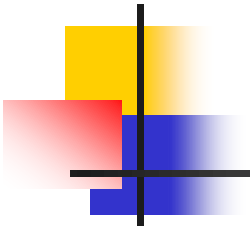
第7讲 数组（Part II）

周水庚

2016年10月27日



提要

- 数组的基本概念
- 一维数组
- 多维数组
- 字符数组与字符串

数组

- 定义形式

  类型说明符　数组名[常量表达式];

  类型说明符　数组名[常量表达式][常量表达式];

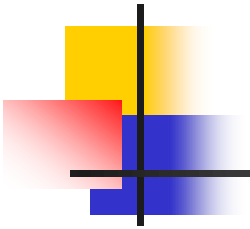
  …

- 引用形式

  数组名[下标]

  数组名[下标][下标]

  …



提要

- 数组的基本概念
- 一维数组
- 多维数组
- **字符数组与字符串**

字符数组

- 字符数组定义形式与其他数组定义形式一样

  char 字符数组名[元素个数];

  - 如： char s[5];
  - 表示数组s有5个元素，每个元素能存放一个字符，整个数组最多可存放5个字符
- 当存于字符数组中的字符列最后有ASCII码值为0（记为‘\0’）时，称该数组中的字符列为字符串。并称**字符‘\0’为字符串结束标志符**

字符数组初始化

- 字符数组除可以与普通数组一样初始化外，也可利用字符串常量给字符数组初始化。用字符串常量对字符数组初始化时，系统会在字符列未尾添加一个字符串结束符
  - 如：char a_str[] = {"I am happy!"};
  - 或简写为 char a_str[] = "I am happy!";
  - 数组a_str[]有12个元素，其中a_str[11]的值是字符串的结束标志符’\0’
  - 如：char str_list[][30]={"I am happy!",
    "I am learning c language."};
  - 字符数组str_list[0]和字符数组str_list[1]各可存储30个字符，现分别存储有11个有效字符的字符串和有25个有效字符的字符串

字符串

- 我们称最后有字符串结束符的字符序列为字符串
  - 字符数组中存储的字符序列本身并不要求最后要有字符'\0'。但当字符数组内存储的内容需要作为字符串时，就必须要有标志符 '\0'
  - 当字符数组内存储的是字符串时，可用"%s"格式输出，若是普通的字符序列，则它不能用格式"%s"，而只能用格式"%c"
    - char s1[]= "student";
    - char s2[]={'s','t','u','d','e','n','t'};
    - 则 printf("%s", s1); 是正确的
    - 而 printf("%s", s2); 是错误的
    - 实际上字符数组s1有8个元素;s2只有7个元素

字符串与字符数组

- 文字信息可作为字符串来处理，而字符串又以字符数组的形式来组织存储
- 字符数组需预先指定长度以保证能存放足够长的文字信息
  - 但限定长度对使用是不方便的。例如定义的字符数组：char str[120]可存储120个字符，如用它来接受输入或组织输出，每次必须键入120个字符，或总得输出120个字符。显然是不合理的，因为通常输入输出的字符个数均小于120
  - 为能使存于字符数组中的字符串的实际长度可长可短，且其长度可随时测定，两个长度不相等的字符串能按字典顺序比较，C语言为字符串规定了一个字符串结束标志符‘\0’

字符串常量

- **字符串常量**的书写形式为：**"字符序列"**
  - 其中字符序列可由零个或多个字符组成，如字符串常量“I am a student.”含15个有效字符
  - 字符串常量"" 不含任何有效字符，其长度为0，习惯称为**空字符串**
  - 在字符串常量的书写形式中，双引号""只充当字符串的界限符，不是字符串的一部分
  - 如果字符串要包含字符‘"’，则可经过转义序列 (如\")来实现，其它转义序列 (如\n，\t)也可以作为单个字符出现在字符串常量中。如 "\tThis is a string.\n"

字符串常量串接规则

- 通常字符串写在一行内。如果一个字符串常量在一行内写不下时，可用字符串常量的串接规则把字符串分成连续多行形式书写
- 字符串常量串接规则有两条
  - 在键入字符‘\’之后紧接键入回车键。如

    “I am a st\(回车换行符)
    ring.”
    - 就是字符串常量 ″I am a string.“
  - 连续两个紧接的字符串常量相当于一个字符串常量。如

