

第10讲 数组（下）

黄永峰

2022.11.108

主要内容

- 字符数组定义、初始化和引用
- 字符处理函数
- 数组名做函数参数
- 应用实例

参考教材的第9章的9.3-9.5

10.1 字符数组

- 字符数组定义: char 数目名[长度]

例如: char c[10];

- 字符数组: 存放字符数据的数组, 每一个元素存放一个字符

c[0]	c[1]	c[2]	c[3]	c[4]	c[5]	c[6]	c[7]	c[8]	c[9]
I	□	a	m	□	h	a	p	p	y

c[0]='I'; c[1]=' '; c[2]='a'; c[3]='m'; c[4]=' ';
c[5]='h'; c[6]='a'; c[7]='p'; c[8]='p'; c[9]='y';

10.1 字符数组

—字符数组的‘字符常量’初始化

- 逐个元素初始化

例： `char c[10] = {'I', ' ', 'a', 'm', ' ', 'h', 'a', 'p', 'p', 'y'};`

c[0]	c[1]	c[2]	c[3]	c[4]	c[5]	c[6]	c[7]	c[8]	c[9]
I	□	a	m	□	h	a	p	p	y

- 初始数据少于数组长度, 多余元素自动为“空” (‘\0’,)

例： `char c[12] = {'c', ' ', 'p', 'r', 'o', 'g', 'r', 'a', 'm'};`

c[0]	c[1]	c[2]	c[3]	c[4]	c[5]	c[6]	c[7]	c[8]	c[9]	c[10]	c[11]
c	□	p	r	o	g	r	a	m	\0	\0	\0

- 初始化时, 若未指定数组长度, 则长度等于初值个数

例： `static char c[] = {'I', ' ', 'a', 'm', ' ', 'h', 'a', 'p', 'p', 'y'};`

10.1 字符数组

- 字符数组的“字符串”常量初始化

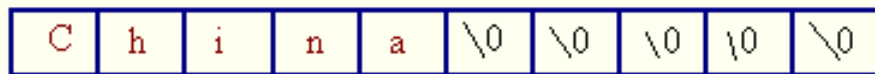
例: `char c[] = {"I am happy"};`

可以不要大括号 `char c[] = "I am happy";`

- 字符串在存储时，系统自动在其后加上结束标志‘\0’；
- `char c[6] = {"China"};`



- `char c[10] = {"China"};`



- 提问: `static char c[5] = { "China" };` 会怎样?

解决方案 "20091110" (C)

- 20091110
 - 头文件
 - 源文件
 - 1.cpp
 - 资源文件

解决... 类视图 属性...

```

1  #include<stdio.h>
2  int main()
3  {static char c[10]={'I',' ','a','m',' ','a',' ','b','o','y'};
4    char b[10]="i am a boy";
5    int i;
6    for(i=0;i<10;i++)
7    {printf("%c",c[i]);
8      printf("%c",b[i]);}
9    printf("\n");
10   return 0;}
11
            
```

输出

显示以下输出(S): 生成

```

1>----- 已启动全部重新生成: 项目: 20091110, 配置: Debug Win32 -----
1>正在删除项目 "20091110" (配置 "Debug|Win32")的中间文件和输出文件
1>正在编译...
1>1.cpp
1>c:\users\huang yf\documents\visual studio 2005\projects\黄\20091110\1.cpp(4) : error C2117: "b": 数组界限溢出
1>      c:\users\huang yf\documents\visual studio 2005\projects\黄\20091110\1.cpp(4) : 参见 "b" 的声明
1>c:\users\huang yf\documents\visual studio 2005\projects\黄\20091110\1.cpp(4) : error C2117: "b": 数组界限溢出
1>      c:\users\huang yf\documents\visual studio 2005\projects\黄\20091110\1.cpp(4) : 参见 "b" 的声明
1>生成日志保存在 "file:///c:/Users/huang yf/Documents/Visual Studio 2005/Projects/黄\20091110/Debug/BuildLog.htm"
1>20091110 - 2 个错误, 0 个警告
===== 全部重新生成: 0 已成功, 1 已失败, 0 已跳过 =====
            
```



```
1 #include<stdio.h>
2 #pragma warning(disable:4996)
3 int main()
4 {
5     char a[10];
6     char b[9];
7     scanf("%s%s", a, b);
8     printf("%s\n", a);
9     printf("%s\n", b);
10    return 0;}
```

已用时间 <= 6,258ms

局部变量

搜索(Ctrl+E)

搜索深度: 3

名称

值

已返回 scanf

2

a

0x000000395acff758 "huang"

[0]

104 'h'

[1]

117 'u'

[2]

97 'a'

[3]

110 'n'

[4]

103 'g'

[5]

0 '\0'

[6]

-52 '?'

[7]

-52 '?'

[8]

-52 '?'

