

有关于CUDA：CUDA（Compute Unified Device Architecture），是显卡厂商[NVIDIA](https://baike.baidu.com/item/NVIDIA" \t "_blank)推出的运算平台。 CUDA™是一种由NVIDIA推出的通用[并行计算](https://baike.baidu.com/item/%E5%B9%B6%E8%A1%8C%E8%AE%A1%E7%AE%97/113443" \t "_blank)架构，该架构使[GPU](https://baike.baidu.com/item/GPU" \t "_blank)能够解决复杂的计算问题。 它包含了CUDA[指令集架构](https://baike.baidu.com/item/%E6%8C%87%E4%BB%A4%E9%9B%86%E6%9E%B6%E6%9E%84" \t "_blank)（[ISA](https://baike.baidu.com/item/ISA" \t "_blank)）以及GPU内部的并行计算引擎。 开发人员可以使用[C语言](https://baike.baidu.com/item/C%E8%AF%AD%E8%A8%80" \t "_blank)来为CUDA™架构编写程序，所编写出的程序可以在支持CUDA™的处理器上以超高性能运行。CUDA3.

浙江大学翁恺老师C语言网课部分不熟练部分：

可以创建长度为0的数组，int a[0]; 但是这个数组没有意义。

数组的集成初始化：

Int a[ ] = {2,3,4,5,2,535,}

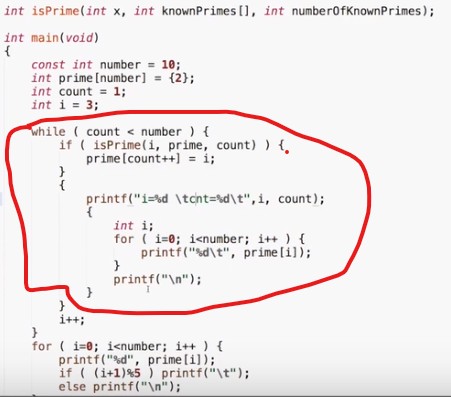
Python中要想知道有关列表长度可以使用len（）

C语言中如果需要知道数组的大小，则需要使用sizeof函数测试其字节长度，让后/单个字节的长度。（sizeof（a）/sizeof（a[0]））

数组变量不可以直接赋值，比如：int a[ ] = {1，2，3，4，57，};

Int b[ ] = a;

这种写法是错误的，如果想把一个数组的内容传递给另一个，必须使用遍历（唯一方法）



这个地方，在函数中单独加入一个大括号是用于调试作用，在小括号内的新变量与括号外面的变量i可以不同，（前提是有一个新的i），作用是插入在函数中间进行调试。

二维数组：

Int a[3][5] 三行五列的矩阵，二维数组的遍历：

For( i=0; i<3;i++){

For( j=0; j<5; j++){

a[i][j] = i\*j ;

