

字符串与字符串函数

张晓平

武汉大学数学与统计学院

homepage: xpzhang.me

2018 年 5 月 18 日

目录

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

① 字符串表示

② 字符串输入

③ 字符串输出

④ 自定义字符串输入/输出函数

⑤ 字符串函数

字符串表示

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- 字符串是以空字符结尾的 `char` 数组，故关于数组与指针的知识也适用于字符串。
- 由于字符串的使用非常广泛，C 提供了很多专为字符串设计的函数。

字符串表示：字符串常量

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

字符串常量是指被一对双引号括起来的任何字符。

```
char greeting[50] = "How " "are" "you?";
```

等价于

```
char greeting[50] = "Howareyou?";
```

字符串表示：字符串常量

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

若想在字符串中使用双引号，可在双引号前加反斜杠 \ 。

```
printf("\\"Ready, go!\"\\\"exclaimed John.");
```

输出结果为

```
"Ready, go!" exclaimed John.
```

字符串表示：字符串常量

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

字符串常量属于静态存储 (static storage) 类。

所谓静态存储是指如果在一个函数中使用字符串常量，即使多次调用了这个函数，该字符串在程序的运行过程中只存储一份。

整个引号中的内容作为指向该字符存储位置的指针。

字符串表示：字符串常量

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // quotes.c:
2 #include <stdio.h>
3 int main(void)
4 {
5     printf("%s, %p, %c\n",
6           "We", "are", *"space farers");
7     return 0;
8 }
```

字符串表示：字符串常量

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // quotes.c:
2 #include <stdio.h>
3 int main(void)
4 {
5     printf("%s, %p, %c\n",
6             "We", "are", *"space_farers");
7     return 0;
8 }
```

输出结果为

We, 0x100000f81, s

字符串表示：字符串数组及其初始化

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

初始化方式一：指定一个足够大的数组来容纳字符串。

```
char s[40] = "Hello world!"
```

字符串表示：字符串数组及其初始化

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

初始化方式一：指定一个足够大的数组来容纳字符串。

```
char s[40] = "Hello world!"
```

这相比于标准的数组初始化要简洁很多：

```
char s[] = {'H', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', '!', '\0'};
```

如果没有空字符，则得到的是一个字符数组而不是一个字符串。

字符串表示：字符串数组及其初始化

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

指定数组大小时，要确保数组长度比字符串长度至少多 1，未被初始化的元素均被自动设置为空字符。

字符串表示：字符串数组及其初始化

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

初始化方式二：省略数组大小，让编译器决定。

```
char s[] = "Hello_world!"
```

字符串表示：字符串数组及其初始化

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

初始化方式二：省略数组大小，让编译器决定。

```
char s[] = "Hello_world!"
```

初始化方式三：使用指针符号建立字符串。

```
char * s = "Hello_world!"
```

字符串表示：字符串数组及其初始化

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

初始化方式二：省略数组大小，让编译器决定。

```
char s[] = "Hello_world!"
```

初始化方式三：使用指针符号建立字符串。

```
char * s = "Hello_world!"
```

这两种方式都声明 `s` 是一个指向给定字符串的指针。

字符串表示：数组与指针的差别

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

考虑如下两个声明：

```
char heart[] = "I love C!";  
char * head = "I love math!";
```

主要的差别在于数组名 `heart` 是个常量，而指针 `head` 则是个变量。

字符串表示：数组与指针的差别

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

(1)、两者都可以使用数组符号。

```
for (i = 0; i < 6; i++)  
    putchar(heart[i]);  
putchar('\n');  
for (i = 0; i < 6; i++)  
    putchar(head[i]);  
putchar('\n');
```

输出结果为

```
I love  
I love
```


字符串表示：数组与指针的差别

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

(2)、两者都可以使用指针加法。

```
for (i = 0; i < 6; i++)  
    putchar(*(heart + i));  
putchar('\n');  
for (i = 0; i < 6; i++)  
    putchar(*(head + i));  
putchar('\n');
```