[9]

-52 '?'

b

0x000000395acff788 "yongfeng"

[0]

121 'y'

[1]

111 'o'

[2]

110 'n'

[3]

103 'g'

[4]

102 'f'

调用堆栈

名称

Project1.exe!main(...) 行 8

[外部代码]

D:\My_New_UserFiles\New_Desk

huang yongfeng


```
1  #include<stdio.h>
2  #pragma warning(disable:4996)
3  int main()
4  {
5      int i;
6      char f[8];
7      for (i = 0; i < 8; i++)
8          scanf("%c", &f[i]);
9      for (i = 0; i < 8; i++)
10         printf("%c", f[i]);
11     return 0;
12 }
13
```

Microsoft Visual Studio 调试控制台

```
yongfeng
yongfeng
D:\My_New_UserFiles\New_
按任意键关闭此窗口. . .
```

```

1  #include<stdio.h>
2  #pragma warning(disable:4996)
3  int main()
4  {
5      char c[10];
6      static char b[10];
7      scanf("%s", c);
8      scanf("%s", b);
9      int i;
10     for (i = 0; i < 10; i++)
11         printf("%c", c[i]);
12     printf("\n");
13     printf("%s\n", b);
14     printf("%s\n", c);
15     return 0;

```

Microsoft Visual Studio 调试控制台

```

huang yong feng
huang 烫烫
yong
huang

```

D:\My_New_UserFiles\New_Desk
要在调试停止时自动关闭控制台
按任意键关闭此窗口. . .

108 % 0 2 < >

局部变量

搜索(Ctrl+E)



< > 搜索深度: 3

名称	值	类型
[3]	103 'g'	char
[4]	0 '\0'	char
[5]	0 '\0'	char
[6]	0 '\0'	char
[7]	0 '\0'	char
[8]	0 '\0'	char
[9]	0 '\0'	char
c	0x00000026acf6f698 "huang"	char[10]
[0]	104 'h'	char
[1]	117 'u'	char
[2]	97 'a'	char

调用堆栈

名称

Project1.exe!main(...) 行 10
[外部代码]

断点 异常设置 命令窗口 即时窗口 输出 自动窗口 局部变量 监视 1

10.1 字符数组

输出注意事项:

- “%s” 格式输出字符串时，printf() 函数的输出项字符数组名，而不是元素名 `char c[6] = "China";`
`printf("%s" , c);` // 注意与 `printf("%c" , c[0])` 差异
- “%s” 格式输出时，即使数组长度大于字符串长度，遇 ‘\0’ 也结束，且输出字符中不包含 ‘\0’
例： `char c[10] = {"China"};`
`printf("%s" , c);` /*只输出5个字符 */

输入注意事项:

- 输入时，遇回车键结束，但获得的字符中不包含回车键本身（0x0D, 0x0A），而是在字符串末尾添 ‘\0’。因此，定义的字符数组必须考虑 ‘\0’。（如，输入5个字符，定义字符数组至少应有6个元素）。
- 数组名代表该数组的起始地址，因此，scanf() 函数中不需要地址运算符&
- scanf() 函数输入多个字符串，输入时以“空格”键作为字符串间的分隔。

例：char str1[15], str2[10];

scanf(“%s”, str1, str2);