    ″I am ″″a string.″
    - 也是字符串常量 ″I am a string.″

字符串输入输出

- 字符串的输入输出可以有两种方式
  - 用格式“%c”，结合循环结构逐个字符输入或输出
  - 用格式“%s”，将字符串整体地输入或输出
  - 例如： char s[] = "C language";
  - 用第一种方法输出

```
for(i = 0; s[i]; ++i)
        printf("%c", s[i]);
```

  - 用第二种方法输出

```
printf("%s", s);
```

字符串注意点-1

- 字符串与存储字符串的字符数组有区别
  - 字符串的有效字符是指从所指位置的第一个字符开始至字符串结束标志符之前的那些字符。如：

```
char str[50] = "Pas\0cal Cobol Fortran C";
printf("%s\n", str);
```

  - 将只输出:Pas
  - 而实际上，数组**str[]**在字符串结束符之后还存有其它许多字符

字符串注意点-2&3

- 用“%s”格式输出字符串时，不包括字符串结束标志符
  - 字符数组名可作为字符串名
- 在调用scanf()为字符数组输入字符串时，输入项是数组名，不要加地址运算符 &
  - 例如：scanf("%s", str);
  - 用 “%s” 格式输入字符串时，掠过前导的空白类字符，只能输入一串不含空白类字符的字符序列，遇空白类字符，或输入了格式指定的字符个数，就结束输入
  - 若要输入一串包括空格符在内的一行字符，要用字符行输入函数gets()

字符串注意点-4

- 若用”%c”格式结合循环输入字符序列，若程序又想将输入的字符序列构成字符串，则程序必须用赋值语句在字符列之后存入字符串结束标志符。如：

```
char s[20]; int k;
for(k = 0; k < 5; k++)
      scanf("%c", &s[k]);
 s[k] = '\0'; /* 使输入的k个字符构成字符串 */
```

  - 当字符串由程序逐个字符生成时，程序也必须在它最后接上字符串结束标志符

```
for(k = 0; k < 26; k++)
     s[k] = 'a'+k;
s[k] = '\0';
```

常用字符串处理函数

- **求字符串长度函数 strlen()**
  - 函数调用strlen(str)返回str中的有效字符(不包括‘\0’)个数

- **字符串拷贝函数strcpy()**
  - 函数调用strcpy(str1,str2)将字符串str2复制拷贝到字符数组str1。限定字符数组str1要定义得足够大，以便能容纳被拷贝的str2的全部内容
    - str1不能是字符串常量

常用字符串处理函数(续)

- 字符串拷贝函数strncpy()
  - 函数调用strncpy(str1, str2, n)的作用是将str2中的前n个字符拷贝到str1(并附加‘\0’)
  - 其中n是整型表达式，指明欲拷贝的字符个数。如果str2中的字符个数不多于n，则函数调用strncpy(str1, str2, n)等价于strcpy(str1, str2)
    - 限制:str1不能是字符串常量

常用字符串处理函数(续)

- **字符串连接函数 strcat()**
  - 函数调用strcat(str1, str2)将str2内容拷贝接在字符数组str1中的字符串的后面，返回str1的开始地址
    - 字符串连接前，str1和str2都各自有‘\0’，连接后，str1中原来的‘\0’在拷贝时被覆盖掉，而在新的字符串有效字符之后再保留一个‘\0’
    - str1不能是字符串常量
  - 例如

```
char str1[30] = "Beijing ";
char str2[30] = "Shanghai";
strcat(str1, str2); printf("%s\n", str1);
```

  - 将输出 “Beijing Shanghai”

常用字符串处理函数(续)

- 字符串比较函数 strcmp()
  - 函数调用strcmp(str1, str2)比较两个字符串的大小
    - 对两个字符串自左至右逐对字符相比较(按字符的ASCII代码值的大小)，直至出现不同的字符或遇到\0字符为止
    - 如直至\0字符，全部字符都相同，则认为相等，函数返回0值
    - 若出现不同的字符，则以这第一个不相同的字符比较结果为准。若str1的那个不相同字符小于str2的相应字符，函数返回一个负整数；反之，返回一个正整数
    - strcmp("abcd","abdc"); strcmp("abcd","aa");
  - 对字符串不允许施行相等“==”和不相等“!=”运算，必须类似于本函数那样，通过逐个字符比较来实现

常用字符串处理函数(续)

