字符串

字符数组

- char word[] = {‘H’, ‘e’, ‘l’, ‘l’, ‘o’,‘!’};

word[0]	H
word[1]	e
word[2]	
word[3]	l
word[4]	o
word[5]	!

这不是C语言的字符串，因为不能用字符串的方式做计算

字符串

- char word[] = {‘H’, ‘e’, ‘l’, ‘l’, ‘o’,‘!’,’\0’};

word[0]	H
word[1]	e
word[2]	l
word[3]	l
word[4]	o
word[5]	!
word[6]	\0

字符串

- 以0（整数0）结尾的一串字符
  - 0或'\0'是一样的，但是和'0'不同
- 0标志字符串的结束，但它不是字符串的一部分
  - 计算字符串长度的时候不包含这个0
- 字符串以数组的形式存在，以数组或指针的形式访问
  - 更多的是以指针的形式
- string.h 里有很多处理字符串的函数

字符串变量

- char *str = “Hello”;
- char word[] = “Hello”;
- char line[10] = “Hello”;

字符串常量

- “Hello”
- ”Hello” 会被编译器变成一个字符数组放在某处，这个数组的长度是6，结尾还有表示结束的0
- 两个相邻的字符串常量会被自动连接起来
- 行末的\表示下一行还是这个字符串常量

字符串

- C语言的字符串是以字符数组的形态存在的
  - 不能用运算符对字符串做运算
  - 通过数组的方式可以遍历字符串
- 唯一特殊的地方是字符串字面量可以用来初始化字符数组
- 以及标准库提供了一系列字符串函数

字符串常量

```
char* s = "Hello, world!";
```

- s 是一个指针，初始化为指向一个字符串常量
  - 由于这个常量所在的地方，所以实际上s是 `const char* s` ，但是由于历史的原因，编译器接受不带const的写法
  - 但是试图对s所指的字符串做写入会导致严重的后果
- 如果需要修改字符串，应该用数组:

```
char s[] = "Hello, world!";
```

指针还是数组?

- char *str = “Hello”;
- char word[] = “Hello”;
  - 数组：这个字符串在这里
    - 作为本地变量空间自动被回收
  - 指针：这个字符串不知道在哪里
    - 处理参数
    - 动态分配空间

- 如果要构造一个字符串—>数组
- 如果要处理一个字符串—>指针

char*是字符串?

- 字符串可以表达为char*的形式
- char*不一定是字符串
  - 本意是指向字符的指针，可能指向的是字符的数组（就像int*一样）
  - 只有它所指的字符数组有结尾的0，才能说它所指的是字符串

字符串运算

字符串赋值?

- char *t = “title”;
- char *s;
- s = t;
- 并没有产生新的字符串，只是让指针s指向了t所指的字符串，对s的任何操作就是对t做的

字符串输入输出

- char string[8];
- scanf(“%s”, string);
- printf(“%s”, string);

- scanf读入一个单词（到空格、tab或回车为止）
- scanf是不安全的，因为不知道要读入的内容的长度

安全的输入

- char string[8];
- scanf(“%7s”, string);

- 在%和s之间的数字表示最多允许读入的字符的数量，这个数字应该比数组的大小小一
  - 下一次scanf从哪里开始?

常见错误

- char *string;
- scanf(“%s”, string);
- 以为char*是字符串类型，定义了一个字符串类型的变量string就可以直接使用了
  - 由于没有对string初始化为0，所以不一定每次运行都出错

空字符串

- char buffer[100]=””;
  - 这是一个空的字符串，buffer[0] == ‘\0’
- char buffer[] = “”;
  - 这个数组的长度只有1!

字符串数组

- char **a
  - a是一个指针，指向另一个指针，那个指针指向一个字符（串）
- char a[][]
  - a是一个二维数组，第二个维度的大小不知道，不能编译
- char a[][10]
  - a是一个二维数组，a[x]是一个char[10]
- char *a[]
  - a是一个一维数组，a[x]是一个char*

```
printf("请输入月份: ");
int month;
scanf("%d", &month);
switch ( month ) {
      case 1: printf("January\n"); break;
      case 2: printf("February\n"); break;
      case 3: printf("March\n"); break;
      case 4: printf("April\n"); break;
      case 5: printf("May\n"); break;
      case 6: printf("June\n"); break;
      case 7: printf("July\n"); break;
      case 8: printf("August\n"); break;
      case 9: printf("September\n"); break;
      case 10:printf("October\n"); break;
      case 11:printf("November\n"); break;
      case 12:printf("December\n"); break;
}
```

今后可以用数组来做

程序参数

- int main(int argc, char const *argv[])
- argv[0]是命令本身
  - 当使用Unix的符号链接时，反映符号链接的名字