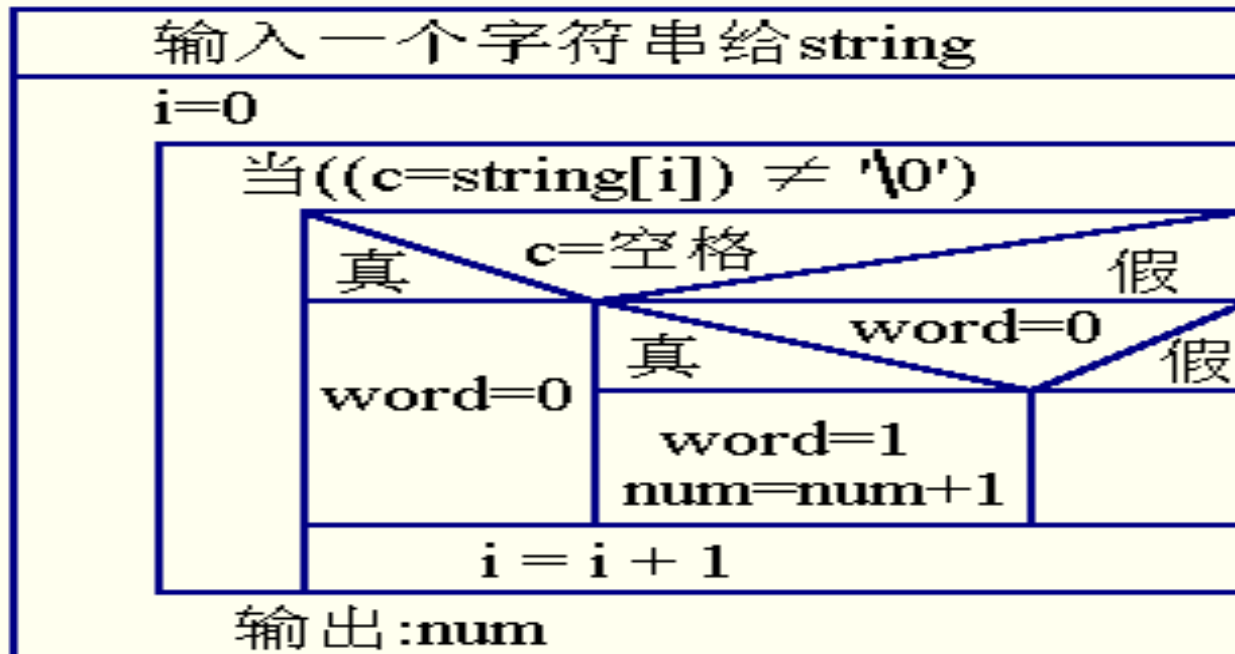
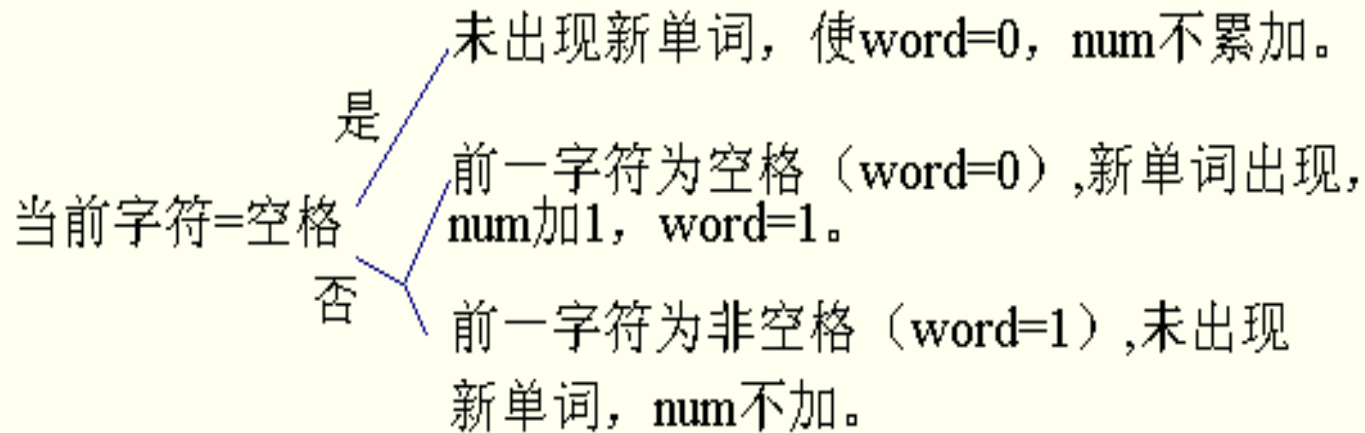
输入：How are you?

- ◆ 提问：str1和str2分别是什么？如果改为“gets(str1);” str1 是什么？

10.1 字符数组

- 输入一行字符，统计其中有多少个单词（单词间以空格分隔）。例如，输入 “I am a boy. ”，有4个单词。
- 算法：
 - 单词的数目由空格出现的次数决定（连续出现的空格记为出现一次；一行开头的空格不算）。应逐个检测每一个字符是否为空格。
 - 用num表示单词数（初值为0）。word=0表示前一字符为空格，word=1表示前一字符不是空格，word初值为0。如果前一字符是空格，当前字符不是空格，说明出现新单词，num加1。

10.1 字符数组



算法重点：字符串I/O函数的使用及差异，单词统计方法

```
1 #include "stdio.h"
2 int main()
3 {char string[81];
4   int i,num = 0,word = 0;
5   char c;
6   gets(string);
7   for(i=0;(c=string[i])!='\0';i++)
8       if (c==' ') word = 0;
9       else if (word == 0)
10          {word = 1;
11            num++;}
12   printf("There are %d words in the line\n",num);
13 }
```

C:\Windows\system32\cmd.exe

```
huang yong feng is a teacher
There are 6 words in the line
请按任意键继续. . .
```


10.1 字符数组

二维字符数组

解决方案 “huang” (1 个项)

- huang
 - 头文件
 - 源文件
 - huang.cpp
 - 资源文件

```

1  #include<stdio.h>
2  #pragma warning(disable:4996)
3  int main()
4  {
5      int i;
6      char a[3][8];
7      scanf("%s%s%s", a[0], a[1], a[2]);
8      for (i = 0; i < 3; i++) 已用时间 <= 7,600ms
9          printf("%s\n", a[i]);
10     return 0;}

```

自动窗口

名称	值	类型
a	0x0012ff3c	char [3][8]
a[0]	0x0012ff3c	char [8]
[0]	-52	char
[1]	-52	char
[2]	-52	char
[3]	-52	char
[4]	-52	char
[5]	-52	char
[6]	-52	char
[7]	-52	char
a[1]	0x0012ff44	char [8]
a[2]	0x0012ff4c	char [8]
i	-858993460	int

解决方案 “huang” (1 个项)

- huang
 - 头文件
 - 源文件
 - huang.cpp
 - 资源文件

解决方案资... 类视图

```

1  #include<stdio.h>
2  #pragma warning(disable:4996)
3  int main()
4  {
5      int i;
6      char a[3][8];
7      scanf("%s%s%s", a[0], a[1], a[2]);
8      for (i = 0; i < 3; i++) 已用时间 <= 7,600ms
9          printf("%s\n", a[i]);
10     return 0;

```

C:\Documents and Settings\...
huang yong feng

C:\WINDOWS\system32\cmd.exe
huang yong feng
huang
yong
feng
请按任意键继续...