输出结果仍为

```
I love  
I love
```

字符串表示：数组与指针的差别

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

(3)、但只有指针可以使用增量运算符。

```
while (*head != '\0')  
    putchar(*(head++));
```

输出结果

```
I love math!
```

字符串表示：数组与指针的差别

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

(4)、可让 head 指针指向 heart 数组的首元素，即

```
head = heart;
```

但不能这么做：

```
heart = head;
```

字符串表示：数组与指针的差别

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

(5)、可改变 heart 数组中的信息。

```
heart[7] = 'R';    OR    *(heart + 7) = 'R';
```

因数组元素是变量，但数组名是常量。

字符串表示：数组与指针的差别

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

(5)、可改变 heart 数组中的信息。

```
heart[7] = 'R';    OR    *(heart + 7) = 'R';
```

因数组元素是变量，但数组名是常量。

字符串表示：字符串数组

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
const char * fruit[4] = {  
    "Apple",  
    "Pear",  
    "Orange",  
    "Peach"  
};
```

该语句声明了一个长度为 4 的数组 `fruit`，每个元素都是指向 `char` 的指针。

- 第一个指针是 `fruit[0]`，它指向第一个字符串的第一个字符；
- 第二个指针是 `fruit[1]`，它指向第二个字符串的第一个字符；
- ...

字符串输入：创建存储空间

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

在读入字符串之前，必须分配足够大的存储区来存放读入的字符串。**不要指望计算机读取字符串时自动计算其长度，然后为其分配空间。**例如，如果你这么做

```
char * name;  
scanf("%s", name);
```

虽然编译会通过，但后果可能会非常严重，因 `name` 可能指向任何地方。

字符串输入：创建存储空间

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

创建存储空间，有两种方式：

- 最简单的方法就是在声明中明确指出数组大小：

```
char name[81];
```

- 另一种方式就是使用 C 库中分配存储空间的函数，如 `malloc()`。

字符串输入：创建存储空间

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

为字符串预留空间后，就可以读取字符串了。C 库提供了三种读取字符串的函数：

- `scanf()`
- `gets()`
- `fgets()`

字符串输入： gets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

函数 gets() 从键盘获取一个字符串，其函数原型为

```
char * gets ( char * str );
```

它读取换行符之前（不包括换行符）的所有字符，在这些字符后添加一个空字符，然后把这个字符串交给调用它的程序。

它将读取换行符并将其丢弃，这样下一次读取就会在新的一行开始。

字符串输入： gets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // name1.c
2 #include <stdio.h>
3 #define MAX 81
4 int main(void)
5 {
6     char name[MAX];
7     printf("Hi, what's your name?\n");
8     gets(name);
9     printf("Nice name, %s.\n", name);
10    return 0;
11 }
```

字符串输入: gets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Hi, what's your name?  
warning: this program uses gets(), which  
is unsafe.  
Ming Li  
Nice name, Ming Li.
```

字符串输入：gets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Hi, what's your name?  
warning: this program uses gets(), which  
is unsafe.  
Ming Li  
Nice name, Ming Li.
```

由于 `gets()` 不检查目标数组是否能够容纳输入，故编译器会给出警告，提醒用户使用 `gets()` 会不安全。

字符串输入: gets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // name2.c:
2 #include <stdio.h>
3 #define MAX 81
4 int main(void)
5 {
6     char name[MAX];
7     char * ptr;
8     printf("Hi, what's your name?\n");
9     ptr = gets(name);
10    printf("%s? Ah! %s!\n", name, ptr);
11    return 0;
12 }
```

字符串输入：gets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Hi, what's your name?
```

```
warning: this program uses gets(), which  
is unsafe.
```

```
Xiaoping Zhang
```

```
Xiaoping Zhang? Ah! Xiaoping Zhang!
```

字符串输入：fgets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

由于 `gets()` 不检查目标数组是否能够容纳输入，多出来的字符简单溢出到相邻的内存区。`fgets()` 改进了该不足，它让用户指定最大读入字符数。

`fgets()` 的函数原型为

```
char * fgets (char * str, int num, FILE * stream);
```


字符串输入：fgets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

gets()与 fgets()的三个不同：

- ① fgets()用第二个参数来说明最大读入字符数。
若该参数值为 n ，则 fgets()会读取最多 $n-1$ 个字符或读完第一个换行符为止。
- ② 若 fgets()读取换行符，并将它存到字符串中，而 gets()读取并丢弃它。
- ③ fgets()用第三个参数来说明读哪一个文件，其中 `FILE *` 表示文件指针，第 13 章会专门介绍。从键盘输入时，可以使用 `stdin` 作为该参数。