- **字符串中大写英文字符转换成小写英文字符函数 strlwr()**
  - 函数调用strlwr(str)将存于字符数组str的字符串内大写英文字符转换成小写英文字符
    - str不能是字符串常量
- **字符串小写字母转换成大写字母函数 strupr()**
  - 函数调用 strupr(str)将存于字符数组str的字符串内的小写英文字符转换成大写英文字符
    - str不能是字符串常量

常用字符串处理函数(续)

- **字符串输出函数 puts()**
  - 函数调用puts(str)将str的字符串输出到终端，并将str中的\0字符转换成换行符‘\n’ 。即输出字符串内容后，并换行
    - puts(str)相当于printf("%s\n",str)
- **字符串输入函数 gets()**
  - 函数调用gets(str)从终端输入字符序列到字符数组str，字符序列以回车符作为结束，将输入时的回车符转换成’\0’字符存储，并返回str的存储开始地址

字符数组和字符串应用样例-1

程序示例：
说明上述库函数的应用

```
#include <string.h>
char s1[100], s2[100], s3[100];
void main()
{ printf("输入字符序列s1\n"); scanf("%s", s1);
  while(getchar() != '\n');
  printf("s1 = %s\n", s1);
  strcpy(s2, s1);  strncpy(s3, s1, 5);
  printf("从s1拷贝的s2 = %s\n", s2);
  printf("从s1只拷贝5个字符的s3 = %s\n", s3);
  strcat(s1,s3);printf("接上s3的s1=%s\n", s1);
  printf("输入字符行到s1\n");  gets(s1);
  printf("s1 = %s\n", s1);
  strcpy(s2, s1);   strncpy(s3, s1, 5);
  puts(s2);puts(s3);strcat(s1,s3);puts(s1);
}
```

字符数组和字符串应用样例-2

- 输入字符行，统计各英文字母出现的次数
- 程序设计分析
  - 程序需要一个存储字符行的数组，还需要52个英文字母计数器，统计52个英文字母的出现次数
  - 顺序扫视输入的字符行中的字符，直至遇字符串结束符结束。当程序发现当前字符是英文字母时，对应英文字母的计数器增1，否则掠过当前字符
  - 用数组count[52]实现52个计数器，并设大写英文字符顺序对应前26个计数器(count[0]~count[25])，小写英文字符顺序对应后26个计数器(count[26]~count[51])

字符数组和字符串应用样例-2(续)

- 输入字符行，统计各英文字母出现的次数

```
#include  <stdio.h>
void main()
{  char buf[120];    int  i, count[52];
   printf("Enter letter line.\n"); gets(buf);
   for(i = 0; i < 52; i++)  count[i] = 0;
   for(i = 0; buf[i] != '\0'; i++)
       if (buf[i] >= 'A' && buf[i] <= 'Z')
          count[buf[i] - 'A']++;
       else if (buf[i] >= 'a' && buf[i] <= 'z')
                count[buf[i] - 'a' + 26]++;
   for(i = 0; i < 52; i++)
       if (count[i]) /* 未出现的字母不输出 */
          printf("%c(%d)\t", i < 26 ? i+'A' :i-26+'a', count[i]);
   printf("\n\n");
}
```

问题

- 给你一部《红楼梦》，请统计各个汉字出现频度

字符数组和字符串应用样例-3

- 输入字符行，统计其中单词个数。约定单词由英文字母组成，其它字符只是用来分隔单词

- 例子

  “ My name is Tom, now I am a student of Fudan University.”

- 英文单词：由连续的英文字母构成；从一个英文字母开始，到一个非英文字母结束。

字符数组和字符串应用样例-3

- 程序设计分析

```
{  设置单词计数器等初值;
   输入一行字符（如用gets()函数）;
   顺序扫视输入的字符行（如用for语句） {
    判定当前字符是否是字母;
     if (现在正在单词中){
        if (当前字符不是字母)
          设置当前状态不在单词中的标志;
     }
     else if (当前字符是字母){/* 当前不是在单词中,但获得一个字母
            设置当前状态在单词中的标志;
            单词计数器增 1 ; /* 状态由不是单词变为单词时，进行统计
    */
     }
   }
}
```

字符数组和字符串应用样例-3(续)

```
#include <stdio.h>
void main()
{  char c, line[120];
   int  i, words, inword, letter; words = 0;
   inword = 0; /* 预置状态不在单词中 */
   printf("Input a line.\n");  gets(line);
   for(i = 0; line[i]; i++) {
     c = line[i];
     letter = ((c>='a' && c<='z') || (c>='A'&& c<='Z'));
     if (inword){if (!letter)  inword = 0; }
     else if (letter){inword = 1; words++; }
   }
   printf("字符行内有 %d 个单词\n",words);
}
```