自动窗口

名称	值	类型
a	0x0012ff3c	char [3][8]
a[0]	0x0012ff3c "huang"	char [8]
[0]	104 'h'	char
[1]	117 'u'	char
[2]	97 'a'	char
[3]	110 'n'	char
[4]	103 'g'	char
[5]	0	char
[6]	-52	char
[7]	-52	char
a[1]	0x0012ff44 "yong"	char [8]
a[2]	0x0012ff4c "feng"	char [8]
i	-858993460	int

10.2 字符串处理函数

在C++的函数库中（Cstring, 或string.h），提供一些字符串处理函数

1、puts() 函数：输出字符串（以'\0'结尾）。例、

```
char c[6]="China";
```

printf、puts均以'\0'结尾.

```
printf("%s\n", c); printf需要格式控制符%s
```

```
puts(c); puts不需要格式控制符，且自动换行
```

2、gets() 函数：输入字符串到数组。例、

```
char str[12];
```

```
gets(str);
```

注意：gets()、puts() 一次只能输入输出一个字符串。gets() 可以输入空格隔开的字符串。

3、strcat()：连接字符串。

```
strcat(字符串1, 字符串2);
```

把“字符串2” 连接到“字符串1” 的后面。

4、strcpy()：字符串拷贝。

```
strcpy(字符串1, 字符串2); 把“字符串2” 的值拷贝到“字符串1” 中。
```

10.2 字符串处理函数

5. `strcmp()`：字符串比较。

`int strcmp(字符串1, 字符串2)`；比较“字符串1”，字符串2”

例：`strcmp(str1, str2)`；或 `strcmp("China", "Korea")`；

比较规则：逐个字符比较ASCII码，直到遇到不同字符或‘\0’，比较结果是该函数的返回值

- 字符串1 < 字符串2 `strcmp()` 返回值<0
- 字符串1 == 字符串2 `strcmp()` 返回值==0
- 字符串1 > 字符串2 `strcmp()` 返回值>0
- 注意：字符串只能用`strcmp`函数比较，不能用关系运算符“==”比较

6. `strlwr()`：将大写字母转换为小写字母（lwr：lowercase）

7. `strupr()`：将小写字母转换为大写字母（upr：uppercase）

- 提问：长度不同的字符串可以进行比较，结果呢？

10. 2字符串处理函数

- 头文件ctype. h中声明了一些测试字符的函数

函数	功能
int isdigit(char c)	判断字符是否是数字(0 – 9)
int isalpha(char c)	判断字符是否是字母(A – Z or a – z)
int isalnum(char c)	判断是否是数字和字母(A – Z, a – z, or 0 – 9)
int islower(char c)	判断字符是否是小写字母
int isupper(char c)	判断字符是否是大写字母
int isspace(char c)	判断字符是否是空格、换行符、制表符等
int ispunct (char c)	判断字符是否是标点符号(字母、数字、空格以外的字符)
int tolower(char c)	将字符转化为小写字符
int toupper(char c)	将字符转化为大写字符

10.2 字符数组应用举例

- 输入三个字符串，并找出其中最大者
- 分析：用`strcmp()`函数比较字符串的大小。首先比较前两个，把较大者拷贝给字符数组变量`string`（用`strcpy()`函数拷贝），再比较`string`和第三个字符串
- 程序：设字符串最长为19个字符

```

1 #include "string.h" /* strcmp、strcpy在string.h中定义 */
2 #include "stdio.h"
3 int main()
4 {char string[20];/* 存最大字符串 */
5   char str[3][20]; /* 三个字符串 */
6   int i;
7   for(i=0;i<3;i++)
8     gets(str[i]); /* 输入三个字符串 */
9   if (strcmp(str[0],str[1])>=0)
10     strcpy(string,str[0]);
11   else strcpy(string,str[1]);
12   if (strcmp(str[2],string)>0)strcpy(string,str[2]);
13   printf("\nthe largest string is: \n%s\n",string);
14   return 0;}

```

C:\Windows\system32\cmd.exe

```

huang yong feng
li xing
Zhang pei

```

```

the largest string is:
li xing
请按任意键继续. . .

