内存地址：内存每个字节都有对应的编号，这个编号称为地址。一般用十六进制表示

对指针取sizeof，在32位编译器下恒为4字节，64位编译器下恒为8字节。

指针变量：用来存内存地址的变量。一旦存储了某个变量的地址，就可以说成该指针指向了那个变量

指针通过%p进行输出。打印结果为十六进制。

运算符：

&：取地址运算符，可以取出变量的地址。

***：解引用运算符**，可以拿出指针指向的变量。表达式的值是变量，可以做左值。

指针加减法：

加法：可以给指针加上一个数字，会给指针加上一个指向类型的长度。例如int * pa，给pa+1就是加了一个sizeof(int)。这个动作相当于加上了一个偏移量。

减法：除了加法的方法外，还可以让两个指针相减，得到之间的偏移量。

空指针是指值为0的指针，**野指针**是指指向不确定的指针。

数组是一种特殊的指针，跟指针的区别有：

1、指针可以随意更改指向的位置，数组名不行

2、给数组名取sizeof得到的数组的总大小，而指针恒为4(32位编译器)或8(64位编译器)。

3、数组名的初始化可以用大括号，指针初始化跟普通变量类似

`a[i]` 相当于`*(a+i)`，编译器在解释`a[i]`时，会先解释成`*(a+i)`，再进行下一步的编译。

一个变量定义，去掉标识符就是这个变量的类型。

一个指针定义，去掉标识符和离它最近的星号，就是它指向的变量类型。

指针的作用1：可以通过指针在函数中访问其他函数中的变量

入参：用来参与函数运算的参数，通常传变量本身

出参：用来将结果带出去的参数，通常传变量地址

拿去用用传本身，帮我改改传地址。

指针的作用2：可以通过指针访问堆空间。

动态内存分配函数：

函数声明	函数功能	参数解释	返回值解释	头文件
`void *malloc(size_t size);`	动态分配内存空间	要分配的内存空间的大小	分配的内存空间的首地址，空间不足返回空指针	stdlib.h
`void *calloc(size_t nmemb, size_t size);`	动态分配内存空间，且初始化该空间	nmemb表示要分配的元素个数，size表示每个元素的大小	分配的内存空间的首地址，空间不足返回空指针	stdlib.h
`void *realloc(void *ptr, size_t size);`	重新分配已经分配的内存空间，并继承原空间的数据	ptr是指向已分配内存空间的指针，size表示重新分配的内存空间的大小	新分配的内存空间的首地址，空间不足返回空指针	stdlib.h
`void free(void *ptr);`	释放已经分配的内存空间	ptr是指向已分配内存空间的指针	释放ptr指向的空间	stdlib.h

函数调用逻辑：main可以调用任何函数，自定义函数可以调用除了main外的一切函数。

递归：任何函数都可以调用自身，调用自身的行为称为递归。