输入字符行，统计其中单词个数。约定单词由英文字母组成，其它字符只是用来分隔单词

问题

- 输入字符行，统计其中单词出现频度
- 输入字符行，统计其中“正确”单词出现频度
- 输入一行中文，统计其中各个词语出现频度。譬如：“我是复旦大学学生。”

问题

- 给你一部《红楼梦》，请统计其中出现的人物（有名有姓）个数

字符数组和字符串应用样例-4

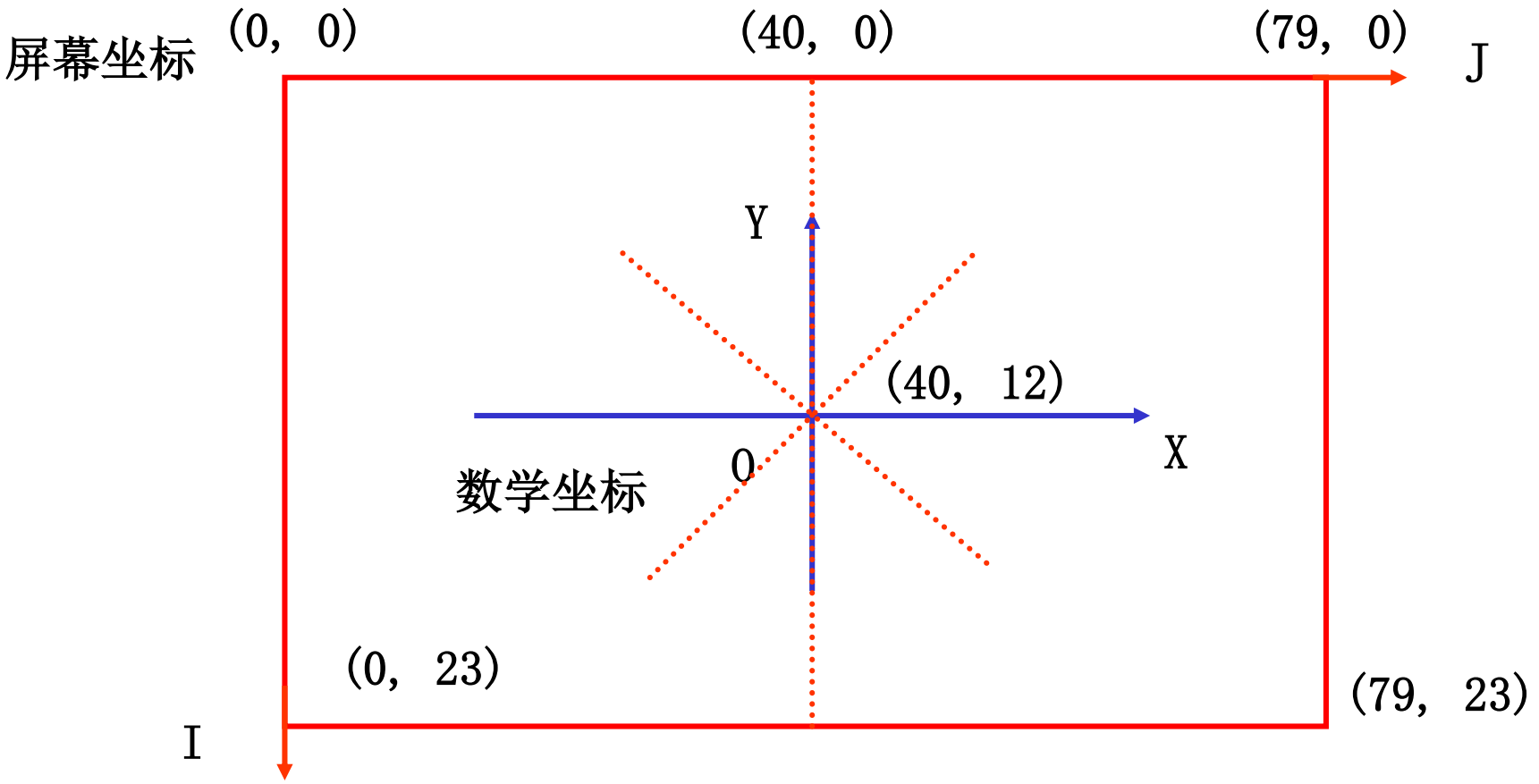
读入一串字符行，以空行(即只键入回车符的行)结束，输出其中最长的行

```
#include <stdio.h>
#include <string.h>
#define MAXLINE 256
void main()
{ int  len, max; char line [MAXLINE], save [MAXLINE];
  max = 0;
  for (; ;) {
    printf("输入字符行(回车结束).\n");  gets(line);
    if ((len = strlen(line)) == 0) break;
    if (len > max) {
        max = len;  strcpy(save, line);  }
    }
  if (max > 0)  printf("最长行是:\n%s\n", save);
}
```