```

10.3 数组作函数参数

- 数组元素做函数参数：值的结合，单向传递
- 数组名做函数参数：**地址结合，双向传递**
 - 数组可以作为函数参数。形参被说明为数组，对应的实参是同类型的数组名
 - 参数传递时，传给被调用函数形参的值是**实参数组的地址**，而不是对实参数组元素的复制。因此，在被调用函数中，通过形参引用的数组元素不是实参数组元素的副本，而是实参数组元素本身
 - 所以被调用函数中对数组元素的任何修改都会直接修改实参数组元素本身


```

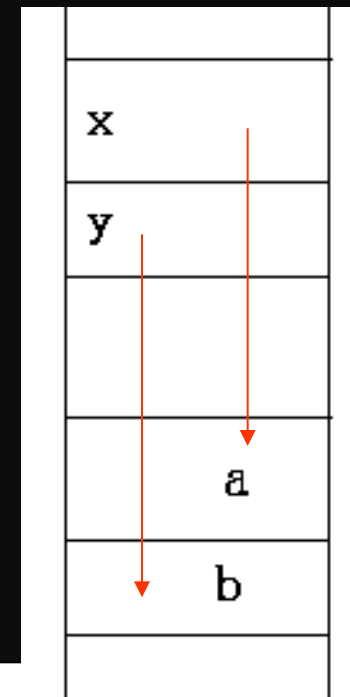
1  #include<stdio.h>
2  #pragma warning(disable:4996)
3  int main()
4  {   int x, y;
5      int exchange(int, int);
6      scanf("%d%d", &x, &y);
7      exchange(x, y);
8      printf("x=%d y=%d\n", x, y);
9      return 0;}
10 int exchange(int a, int b)
11 {
12     int temp;
13     temp = a;
14     a = b;
15     b = temp;
16     printf("a=%d b=%d\n", a, b);
17     return 0;
18 }

```

Microsoft Visual Studio 调试控制

4 8
a=8 b=4
x=4 y=8

D:\My_New_UserFiles\
按任意键关闭此窗口。



算法重点：变量做函数参数，单向传递

```

1  #include<stdio.h>
2  #pragma warning(disable:4996)
3  int main()
4  {   int x[2];
5      int exchange(int a[]);
6      scanf("%d%d", &x[0], &x[1]);
7      exchange(x);
8      printf("x[0]=%d x[1]=%d\n", x[0], x[1]);
9      return 0;}
10 int exchange(int a[])
11 {
12     int temp;
13     temp = a[0];
14     a[0] = a[1];
15     a[1] = temp;
16     printf("a[0]=%d a[1]=%d\n", a[0], a[1]);
17     return 0;

```

Microsoft Visual Studio 调试控制台

```

4 8
a[0]=8 a[1]=4
x[0]=8 x[1]=4

```

D:\My_New_UserFiles\New
按任意键关闭此窗口. . .

x		a
	$x[0]/a[0]$	
	$x[1]/a[1]$	

算法重点：数组做函数参数，双向传递

10.3 数组作函数参数

```

1  #include <stdio.h>
2  #pragma warning(disable:4996)
3  int main()
4  {   int x[2];
5      int exchange(int a[]);
6      scanf("%d%d", &x[0], &x[1]);
7      exchange(x);
8      printf("x[0]=%d x[1]=%d\n", x[0], x[1]);
9      return 0;}
10 int exchange(int a[])
11 {
12     int temp;
13     temp = a[0];
14     a[0] = a[1];
15     a[1] = temp;
16     printf("a[0]=%d a[1]=%d\n", a[0], a[1]);
17     return 0;}

```

C:\Documents and Settings\Administrator\I
4 8

自动窗口

名称	值
x	0x0012ff5c
[0]	4
[1]	8
x[0]	4
x[1]	8

自动.. 局部.. 线程 模块 监视 1

```

1  #include<stdio.h>
2  #pragma warning(disable:4996)
3  int main()
4  {   int x[2];
5      int exchange(int a[]);
6      scanf("%d%d", &x[0], &x[1]);
7      exchange(x);
8      printf("x[0]=%d x[1]=%d\n", x[0], x[1]);
9      return 0;}
10 int exchange(int a[])
11 {
12     int temp;
13     temp = a[0]; 已用时间 <= 1ms
14     a[0] = a[1];
15     a[1] = temp;
16     printf("a[0]=%d a[1]=%d\n", a[0], a[1]);
17     return 0;}
  
```

监视 1

名称	值
+ a	0x0012ff5c
+ x	0x0012ff5c

自动窗口 局部变量 线程 模块 监视

```

1  #include<stdio.h>
2  #pragma warning(disable:4996)
3  int main()
4  {   int x[2];
5      int exchange(int a[]);
6      scanf("%d%d", &x[0], &x[1]);
7      exchange(x);
8      printf("x[0]=%d x[1]=%d\n", x[0], x[1]);
9      return 0;}
10 int exchange(int a[])
11 {
12     int temp;
13     temp = a[0];
14     a[0] = a[1];
15     a[1] = temp;
16     printf("a[0]=%d a[1]=%d\n", a[0], a[1]);
17     return 0;}

```



数组元素做函数参数

```

1  #include<stdio.h>
2  #pragma warning(disable:4996)
3  int main()
4  {   int max_value(int x, int max);
5      int i, j, row = 0, colum = 0, max;
6      int a[3][4] = { {5,12,23,56},{19,28,37,46},{-12,-34,6,8} };
7      max = a[0][0];
8      for(i=0;i<=2;i++)
9          for(j=0;j<=3;j++)
10             {   max = max_value(a[i][j], max);
11                 if(max==a[i][j])
12                     {   row = i;
13                         colum = j;}}
14     printf("max=%d,row=%d,colum=%d", max, row, colum);
15     return 0;}
16 int max_value(int x,int max)
17 {   if (x > max) return x;
18     else return max;}

```

Microsoft Visual Studio 调试控制台

```

max=56, row=0, colum=3
D:\My_New_UserFiles\New_Desktop\te
按任意键关闭此窗口. . .