}

}

要使用双重循环来进行遍历。

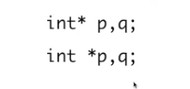
注意二维数组必须要给出列数，可以让计算机计算行数，但是无法让计算机计算列数。

所以一般要给出a【】【5】这样的二维数组。

输出一个地址：printf（“%p”，&i）；

由于C语言的变量是自顶向下堆放的堆栈变量，所以相当于一条通道栈道下来，所以先定义的变量的地址要比后定义的地址大。

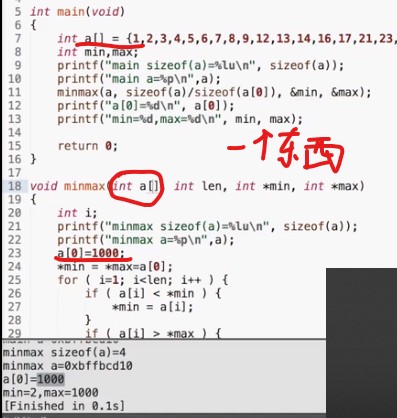
指针：就是保存地址的变量，比如说p指向i，则表明p代表了i的指针，

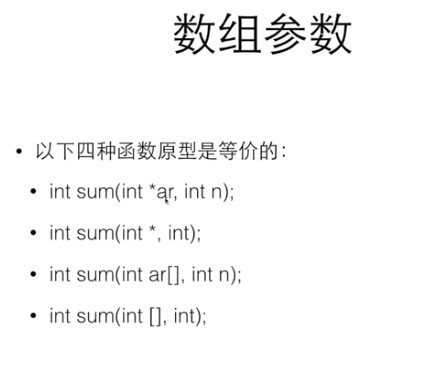
这两种说法都是表明p是一个指针，q不是指针！！！