字符串输入: fgets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // name3.c
2 #include <stdio.h>
3 #define MAX 81
4 int main(void)
5 {
6     char name[MAX];
7     char * ptr;
8     printf("Hi, what's your name?\n");
9     ptr = fgets(name, MAX, stdin);
10    printf("%s? Ah! %s!\n", name, ptr);
11    return 0;
12 }
```

字符串输入：gets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Hi, what's your name?  
Michael Jordan  
Michael Jordan  
? Ah! Michael Jordan  
!
```

字符串输入：gets()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Hi, what's your name?  
Michael Jordan  
Michael Jordan  
? Ah! Michael Jordan  
!
```

该显示表明了 `fgets()` 的不足，原因在于 `fgets()` 把换行符存储到字符串中，导致每次显示字符串时就会显示换行符。

字符串输入：scanf()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

可用带 %s 的 scanf() 来读取一个字符串。

scanf() 与 gets() 的主要差别在于字符串何时结束。

scanf() 更基于获取单词而不是获取字符串，而 gets() 会读取所有字符，直到遇到第一个换行符为止。

字符串输入：scanf()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

scanf()使用以下两种方法决定输入何时结束。(但无论哪种方法，都从第一个非空白字符开始读取。)

- ① 若使用占位符 %s ， 字符串读到（但不包括）下一个空白字符。
- ② 若指定字段宽度，如 %10s ， 则 scanf()会读入 10 个字符或直到遇到第一个空白字符。

字符串输入：scanf()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

输入语句	原始输入队列	name 中的内容	剩余队列
<code>scanf("%s", name);</code>	Fleebert Hup	Fleebert	Hup
<code>scanf("%5s", name);</code>	Fleebert Hup	Fleeb	ert Hup
<code>scanf("%5s", name);</code>	Ann Ular	Ann	Ular

字符串输入：scanf()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // scan_str.c:
2 #include <stdio.h>
3 int main(void)
4 {
5     char name1[11], name2[11];
6     int count;
7     printf("Please enter 2 names.\n");
8     count = scanf("%5s%10s", name1, name2);
9     printf("I read the %d names %s and %s.\n",
10           count, name1, name2);
11     return 0;
12 }
```


字符串输入：scanf()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Please enter 2 names.  
Jesse Jukes  
I read the 2 names Jesse and Jukes.
```

```
Please enter 2 names.  
Liza Applebottham  
I read the 2 names Liza and Applebotth.
```

```
Please enter 2 names.  
Portensia Callowit  
I read the 2 names Porte and nsia.
```

字符串输出：puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

C 有三个用于输出字符串的标准库函数：

- puts()
- fputs()
- printf()

字符串输出: puts() |

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // put_out.c:
2 #include <stdio.h>
3 #define DEF "I am a #defined string."
4 int main(void)
5 {
6     char str1[80] = "An array was
    initialized to me.";
7     const char * str2 = "A pointer was
    initialized to me.";
8
9     puts("I'm an argument to puts().");
10    puts(DEF);
11    puts(str1);
12    puts(str2);
13    puts(&str1[5]);
```

字符串输出：puts() II

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
14 puts(str2+4);  
15  
16 return 0;  
17 }
```

字符串输出：puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
I'm an argument to puts().  
I am a #defined string.  
An array was initialized to me.  
A pointer was initialized to me.  
ray was initialized to me.  
inter was initialized to me.
```

字符串输出：puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- 使用 puts(), 只需给出字符串参数的地址。
- 与 printf()不同, puts()显示字符串时会自动在其后添加一个换行符。
- puts()遇到空字符时便会停止输出, 故应确保有空字符存在。

字符串输出：puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
char str1[80] = "An array was initialized  
to me.";  
...  
puts(&str1[5]);
```

输出结果为

```
ray was initialized to me.
```

字符串输出：puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
char str1[80] = "An array was initialized  
to me.";  
...  
puts(&str1[5]);
```

输出结果为

```
ray was initialized to me.
```

因 `&str1[5]` 是数组 `str1` 第 6 个元素的地址，该元素为字符 `'r'`，也正是 `puts()` 输出字符串的起点。

字符串输出: puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
const char * str2 = "A pointer was  
initialized to me.";  
...  
puts(str2+4);
```

输出结果为