```

10.3 数组作函数参数

- ◆ 冒泡排序算法
- ◆ 选择排序算法
- ◆ 插入排序算法
- ◆ 快速排序算法

.....

□ 冒泡排序思想

- 将待冒泡的数据从头到尾依次处理：比较相邻的两个元素，如果大的在前小的在后，就交换这两个元素。这样经过从头到尾的检查，最大的一个就被交换到最后了。剩下元素可以进行递归冒泡。
- 如果在一次冒泡中没有发生交换，则表示数据都已拍好序，不需要再进行冒泡

10.3 数组作函数参数

原始序列	5	7	3	0	4	2	1	9	6	8
第一趟排序	5	3	0	4	2	1	7	6	8	9
第二趟排序	0	3	2	1	4	5	6	7	8	9
第三趟排序	0	3	2	1	4	5	6	7	8	9
第四趟排序	0	1	2	3	4	5	6	7	8	9
第五趟排序	0	1	2	3	4	5	6	7	8	9

冒泡排序算法设计

采用两重计数型循环：

- 步骤1： 将待排序的数据放入数组中；
- 步骤2： 置j为1； $j \leq n$ ；
- 步骤3： 让i从1到n-j，比较a[i]与a[i+1]，
如果 $a[i] \geq a[i+1]$ ，位置不动；
如果 $a[i] < a[i+1]$ ，位置交换，即
$$p = a[i]; a[i] = a[i+1]; a[i+1] = p;$$

步骤3结束后 a[n-j+1]中的数为最小的数
- 步骤4： 让j=j+1；只要j!=n就返回步骤3，
将a[n-j+1]的值排好。当j==n时执行步骤5
- 步骤5： 输出排序结果。

```

1  #include <stdio.h>
2  int main()
3  {int i, j, p, a[7];
4  for (i=1; i<=6; i=i+1)
5  {printf("请输入待排序的数a[%d]=", i);
6   scanf("%d", &a[i]);
7  }
8  for (j=1; j<=5; j=j+1)
9  for (i=1; i<=6-j; i=i+1)
10 {if(a[i]<a[i+1] )
11   {p=a[i];
12    a[i]=a[i+1];
13    a[i+1]=p;}}
14 for (i=1; i<=6; i=i+1)
15 printf("%d\n", a[i]);
16 return 0;
17 }
18

```

C:\Windows\system32\cmd.exe

```

请输入待排序的数a[1]=32
请输入待排序的数a[2]=3
请输入待排序的数a[3]=2
请输入待排序的数a[4]=3
请输入待排序的数a[5]=4
请输入待排序的数a[6]=44
44
32
4
3
3
2
请按任意键继续. . .

```

冒泡排序法

解决方案 "20091110"

- 20091110
 - 头文件
 - 源文件
 - 1.cpp
 - 2.cpp
 - 资源文件

```

1  #include <stdio.h>
2  int main()
3  {int i, a[7];
4  for(i=1; i<=6; i=i+1)
5  {printf("请输入待排序的数a[%d]=", i);
6  scanf("%d", &a[i]);
7  }
8  int sort(int a[], int);
9  sort(a, 6);
10 for(i=1; i<=6; i=i+1)
11 printf("%d\n", a[i]);
12 return 0;
13 }
14

```

自动窗口

名称	值	类型
a	0x0012ff04	int [7]
[0]	-858993460	int
[1]	22	int
[2]	1	int
[3]	223	int
[4]	221	int
[5]	2	int
[6]	55	int

解决方案 "20091110"

20091110

头文件

源文件

1.cpp

2.cpp

资源文件

```
1 int sort(int b[], int n)
2 {int p, i, j;
3  for (j=1; j<=n; j=j+1)
4  for (i=1; i<=n-j; i=i+1)
5  {if (b[i]<b[i+1] )
6    {p=b[i];
7      b[i]=b[i+1];
8      b[i+1]=p;}}
9  return 0;}
10
```

解决方...