在利用地址修改一个参数时，可以更改其值。

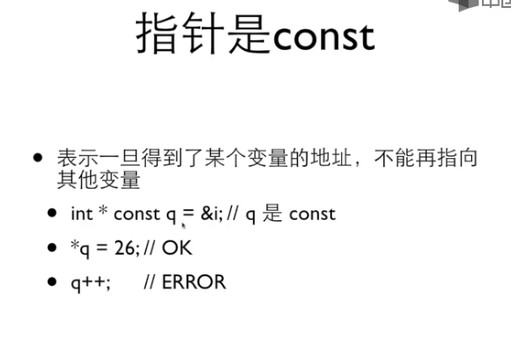
应用场景：返回多个值，例如swap函数，

如果函数里面的参数调用了数组，则此数组调用的其实是数组的指针，

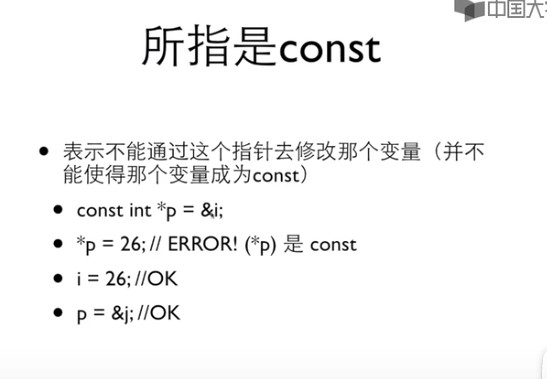
其实不是数组，而是指针

 一和三是一样的表达。

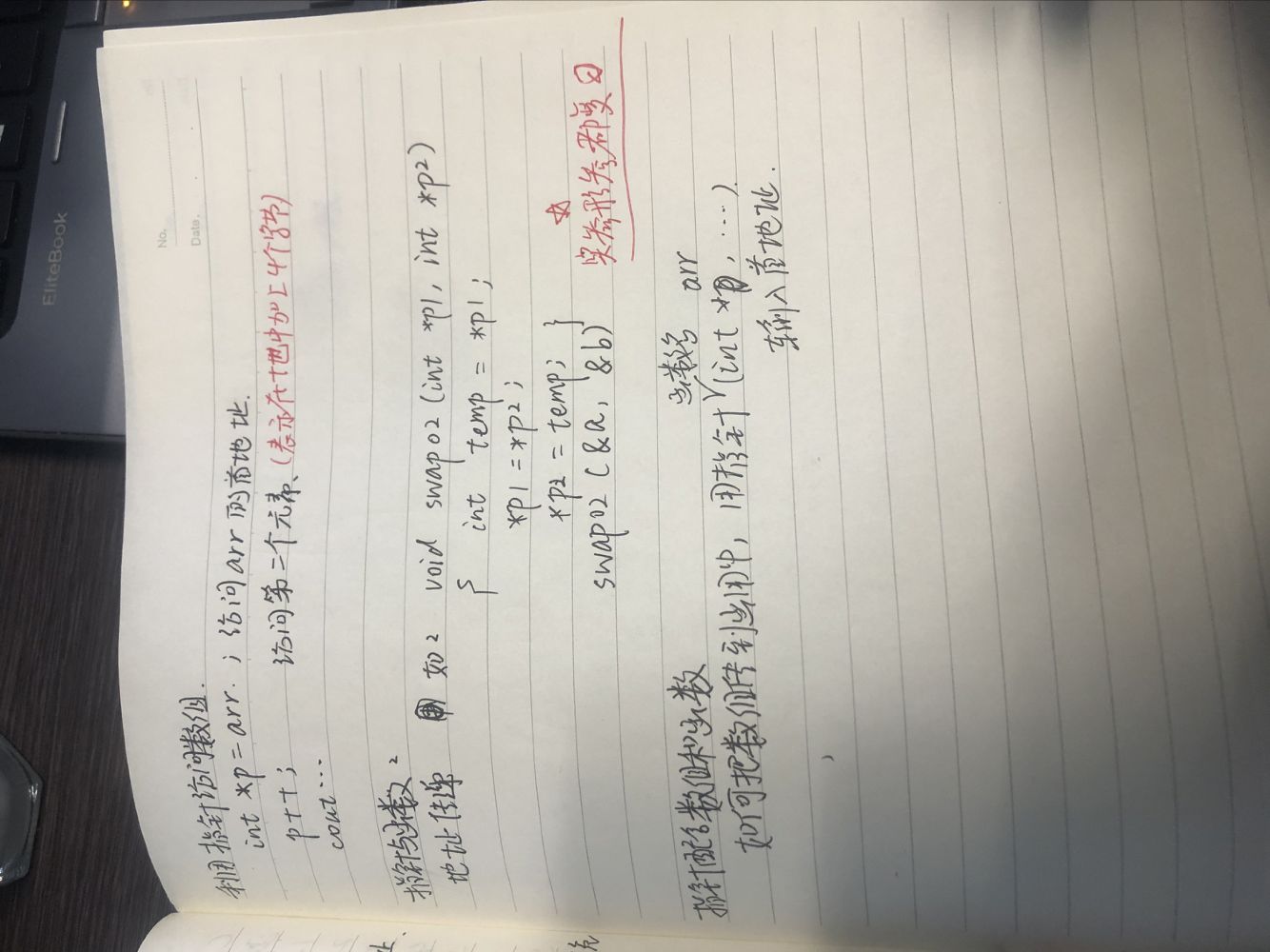
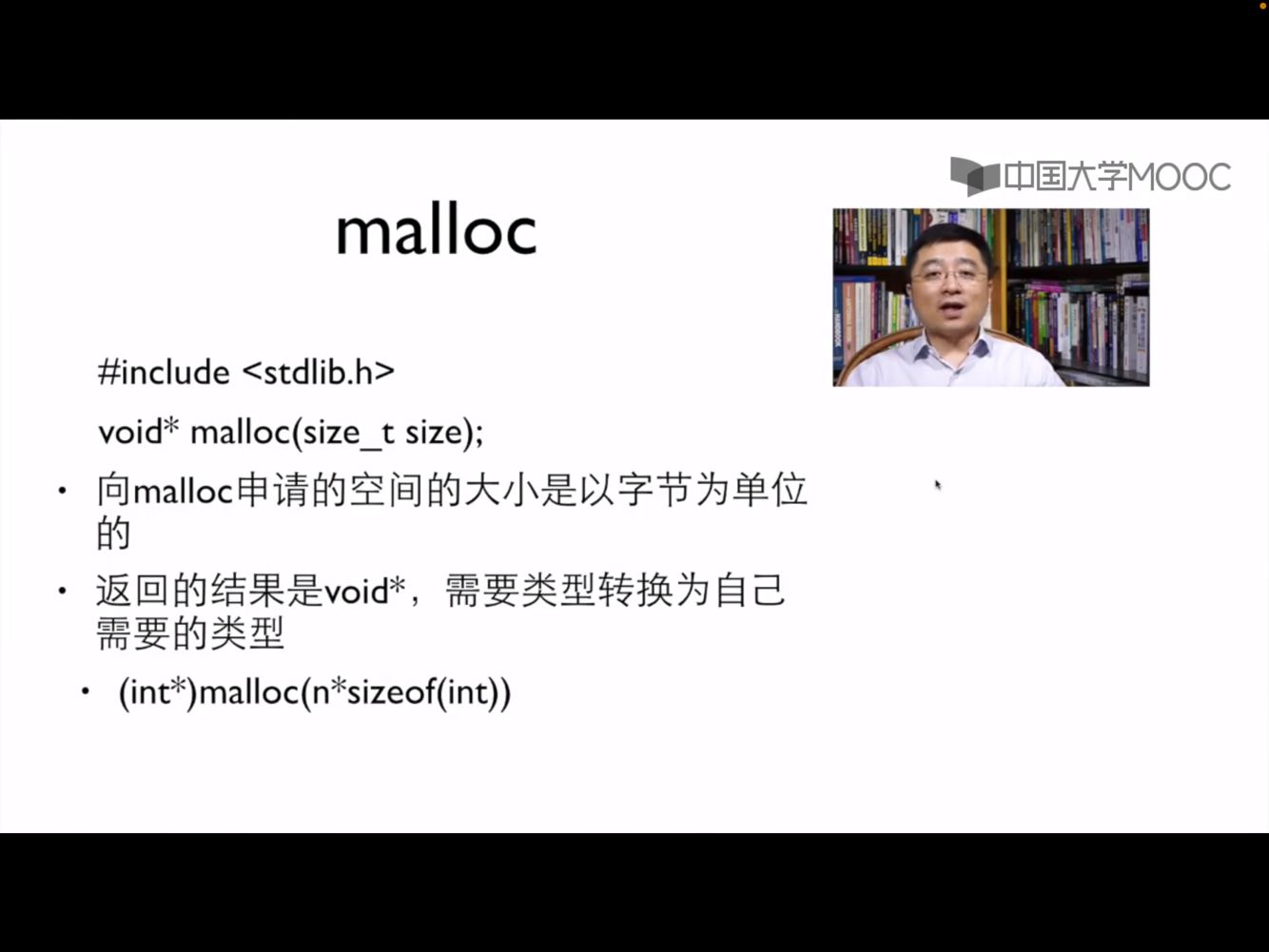
数组变量是特殊的指针，数组变量本身就代表地址，但是对于单个变量取地址要加&，数组指针变量是const类型，他不能被改变。



