

1 字符串

C语言字符串简介

字符串实际上就是以null字符'\0'结尾的一维数组。'\0'是系统自动添加作为该字符串结束的标识符。

例：字符串"hello"总共有五个字符，但实际上是占据了六个字节。

C语言中，字符变量是用char定义的，但是C中没有储存字符串的数据类型。所以C中一般通过字符数组和字符指针来存储。

字符数组输入和输出：

scanf: 函数有%c和%s两种类型。

- 1. %c可以将空格还有换行符输入，
- 2. %
- 3. 结束，
- 4. %s输入不用加&。

getchar()、putchar()单字符输入输出。有时候多次输入之间需换行的话，有时候需要多加一个getchar()来吸收回车符。

gets()和puts()用来输入和输出一行字符串，以换行符作为输入结束。输出的时候会自动紧跟一个换行符。

注意：字符数组末尾是以'\0'结尾的，scanf的%s和gets会自动在末尾添加'\0'，但%c,getchar则不会,所以要注意手动添加。

1.1 操作字符串的函数

	函数	作用
1	strcpy(a,b)	复制字符串b至字符串a
2	strcat(a,b)	将字符串b连接至a的尾部
3	strlen(a)	返回字符串a的长度
4	strcmp(a,b)	如果a=b，则返回0 如果a<b，则返回小于0 如果a>b，则返回大于0
5	strchr(a,'s') strrchr(a,'s')	返回的是一个指针；strchr是从左至右指向字符's'在字符串a中第一次出现的位置 strrchr则是从右至左指向字符's'在字符串a中第一次出现的位置 如果's'不存在则返回NULL
6	strstr(a,b)	返回一个指针指向字符串b在a中第一次出现的位置

下面对部分函数进行进一步的解释：

1.1.1 strlen函数和sizeof的区别

两者虽然都可以计算字符串的长度，但是区别还是相当的大。

一：从用处方面来说

strlen是专门针对计算字符串的函数，而sizeof作为单目运算符，它的参数可以是数组、指针、函数等等

返回长度

strlen作为函数，它遇到'\0'结束返回字符串的长度，但是sizeof则是把结束符'\0'计算在内。

例：

```
char s1[] = "abcdefghijklmn";

printf("strlen函数返回的字符长度是%d\n",strlen(s1));

printf("sizeof返回的字符长度是%d\n",sizeof(s1));
```

输出：

```
strlen函数返回的字符长度是15
sizeof返回的字符长度是16
```

返回类型

strlen函数返回的是size_t 类型（即无符号整型），所以在一些条件判断中使用要格外的小心

例：

```
if(strlen(a)-strlen(b)>=0)
```

因为返回的是unsigned int型，所以if条件句里永远是真

1.1.2 strchr 函数

上面表格中已经介绍了，strchr函数返回的是一个指针；

```
char s2[] = "3asdadasdas";

char *q = strchr(s2,'d');

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

输出： dasdasdas

从上面可以看出它是从左至右开始查找目的字符，若查找成功返回的值是从目的字符向右到结束的字符串。

1.1.3 strrchr函数

它是从右至左开始查找目的字符，返回的同样是从目的字符向右到结束的字符串。

```
char s2[] = "3asdadasdas";

char *q = strrchr(s2,'d');

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

输出： das

1.1.4 strstr函数

strstr函数同样返回的是一个指针，strstr查找的是字符串

例：

```
char s2[] = "3asdadasdas";

char *q1 = strstr(s2,"das");

printf("%s\n",q1);

输出： dasdasdas
```

1.1.5 gets()/puts() 数组函数

gets可以把空格一块输入，puts输出完的时候自动跟一个换行符。

sscanf&sprintf

```
char str[100] = "203200:132.12,nice!",str2[100],str3[100];

int n;

double db;

//sscanf是把字符串数组str里面的内容赋值给其他的部分

sscanf(str,"%d:%lf,%s",&n,&db,&str2);

//sprintf是把其他的内容赋值给字符串数组str3

sprintf(str3,,"%d:%lf,%s",&n,&db,&str2);
```

2 常量指针与指针常量的区别

三个名词虽然非常绕嘴，不过说的非常准确。用**中国话**的语义分析就可以很方便地把三个概念区分开。

1.const

const是constant的简写，只要一个变量前面用const来修饰，就意味着该变量里的数据可以被访问，不能被修改。也就是说const意味着“只读”。任何修改该变量的尝试都会导致编译错误。const是通过编译器在编译的时候执行检查来确保实现的（也就是说const类型的变量不能改是编译错误，不是运行时错误。）所以我们只要想办法骗过编译器，就可以修改const定义的常量，而运行时不会报错。

规则：

- const离谁近，谁就不能被修改；
- 因为常量在定义以后就不能被修改，所以使用const定义变量时必须初始化。

```
#include <stdio.h>
```

```
int main(void)
{
    int i = 10;
    int j = 20;
    const int *ptr = &i;

    printf("ptr: %d\n", *ptr);
    *ptr = 100;          /* error: object pointed cannot be modified using the
pointer ptr */

    ptr = &j;            /* valid */
    printf("ptr: %d\n", *ptr);
    return 0;
}
```

2.const point

声明指针时，可以在类型前或后使用关键字const，也可在两个位置都使用。三种定义形式如下：

常量指针

```
- const int *ptr;
- int const *ptr;
```

指针常量

```
int *const p2
```

加深记忆记住三句话：

指针和 const 谁在前先读谁；
*象征着地址，const象征着内容；
谁在前面谁就不允许改变。

例如：

```
int const *p1 = &b; //const 在前，定义为常量指针
int *const p2 = &c; // *在前，定义为指针常量
```

常量指针是指指向常量的指针，顾名思义，就是指针指向的是常量，即，它不能指向变量，它指向的内容不能被改变，不能通过指针来修改它指向的内容，但是指针自身不是常量，它自身的值可以改变，从而指向另一个常量。

指针常量是指指针本身是常量。它指向的地址是不可改变的，但地址里的内容可以通过指针改变。它指向的地址将伴其一生，直到生命周期结束。有一点需要注意的是，指针常量在定义时必须同时赋初值。

案例1：指针b指向的地址内容不能修改，而指针b可以指向其他地址。

```
#改变指针b指向地址的内容
int main()
{
    int a = 2;
    int const *b = &a;
    *b = 3;          //报错: expression must be a modifiable lvalue
    printf("albert:%d\n",a);
}
```

```

}
-----
#改变指针b的指向
int main()
{
    int a = 2;
    int b = 3;
    int const *c = &a;
    printf("albert:%p\n", c);
    c = &b;
    printf("albert:%p\n",c);
}

```

案例2: 指针指向的地址不可以重新赋值, 但内容可以改变。

```

int main()
{
    int a = 2;
    int b = 3;
    int *const c = &a;
    printf("albert:%p\n", c);
    c = &b; //报错: expression must be a modifiable lvalue
    printf("albert:%p\n",c);
}
-----
int main()
{
    int a = 2;
    int b = 3;
    int *const c = &a;
    *c = 4;
    printf("albert:%d\n",*c); //4
}

```

实例:



```

#include <stdio.h>

int main()
{
    int a[] = { 1,5,10,20 };
    int b = *a++; // *a++ 等同于 *(a++)
    printf("b = %d\n",b);
}

```

如上所示, 编译报错: 由于数组名是常量指针, 所以不能执行a++, 进行修改。

 **E0137** expression must be a modifiable lvalue
 **C2105** '++' needs l-value