字符数组和字符串应用样例-5

- 描绘图像 r(θ) = sin(2θ)

数学坐标与屏幕坐标



字符数组和字符串应用样例-5(续)

- 描绘图像 r(θ) = sin(2θ)(续)
- 程序设计分析
  - 用字符在显示屏上描绘 r = r(θ) 图像的一般方法
    - 当θ变化时，r(θ)描绘出一个封闭平面图，用一个二维字符数组代表图像r(θ)的画面，可定义为

      char canvas [24][80];
    - 普通显示屏在25mm(一英寸)见方的区域上纵向 5.5 行，横向约10列。按这个比例设定X、Y的比例。约定屏幕左上角的字符位置对应canvas[0][0]，图像坐标系的原点位于 canvas[12][40]
    - θ取0到360范围内的值，步长为1。对于具体函数，应充分利用函数图像对称性：X 轴对称、Y 轴对称、关于原点中心对称等

字符数组和字符串应用样例-5(续)

- 程序设计分析(续)
  - 描绘图像程序的算法结构

```
{ 置画面数组为全空白符;
  for (seta = 0; seta < 360; seta++) {
    alpha = seta / 180.0 * pi ; /* 度->弧度 */
    r = r(seta)*scale ;              /* 放大 */
    x = r*cos(alpha);  y = r*sin(alpha); /* 计算数学坐标
    i = 12-(int)(0.55*y); /*计算数学坐标y->屏幕坐标i；屏幕的行号递增方向与图像 Y 轴的方向相反 */
    j = 40+x; /*计算数学坐标x->屏幕坐标j；屏幕的列号递增方向与图像 x轴的方向相同 */
    canvas[i][j] = '*' ;
  }
}
```

字符数组和字符串应用样例-5(续)

```
#include <stdio.h>
#include <math.h>
#define SCALE 20.0
#define MID   12
#define PI    3.14159
#define RATE  0.55
void main()
{ char canvas[2*MID][80];
  double r, alpha, x, y;
  int i, j, seta;
 for(i = 0; i < 2*MID; i++)
    for(j = 0; j < 80; j++)
       canvas[i][j] = ' ';
```

描绘图像 r(θ) = sin(2θ)(续)

对于本题，scale 可定义为20。以下程序利用图像关于X 轴、Y 轴及关于直线Y = X的对称性

```
for(i=0;i<2*MID;i++) canvas[i][79]='\0';
for(seta=0; seta<45; seta++) {
    alpha = seta/180.0*PI;
    r     = sin(2*alpha)*SCALE;
    x     = r*cos(alpha);
    y     = r*sin(alpha);
    i     = (int)(RATE*y);
    j     = (int)(x);
```

字符数组和字符串应用样例-5(续)

- 描绘图像 r(θ) = sin(2θ)(续)

  对于本题，scale 可定义为20。以下程序利用图像关于X 轴、Y轴及关于直线Y = X的对称性

```
    canvas[MID+i][40+j] =  /* X 轴对称 */
    canvas[MID+i][40-j] =  /* 中心对称 */
    canvas[MID-i][40-j] =  /* Y轴对称 */
    canvas[MID-i][40+j] = '*';/* 本身图像点 */
    i = (int)(RATE*x); /* 关于直线X = Y 对称 */
    j = (int)(y);
    canvas[MID+i][40+j] =  /* X 轴对称 */
    canvas[MID+i][40-j] =  /* 中心对称 */
    canvas[MID-i][40-j] =  /* Y轴对称 */
    canvas[MID-i][40+j] = '*'; /* 本身图像点 */
  }
  for(i = 0; i < 2*MID; i++) puts(canvas[i]);
}
```

第四章作业

- 思考题与习题
  - 第5，11，12，14，16题