两种不同形式



指针运算：，原因sizeof（char）=1，sizeof（int）=4，所以指针+1时是加sizeof

指针的动态内存分配

例题：

char s[2][3]={“ab”, “cd”}, \*p=(char \*)s;

那么下列表达式语法正确，并且其值与 s[1][1]相等的表达式（并非一定与其等价）是

A. \*（s+3）

B. p[1][1]

C. \*(p+3)

D. \*++P+2

答案：D

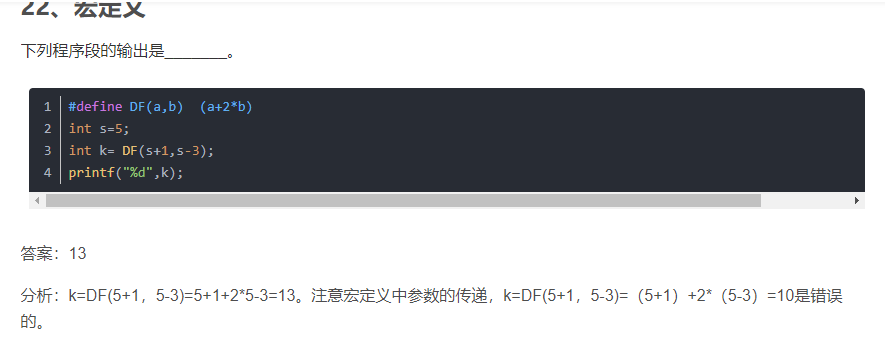
分析：这道题很好。首先字符型指针p存放的是字符串s的首地址，而字符串s表示的是这样一个字符串：