```
inter was initialized to me.
```

字符串输出：puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
const char * str2 = "A pointer was  
initialized to me."  
...  
puts(str2+4);
```

输出结果为

```
inter was initialized to me.
```

str2+4 指向包含第一个 'i' 字符的内存单元， puts() 输出字符串从这里开始。

字符串输出: puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // nono.c:
2 #include <stdio.h>
3 int main(void)
4 {
5     char side_a[] = "Side A";
6     char dont[] = {'W', 'O', 'W', '!' };
7     char side_b[] = "Side B";
8
9     puts(dont);
10
11     return 0;
12 }
```

字符串输出： puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // nono.c:
2 #include <stdio.h>
3 int main(void)
4 {
5     char side_a[] = "Side A";
6     char dont[] = {'W', 'O', 'W', '!' };
7     char side_b[] = "Side B";
8
9     puts(dont);
10
11     return 0;
12 }
```

不要效仿该程序！

字符串输出：puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

输出结果为

```
WOW!Side A
```

字符串输出：puts()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

输出结果为

```
WOW!Side A
```

dont 缺少空字符，不是字符串，于是 puts() 就不知道该在哪里停止。它会一直输出内存中 dont 后面的字符，直到发现一个空字符。

从运行结果来看，在内存中 side_a 数组存储在 dont 数组之后。

字符串输出：fputs()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

fputs()是 puts()的面向文件版本。两者的主要区别是：

- fputs()需要第二个参数来说明要写的文件。可用 stdout 作为参数来进行输出显示，stdout 在 stdio.h 中定义。
- 与 puts()不同， fputs()并不为输出自动添加换行符。

字符串输出： fputs()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

gets() 丢掉输入的换行符，而 puts() 为输出添加换行符。

假定写一个循环，读取一行并把它回显在下一行，可这么写：

```
char line[81];  
while (gets(line))  
    puts(line);
```


字符串输出：fputs()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

fgets() 存储输入的换行符，而 fputs() 不会为输出添加换行符。

故以上程序也可写成

```
char line[81];  
while (fgets(line, 81, stdin))  
    fputs(line, stdout);
```

以上两段代码中，line 数组中的字符都被显示在单独的一行。

字符串输出：fputs()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

注意： puts()是为和 gets()一起使用而设计的，
而 fputs()是为和 fgets()一起使用而设计的。

字符串输出：printf()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

输出字符串时，`printf()`不如`puts()`使用方便，但其可以格式化多种数据类型，因而更通用。

```
printf("%s\n", string);
```

等效于

```
puts(string);
```

但前者更简洁。

字符串输出：printf()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

不过，想在一行上输出多个字符串时， `printf()` 更为简单。
如

```
printf("Well, %s, %s\n", name, MSG);
```

自定义字符串输入/输出函数

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

不一定非要使用标准 C 库的函数进行输入输出。如果你不喜欢用它们，可以自行定义，在 `getchar()` 和 `putchar()` 的基础上设计自己的函数。

自定义字符串输入/输出函数

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

假定你想设计一个类似 `puts()` 但并不自动添加换行符的函数，可以这么做

```
void put1(const char * string)
{
    while (*string != '\0')
        putchar(*string++);
}
```

自定义字符串输入/输出函数

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

假定你想设计一个类似 `puts()` 的函数，并且还可以统计输出的字符个数，可以这么做

```
int put2(const char * string)
{
    int count = 0;
    while (*string)
    {
        putchar(*string++);
        count++;
    }
    putchar('\n');

    return(count);
}
```

自定义字符串输入/输出函数

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
#include <stdio.h>
void put1(const char *);
int put2(const char *);
int main(void)
{
    put1("If I'd as much money");
    put1(" as I could spend,\n");
    printf("I count %d characters.\n",
           put2("I never would cry old
                chairs to mend.));

    return 0;
}
```


自定义字符串输入/输出函数

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
If I'd as much money as I could spend,  
I never would cry old chairs to mend.  
I count 37 characters.
```

字符串函数

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

C 库提供了许多处理字符串的函数，在 `string.h` 中给出其函数原型。

- ① `strlen()`
- ② `strcat()`
- ③ `strncat()`
- ④ `strcmp()`
- ⑤ `strncmp()`
- ⑥ `strcpy()`
- ⑦ `strncpy()`