类视图

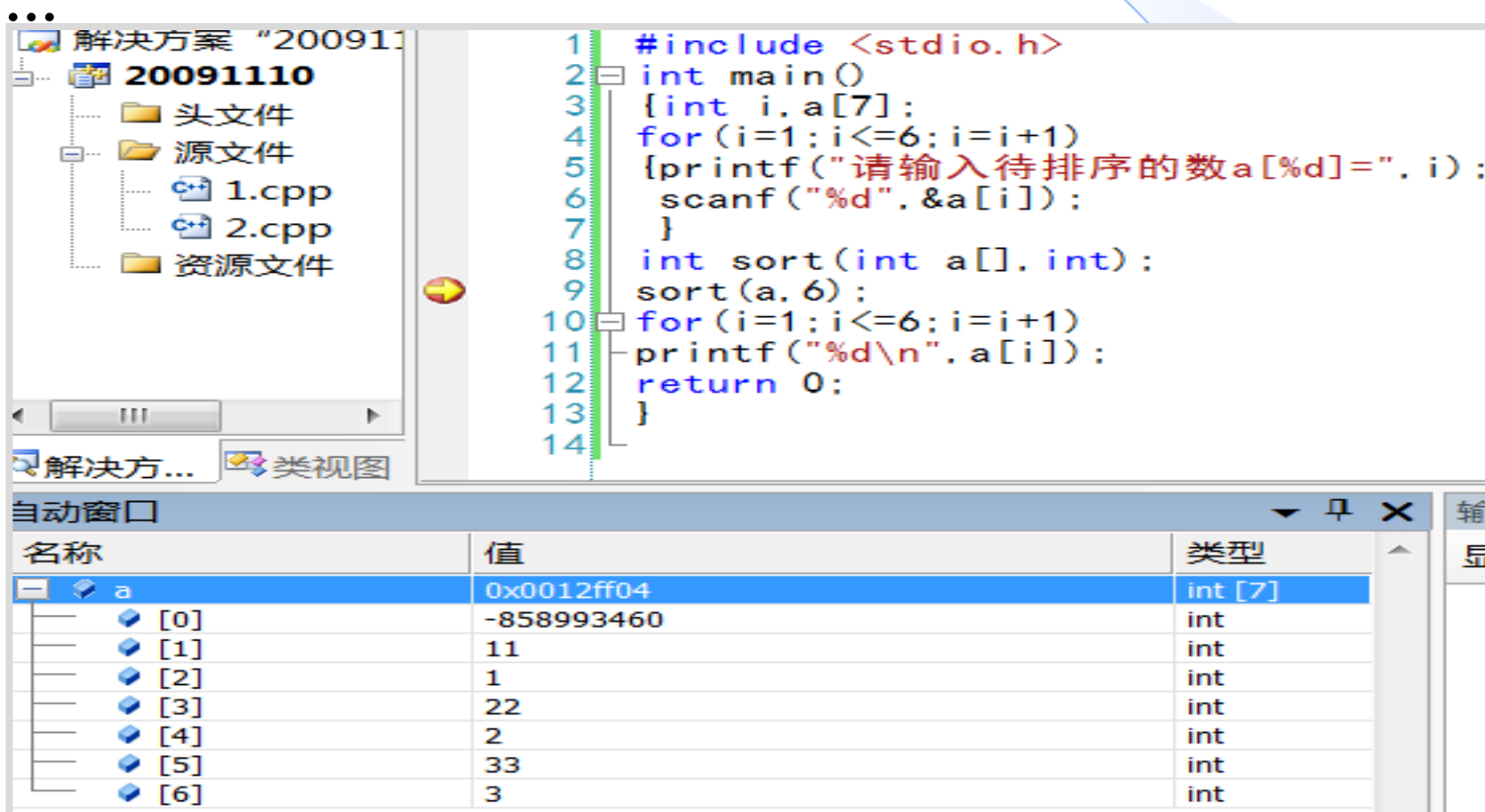
自动窗口

名称	值	类型
+ b	0x0012ff04	int *
b[i+1]	22	int
b[i]	1	int
i	4	int
j	1	int
n	6	int

10.3 数组作函数参数

● 选择法排序思想：

- (1) 从中选择一个最小的元素将其放在第1
- (2) 从剩下的子表中选择一个最小的元素将其放在第2



The screenshot shows a C++ IDE with a project named "20091110". The project structure includes "头文件" (Header Files), "源文件" (Source Files), and "资源文件" (Resource Files). The source files are "1.cpp" and "2.cpp". The "2.cpp" file is open, showing the following code:

```
1 #include <stdio.h>
2 int main()
3 {int i, a[7];
4 for(i=1; i<=6; i=i+1)
5 {printf("请输入待排序的数a[%d]=", i);
6 scanf("%d", &a[i]);
7 }
8 int sort(int a[], int);
9 sort(a, 6);
10 for(i=1; i<=6; i=i+1)
11 printf("%d\n", a[i]);
12 return 0;
13 }
14
```

The "自动窗口" (Automatic Window) at the bottom shows the memory dump for the variable "a". The memory address is 0x0012ff04, and the type is "int [7]". The values are as follows:

名称	值	类型
a	0x0012ff04	int [7]
[0]	-858993460	int
[1]	11	int
[2]	1	int
[3]	22	int
[4]	2	int
[5]	33	int
[6]	3	int