ab\_

cd\_

为了便于理解，“\_”表示空格，即没有元素。题中s[1][1]是d，所以要在ABCD中寻找表示s[1][1]的数。

先来看\*（s+3），s表示数组首地址,不是指针型变量，所以s+3没有意义。p[1][1]同样由于p是指针，不存在这种表达。\*（p+3）是是p表示的指针后移3位，因此\*（p+3）是c。D选项有点复杂，\*++p+2 这个比较有趣,p是一个char\*指针,指向字符数组首地址,&s[0][0]。首先++的优先级更高执行++p得到的是&s[0][1],然后\*运算符解引用,得到s[0][1]，s[0][1]的字符是'b',然后'b'+2,得到的就是'd'的ASCII码.



Static int w；//表示初始化变量，如果不赋初值，则w==0.

\*表示指针，\*\*表示指针的指针。

例如：int \*a；这个语句声明了一个变量a，a的数据类型是int \*，也就是整型变量的指针类型（如果不懂什么是指针，那这个问题就没有意义了）。也就是说 a的值是一个内存地址，在这个地址所在的内存空间中存放的是一个整型变量。再看：int \*\*b；这个语句也声明了一个变量b，b的数据类型是int \*\*，也就是整型变量的指针的指针类型（二级指针）。也就是说 b的值是一个内存地址，该地址所在的内存空间中存放的是一个整型变量的指针（一级指针，或许就是上面那个a的值）。

首先先给大家讲解下什么是指针数组，什么又是数组指针呢？

eg:char \*c[5] 因为[]的优先级高于\*，所以c先与[]结合形成一个数组，类型为char\* 类型，大小为5。里面存放属于char类型的指针。

而对于char(\*c)[5] 因为（）的优先级最高，所以c先和\*形成一个指针，然后再与数组结合，这就形成了数组指针，即为指向数组的指针。它指向包含5个char类型元素的一维数组，此时c的增量以它所指向的一维数组长度为单位；