字符串函数: strlen()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

- 函数原型

```
int * strlen(const char * s);
```

- 功能

返回字符串 `s` 的长度，即 `s` 中的字符数（不包含空字符）。

字符串函数: strlen() |

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // test_fit.c
2 #include <stdio.h>
3 #include <string.h>
4 void fit(char *, unsigned int);
5 int main(void)
6 {
7     char mesg[] = "Hold on to your hats,
8     hackers.";
9     puts(mesg);
10    fit(mesg, 7);
11    puts(mesg);
12    puts("Let's look at some more of the
13    string.");
14    puts(mesg + 8);
```

字符串函数: strlen() II

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
14
15     return 0;
16 }
17
18 void fit(char *string, unsigned int size)
19 {
20     if (strlen(string) > size)
21         *(string + size) = '\0';
22 }
```

字符串函数：strlen()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Hold on to your hats, hackers.
```

```
Hold on
```

```
Let's look at some more of the string.  
to your hats, hackers.
```

字符串函数：strcat()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- 函数原型

```
char * strcat(char * s1,  
               const char * s2);
```

- 功能

把字符串 s2 （包括空字符）追加到字符串 s1 的结尾，字符串 s2 的第一个字符覆盖字符串 s1 中的空字符。

字符串函数：strcat()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- s1 成为一个新的字符串， s2 没有改变。
- strcat() 为 `char *` 类型，返回 s1 。

字符串函数: strcat() I

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 #include <stdio.h>
2 #include <string.h>
3 #define SIZE 80
4 int main(void)
5 {
6     char flower[SIZE];
7     char addon[] = "s smell like old shoes."
8     ;
9     puts("What is your favorite flower?");
10    gets(flower);
11    strcat(flower, addon);
12    puts(flower);
13    puts(addon);
14
```

字符串函数：strcat() II

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
15 |     return 0;  
16 | }
```

字符串函数：strcat()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
What is your favorite flower?  
Rose  
Roses smell like old shoes.  
s smell like old shoes.
```

字符串函数：strcat()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

`strcat()`并不检查第一个数组是否能够容纳第二个字符串。如果没有为第一个数组分配足够大的空间，多出的字符溢出到相邻单元时就会出现问题的。

该问题可通过使用 `strncat()` 加以解决，该函数需要另一个参数来指明最多允许添加的字符数目。

字符串函数：strncat()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- 函数原型

```
char * strcat(char * s1,  
               const char * s2,  
               size_t n);
```

- 功能

把字符串 s2 的前 n 个字符或直到空字符为止的字符追加到字符串 s1 的结尾，s2 的第一个字符覆盖 s1 中的空字符，总在最后添加一个空字符。

- strncat() 为 char * 类型，返回 s1 。

字符串函数: strncat() I

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 #include <stdio.h>
2 #include <string.h>
3 #define SIZE 30
4 #define BUGSIZE 4
5 int main(void)
6 {
7     char flower[SIZE];
8     char addon[] = "s smell like old shoes."
9     ;
10    char bug[BUGSIZE];
11    int available;
12
13    puts("What is your favorite flower?");
14    gets(flower);
```

字符串函数: strncat() II

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
14  if ((strlen(addon) + strlen(flower) + 1)
    <= SIZE)
15      strcat(flower, addon);
16  puts(flower);
17
18  puts("What is your favorite bug?");
19  gets(bug);
20  available = BUGSIZE - strlen(bug) - 1;
21  strncat(bug, "is sdfsd", 1);
22  puts(bug);
23
24  return 0;
25 }
```

字符串函数：strncat()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
What is your favorite flower?  
Rose  
Roses smell like old shoes.  
What is your favorite bug?  
Aphid  
Aphids smell
```


字符串函数：strcmp()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- 函数原型

```
int strcmp(const char * s1,  
           const char * s2);
```

- 功能

比较字符串 `s1` 和 `s2` 。若字符串相同，则返回 0 ；否则就比较两个字符串的第一个不匹配的字符对（使用 ASCII 码进行比较）。

字符串函数：strcmp()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- 若第一个字符串小于第二个则返回一个负数；
- 若第一个字符串较大就返回一个整数。

字符串函数: strcmp() I

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 /* compare.c -- this will work */
2 #include <stdio.h>
3 #include <string.h>
4 #define ANSWER "Grant "
5 #define MAX 40
6 int main(void)
7 {
8     char try[MAX];
9     puts("Who is buried in Grant's tomb?");
10    gets(try);
11    while (strcmp(try,ANSWER) != 0)
12    {
13        puts("No, that's wrong. Try again.");
14        gets(try);
15    }
```