解决方案 "20091110"

20091110

头文件

源文件

1.cpp

2.cpp

资源文件

1

2

3

4

5

6

7

8

9

10

11

12

13

14

```

int sort(int b[], int n)
{
    int k, i, j, temp;
    for(j=1; j<n; j=j+1)
    {
        k=j;
        for(i=1+j; i<=n; i=i+1)
        {
            if(b[k]<b[i])
            {
                k=i;
            }
            if(k!=j)
            {
                temp=b[j];
                b[j]=b[k];
                b[k]=temp;
            }
        }
    }
    return 0;
}

```

自动窗口

名称	值	类型
b	0x0012ff04	int *
b[i]	3	int
b[k]	22	int
i	6	int
j	2	int
k	3	int
n	6	int

本讲重点

1. 字符数组的字符常量和字符串常量初始化方法.
2. 数组名做函数参数
地址结合方式；即形参数组名存放的实参数组的首地址；可以实现数据的双向传递；形参可以不指明大小。但为了控制明确，一般采用参数来指明形参数组的个数
3. 字符处理函数
4. 排序算法

附：字符串处理函数的编写实例

```
int strcmp (char s[ ],char t[ ])
{ int i = - 1;
  while (++i,s[i]==t[i]&& s[i]!='\0');
  return(s[i]=='\0'&&t[i]=='\0'?1:0);
}
```

```
void strcpy (char t[],char s[])
{
  int i=0;
  while ((t[i]=s[i])!='\0')
    i++;
}
```


附：字符串处理函数的编写实例

```
int strlen (char s [])  
{  
    int i =0;  
    while (s[i]!='\0')  
        i++;  
    return(i);  
}
```

```
void strcat (char s [],cahr t[])  
{ int i =0,j=0;  
    while (s[i]!='\0') /* find end of s */  
        i++;  
    while ((s[i]=t[j])!='\0'))  
        {i++, j++;}}
```

第9次实验练习 (必做题)

本次必做3道题，在第11周末之前提交源码和运行结果

第1题：子串模式匹配。给定两个字符串sA和sB：长字符串sA长度不超过30，可能由字母/数字/符号等任意字符构成，但不含空格、换行符；模式字符串sB长度不超过sA的长度，可能包含除空格和换行符外的任意字符。现要求在长字符串sA中查找匹配模式字符串sB，请你找到sB在sA中出现的所有位置。注意：模式字符串sB中含有一个或若干个特殊字符'?'，在匹配过程中，每个'?'可以匹配sA中的任意一个字符，而sB中的其他非'?'字符必须与sA中匹配的子串完全相同。

要求：输入两行，第一行为长字符串sA，第二行为模式字符串sB；输出：sB在sA中出现的所有位置（用若干非负整数表示），sA的起始位置从0开始计算；如果没有找到任何匹配，输出"No match found"。

样例：输入：abcdefghc*exyzcferpk（回车）c?e（回车）

输出：2 8 14

第2题：元音替换。对于一个由a-z（小写）组成的字符串，将其中的元音反转，而辅音不反转。如对于字符串“hello”，替换后的字符为“holle”。

注：元音为a, e, i, o, u

测试样例：输入一个由a-z（小写）组成的字符串，输出反转后的字符串

例如：输入：hello；输出：holle

提示：本题通过字符串数组和循环分支能够解决。参考算法：首先遍历字符串，将其中所有的元音存入字符串数组中，之后再次遍历字符串，将其中所有的元音反序替换。

第3题：输入一段由英文字母（区分大小写）组成的字符串，将其按ASCII码从大到小顺序输出。测试样例：输入一个由英文字母（区分大小写）组成的字符串，输出符合要求的字符串。例如：输入：aggregate；输出：trgggeeeaa

参考算法：本题通过字符串数组、循环分支能够求解。算法可以为冒泡排序。若同学不会冒泡排序，则根据出现的字符仅在英文字母范围，遍历法寻找字符串中从z到A的字母，也可求得。

选做题

第1题：从键盘输入两个字符串，判断它们是否属于异位字符串。所谓异位字符串，就是把一个字符串中字母的顺序改变，得到的字符串。例如：s = "anagram", t = "nagaram", 是异位字符串。s = "rat", t = "car", 不是异位字符串。为了简化，假设字符串只包含小写英文字母。

要求：如果是异位字符串输出1，否则输出0。

第2题：在网络编程时，经常需要把IP地址转换计算机内部的整型数来处理。C++系统提供的atoi()就是实现该功能。参考该函数，编写另一个函数（如aton()），其功能是将输入IPV4地址（字符串，例如166.111.64.89）字符串转换成32位整数输出。

输入参数：str, 输入字符串

返回值：转换结果, 若str无法转换成整数, 返回0

函数申明：int aton(const char str[]);

程序员写的信

....你就像越界的数组和未清理的野指针，难以预料。而我对你态度，就像const变量一样永不改变。我到底做错了什么，你知不知道我的心里也正因使用穷举法寻找解决方案而导致资源耗尽而死机？.....。