C语言的字符串，在最后必须占有有‘\0‘=0，但是不是’0‘的字符，所以字符串要比你看到的多一个位置。

字符串已数组的形式存在，以数组或指针的形式访问。

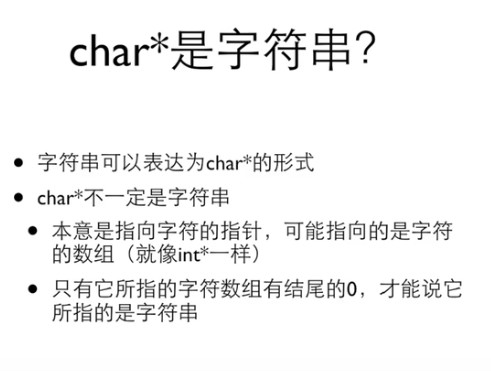
字符串的处理放在string.h 头文件来处理

Char \* s = ‘hello world’；

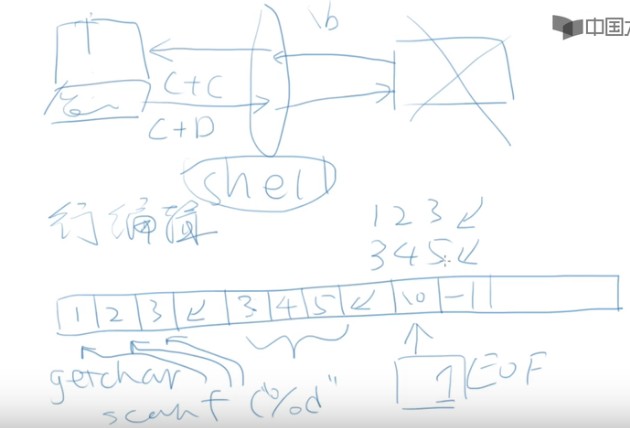
S是一个指针，初始化为指向一个字符串常量

由于这个常量所在的地方，所以实际上s是const char\* s，但是由于历史原因，编译器接受不带const的写法，所以无法对s所指的字符串进行修改。

如果想修改，要定义为数组形式，char s【】 = 。。。。



Scanf（”%7s”，word）；

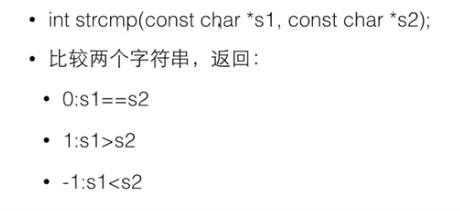
表示输入一个字符串，%7s表示最多输入七个，防止越界

Shell是用户输入和你的程序之间的程序。

字符串的函数：

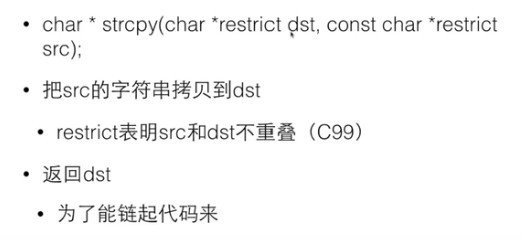


Strlen（）测量字符串的长度，不包含结尾的0

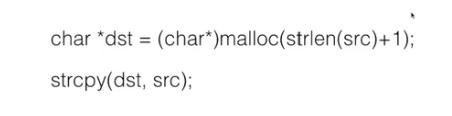
Strcmp（） 比较两个字符串

相比的依据是两个字符串的阿克塞思码谁大谁小，而且会输出差值。

Strcpy（）函数，



只返回第一个字符串，第二个字符串复制到第一个中，也要包含末尾的\0，如果有重复（重叠）则不能使用这个函数



+1和char\*容易遗忘

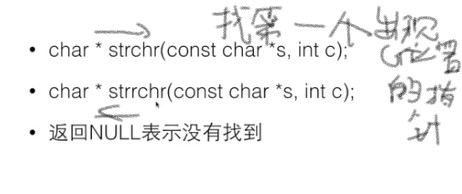
Strcat,把s2拷贝到s1的后面，接成一个长的字符串，返回s1，s1必须具有足够的空间，

Char \*strcat（char\*restrict s1，const char\*restrict）

以上两个函数都存在越界的风险，所以尽量使用其安全版本，即在函数中加个n，



Strncmp，表示指定判断前多少个字符是否相同。



返回的从第一个开始之后的东西，包括

翁恺网课：

1. 定义想函数一样的宏：

宏的名字里面可以有参数： #define cube（x）（（x）\*（x）\*（x）） //定义函数宏

后面调用的时候就写cube（5），则会得到5的立方

易错点：

#define RADTODEG（x）（x\*57.29578）

#define RADTODEG（x）（x）\*57.29578

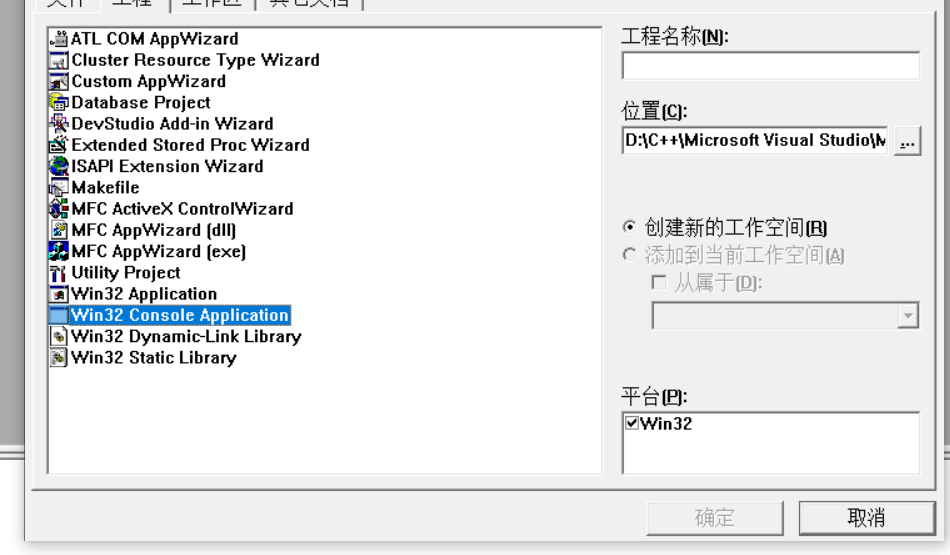
以上两种的定义都不可以，会按照数学运算先计算，所以在运用时一切都要有括号，每个参数都要有括号。