字符串函数：strcmp() II

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
16 puts("That's right!");  
17 return 0;  
18 }
```

字符串函数：strcmp()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

我的系统的输出结果为

```
strcmp("A", "A") is 0
strcmp("A", "B") is -1
strcmp("B", "A") is 1
strcmp("C", "A") is 2
strcmp("Z", "a") is -7
strcmp("apples", "apple") is 115
```

字符串函数: strcmp()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

而有些系统输出结果为

```
strcmp("A", "A") is 0
strcmp("A", "B") is -1
strcmp("B", "A") is 1
strcmp("C", "A") is 1
strcmp("Z", "a") is -1
strcmp("apples", "apple") is 1
```

字符串函数：strncmp()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

- 函数原型

```
int strncmp(const char * s1,  
            const char * s2,  
            size_t n);
```

- 功能

比较字符串 `s1` 和 `s2` 的前 `n` 个字符或直到第一个空字符为止。返回结果与 `strcmp()` 类似。

字符串函数：strncmp()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

如果想搜索以 "astro"开头的字符串，可以限定搜索前 5 个字符。

字符串函数: strncmp() I

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 /* starsrch.c -- use strncmp() */
2 #include <stdio.h>
3 #include <string.h>
4 #define LISTSIZE 5
5 int main(void)
6 {
7     const char * list[LISTSIZE] = {
8         "astronomy", "astounding",
9         "astrophysics", "ostracize",
10        "asterism"
11    };
12    int count = 0;
13    int i;
14
15    for (i = 0; i < LISTSIZE; i++)
```

字符串函数: strncmp() II

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
16         if (strncmp(list[i], "astro", 5) ==  
17             0)  
18         {  
19             printf("Found: %s\n", list[i]);  
20             count++;  
21         }  
22     printf("The list contained %d words  
23     beginning  
24         " with astro.\n", count);  
25     return 0;  
26 }
```

字符串函数：strncmp()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Found: astronomy
```

```
Found: astrophysics
```

```
The list contained 2 words beginning with  
astro.
```

字符串函数: strcpy()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- 函数原型

```
char * strcpy(char * s1,  
               const char * s2);
```

- 功能

把字符串 s2 （包括空字符）复制到 s1 指向的位置，
返回 s1 。

字符串函数: strcpy()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- s2 称为源 (source) 字符串, s1 称为目标 (target) 字符串。
- 指针 s2 可以是一个已声明的指针、数组名或字符串常量。
- 指针 s1 应指向空间大到足够容纳字符串 s2 的数组。

谨记: 声明一个数组将为数据分配存储空间, 而声明一个指针值为一个地址分配存储空间。

字符串函数: strcpy() I

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 /* copy1.c -- strcpy() demo */
2 #include <stdio.h>
3 #include <string.h>
4 #define SIZE 40
5 #define LIM 5
6 int main(void)
7 {
8     char qwords[LIM][SIZE];
9     char temp[SIZE];
10    int i = 0;
11
12    printf("Enter %d words beginning with q
13    :\n", LIM);
14    while (i < LIM && gets(temp))
15    {
```

字符串函数: strcpy() II

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
15     if (temp[0] != 'q')
16         printf("%s doesn't begin with q!\n",
                temp);
17     else {
18         strcpy(qwords[i], temp);
19         i++;
20     }
21 }
22 puts("Here are the words accepted:");
23 for (i = 0; i < LIM; i++)
24     puts(qwords[i]);
25 return 0;
26 }
```

字符串函数: strcpy()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Enter 5 words beginning with q:
```

