，则初始化为’\0’，如果是指针型数组，则初始化为null，即空指针）。

二维数组定义的一般形式：

类型说明符 数组名 [常量表达式] [常量表达式];

用矩阵形式（如三行四列形式）表示二维数组，是逻辑上的概念，能形象地表示出行列关系。而在内存中，各元素是连续存放的，而不是二维的，是线性的。

二维数组元素的表示形式：数组名 [下标] [下标]

例如：a[2] [3]表示a数组中序号为2的行中序号为3的列的元素，下标应是整型表达式，如a[2-1] [2\*2-1]。

二维数组的初始化：

1.分行给二维数组赋初值。

例：int a[3] [4]={{1,2,3,4}，{5,6,7,8}，{9,10,11,12}} ;

把第一个花括号内的数据给第一行的元素，第n个花括号内的数据赋给第n行的元素……即按行赋初值。

2.可以对部分元素赋初值。

例：int a[3] [4}={{1}，{5}，{9}} ;

只对各行第一列（即序号为0的列）的元素赋初值，其余元素自动为0。

3.如果对全部元素都赋初值，则定义数组时对第一维的长度可以不指定，但第二维的长度不能省。

例int a[2] [3]={1,2,3,4,5,6,} ;

等价于：int a[ ] [3]={1,2,3,4,5,6} ;

C语言中没有字符串类型，字符串是存放在字符型数组中的。由于字符型数据是以整数形式（ASCII代码）存放的，因此也可以用整型数组来存放字符数据，这合法，但浪费存储空间。

字符数组的初始化：如果花括号中提供的初值个数（即字符个数）大于数组长度，则出现语法错误；如果初值个数小于数组长度，则只将这些字符赋给数组中前面那些元素，其余的元素自动定为空字符（’\0’） 。

C语言规定了以字符’\0’作为字符串结束标志，是一个ASCII码为0的字符，不是一个可以显示的字符，而是一个“空操作符”，即什么也不做。

char c[10];

scanf("%s\n",c);

注意：scanf函数中的输入项如果是字符数组名，不要再加地址符&，因为在c语言中数组名代表该数组的起始地址。

**使用字符串处理函数应在文件开头加“#include <string.h>”**

Puts函数：输出字符串的函数 形式：puts（字符数组）

gets函数：输入字符串的函数 形式：gets（字符数组）

注意：用puts和gets函数只能输出或输入一个字符串。

Strcat函数：字符串连接函数

形式：strcat(字符数组1，字符数组2)

例：char str1[30]={“people’s republic of ”};

Char str2[ ]={“china”};

Printf(“%s”, strcat(str1,str2));

输出：people’s republic of china

说明：字符数组1必须足够大，以便容纳连接后的新字符串。

Strcpy(字符数组1，字符串2)

作用：表示“字符串复制函数”，能将字符串2复制到字符数组1中。

例： char str1[10],str2[ ]=”china” ;

Strcpy(str1,str2); // 与strcpy(str1,”china”);相同

说明：1.”字符数组1”必须写成数组名形式（如str1），“字符串2”可以是字符数组名，也可以是一个字符串常量。

2.如果在复制前未对str1数组初始化或赋值，则str1各字节中的内容是无法预知的。

3.不能用赋值语句将一个字符串常量或字符数组直接给一个数组。

只能用strcpy函数将一个字符串复制到另一个字符数组中，而赋值语句只能将一个字符赋给一个字符型变量或字符型数组元素。

Strncpy函数：将字符串2中前面n个字符复制到字符数组1中,取代其原有的最前面n个字符。

例：strncpy(str1,str2,2);

Strcmp函数：字符串比较函数

形式：strcmp(字符串1，字符串2) 作用：比较字符串1，2。

说明：1.将两个字符串自左到右逐个字符相比（按ASCII码值大小相比），直到出现不同的字符或遇到’\0’为止。

2.若出现不相同的字符，则以第一对不相同的字符的比较结果为准。

3.英文字母的比较，位置在后面的为大，小写字母大于大写字母。

比较的结果由函数值带回：

1. 如果字符串1=字符串2，则函数值为0。
2. 如果字符串1>字符串2，则函数值为一个正整数。
3. 如果字符串1<字符串2，则函数值为一个负整数。

对两个字符串比较，不能用以下形式：

If(str1>str2)

Printf(“yes”);

而只能用

If (strcmp(str1,str2)>0)

Printf(“yes”);

Strlen函数：测字符串长度的函数

形式：strlen(字符数组)

Strlwr函数：转换小写的函数

形式：strlwr(字符串)

Strupr函数：转换大写的函数

形式：strupr(字符串)