在定义宏的时候后面不要加分号。

而且宏接收到的参数是没有类型定义的，就比如可以输入一个整型，也可以输入一个字符串。

1. 如何合理地分出几个.c文件，使代码冗余量降低

首先，新建一个项目：



然后和你放主函数和功能函数的文件放在同一个文件夹下，就可以互相调用。

在原函数中要保留调用功能（其他功能程序）的声明！！因为如果不声明的话，对于函数输入和输出的类型，编译器并不懂，就会去猜，很容易导致报错。

一个.c文件使一个编译单元，编译器每次编译只能处理一个编译单元。

所以如果不想继续声明的话，就需要把另外一个文件当成头文件进行引用，而且并不是把这个函数整体拿过去，而是把他的声明放在一个新建的.h文件结尾里面。

然后调用的时候#include “max.h” 注意这里不是单名号，是双引号。

“”表示现在当前目录找，如果没找到去编译器指定的目录去找。

<>让编译器旨在指定的目录去找

所以一般系统的库用<>,自己写的用“”

c语音内全局变量的定义不用globe，直接在函数外定义就可以。

如果想要调用在另外一个文件内的.c文件内的全局变量的话：需要在.h头文件内部extern int a。进行一个说明（声明）。

防止重复定义的方法：写在main（）函数里面

#ifndef 函数名（\_\_MAX\_H\_\_）

#define 函数名（\_\_MAX\_H\_\_）

中间程序

#endif

这个是一个判断作用，如果已经定义了，就不会再定义

1. 有关printf和scanf函数格式化的内容：
   1. %d表示整数，其实中间还有很多（%d和%i是一样的，但是%i更灵活）
   2. %-9d -表示左对齐，9表示输出占9个字节，如果不加-会在最右边显示。
   3. +号表示前面加个加号
   4. %09d 表示9个字节，中间0填充
   5. %9.2f表示占9个字节，保留小数点后两位
   6. %n表示输出的字节的个数，%n，在printf中后面变量要加&
   7. \*表示跳过，%[^,]表示到逗号位置的所有内容。
2. 读取文件
   1. fopen r只读，r+打开读写只能从头开始写，w打开只写，w+打开读写，a表示从末尾追加
3. fscanf ， fprintf ， fclose
4. FILE\* fopen（）

\*\*补充：unsigned的在声明时的作用，让数字类型无符号化，比如int的范围是-2^31~2^31，使用unsigned后就变成了0~2^32,相当于往上规整了

1. 按位运算（当成二进制来处理）：//有点偏底层，了解一下，后面如果要深层掌握再说

相余&，二进制数，两个1为1，有0为0，按位相余。

取或 | ，有1就为1

按位取反 ~，从1-0，0-1

取补码 -：10000000000-二进制数

逻辑运算和按位运算的关系：因为计算机逻辑运算时只看到两个值0和1，非0的任何数都当作1，所以逻辑运算相当于把所有非0值都变成1之后按位运算。

i<<j 左移j位，<<1 ,相当于翻两倍，>>1右移1相当于变为原本1/2