```
quit
```

```
quarter
```

```
quite
```

```
quotient
```

```
nomore
```

```
nomore doesn't begin with q!
```

```
quiz
```

```
Here are the words accepted:
```

```
quit
```

```
quarter
```

```
quite
```

```
quotient
```

```
quiz
```


字符串函数：strcpy()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

strcpy()还有两个重要的属性：

- 它是 `char *` 类型，返回第一个参数的值；
- 第一个参数不需要指向数组的开始，这样就可以复制到目标字符串的指定位置。

字符串函数: strcpy() I

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 #include <stdio.h>
2 #include <string.h>
3 #define WORDS "beast"
4 #define SIZE 40
5
6 int main(void)
7 {
8     const char * orig = WORDS;
9     char copy[SIZE] = "Be the best that you
10    can be.";
11
12     char * ps;
13
14     puts(orig);
15     puts(copy);
```

字符串函数: strcpy() II

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
15  ps = strcpy(copy + 7, orig);  
16  
17  puts(copy);  
18  puts(ps+6);  
19  
20  return 0;  
21 }
```

字符串函数：strcpy()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
beast
```

```
Be the best that you can be.
```

```
Be the beast
```

```
beast
```

字符串函数: strncpy()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- 函数原型

```
char * strncpy(char * s1, const  
char * s2,  
                size_t n);
```

- 功能

把字符串 s2 的前 n 个字符或直到空字符为止的字符复制到 s1 指向的位置，第三个参数 n 用于指明最大可复制的字符数。

字符串函数：strncpy()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

- 若源字符串的字符数小于 n ，则整个字符串都被复制过来，包括空字符；
- 复制的字符数不能超过 n ，必须要给空字符留位置。处于这个原因，调用该函数时， n 一般设置为目标数组长度减 1。
- 函数返回 $s1$ 。

字符串函数: strncpy() I

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 // copy3.c
2 #include <stdio.h>
3 #include <string.h>
4 #define SIZE 40
5 #define TARGSIZE 7
6 #define LIM 5
7 int main(void)
8 {
9     char qwords[LIM][TARGSIZE];
10    char temp[SIZE];
11    int i = 0;
12
13    printf("Enter %d words begin with q:\n",
14           LIM);
15    while (i < LIM && gets(temp)) {
```

字符串函数: strncpy() II

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
15     if (temp[0] != 'q')
16         printf("%s doesn't begin with q!\n",
            temp);
17     else {
18         strncpy(qwords[i], temp, TARGSIZE -
            1);
19         qwords[i][TARGSIZE - 1] = '\0';
20         i++;
21     }
22 }
23
24 puts("Here are the words accepted:");
25 for (i = 0; i < LIM; i++)
26     puts(qwords[i]);
27
```


字符串函数：strncpy() III

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

28
29

```
return 0;  
}
```

字符串函数: strncpy()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Enter 5 words begin with q:
```

```
quack
```

```
quadratic
```

```
quisling
```

```
quota
```

```
quagga
```

```
Here are the words accepted:
```

```
quack
```

```
quadra
```

```
quisli
```

```
quota
```

```
quagga
```

字符串函数：sprintf()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

sprintf() 在 `stdio.h` 中声明。

- 作用同 `printf()` 一样，但它写到字符串中而不是输出显示。
- 第一个参数是目标字符串的地址，其余参数同 `printf()`。

字符串函数: sprintf() I

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串输入/输出函数

字符串函数

```
1 #include <stdio.h>
2 #define MAX 20
3 int main(void)
4 {
5     char first[MAX];
6     char last[MAX];
7     char formal[2*MAX+10];
8     double prize;
9
10    puts("Enter your first name:");
11    gets(first);
12    puts("Enter your last name:");
13    gets(last);
14    puts("Enter your prize money:");
15    scanf("%lf", &prize);
```

字符串函数: sprintf() II

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
16    sprintf(formal, "%s, %-19s: $%6.2f\n",  
17                last, first, prize);  
18    puts(formal);  
19  
20    return 0;  
21 }
```

字符串函数: sprintf()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Enter your first name:
warning: this program uses gets(), which
is unsafe.
Teddy
Enter your last name:
Bear
Enter your prize money:
2000
Bear, Teddy                : $2000.00
```

字符串函数: sprintf()

字符串与字符串函数

张晓平

目录

字符串表示

字符串输入

字符串输出

自定义字符串
输入/输出函数

字符串函数

```
Enter your first name:
warning: this program uses gets(), which
is unsafe.
Teddy
Enter your last name:
Bear
Enter your prize money:
2000
Bear, Teddy                : $2000.00
```

sprintf() 获取输入，并把输入格式化为标准形式后存放在字符串 format 中。