# **指针的终极奥义**

## **多重指针（多级指针）**

### **1、什么是多重指针**

**之前有了解过变量、函数等都有其对应的地址，都可以由其对应数据类型的指针变量指向这个地址。那么指针变量也有地址么？**

**指针变量也是有其对应地址的，那么既然有地址，就可以用另一个指针变量指向它的地址，也就是指向指针变量地址的指针，简称指向指针的指针（双重指针/二级指针）。而指向指针的指针也是有地址的，那又可以有指向其地址的指针，这就是多重指针了。**

### **多重指针的定义**

**定义双重指针（二级指针）基本格式：数据类型 \*\*指针变量名;**

**定义三重指针（三级指针）基本格式：数据类型 \*\*\*指针变量名;**

**依次类推：四级指针、五级指针……**

### **多重指针的使用**

**如有以下定义：**

**int a = 10,\*p,\*\*q,\*\*r; //定义整型变量a、指针p、双重指针q、三重指针r**

**就可以有以下赋值语句：**

**p=&a; //使一级指针p指向变量a的地址**

**q=&p; //使双重指针q指向一级指针p的地址**

**r=&q; //使三重指针r指向双重指针q的地址**

**\*p=20; //使用一级指针p给变量a赋值**

**\*\*q=30; //使用二级指针q给变量a赋值**

**\*\*\*r=40; //使用三级指针r给变量a赋值**

### **双重指针指向指针数组**

**如有以下定义：**

**char \*str[6],\*\*p; //定义一个指针数组和一个双重指针**

**p=str; //使双重指针p指向指针数组str的首地址,相当于p=&str[0];**

**p++; //双重指针的移动，相当于p+=1;这时双重指针p就指向str[1]的地址了**

## **动态内存分配与指向它的指针变量**

### **什么是动态内存分配**

**动态内存分配就是使用户可以根据自己的需要，向系统申请所需大小的内存空间；由于没有声明部分来定义它们是为变量的地址还是为数组的地址，所有只能通过指针来引用它们。**

### **2、怎样建立内存的动态分配**

**①使用malloc函数**

**malloc(int size);**

**例如：\*p=malloc(100)**

**用于分配一个大小为size的内存区域**

**②使用calloc函数**

**calloc(unsigned n,int size);**

**例如：\*p=calloc(10,4);**

**用于分配n个大小为size的连续内存区域,可以为一维数组开辟动态内存空间，n为数组元素个数，每个数组元素大小为size。**

**③使用realloc函数**

**realloc(void \*p,unsigned int size);**

**例如：calloc(\*p,12);**

**用于重新分配已经通过malloc函数或calloc函数开辟的内存空间，可以改变其内存空间的大小**

**④使用free函数**

**void free(void \*p);**

**例如：free(p);**

**用于释放指针变量p所指向的动态内存空间，使得这部分空间能被其他变量使用，否则这段内存空间需要等到程序结束后才会被释放。**

**每次使用完动态内存空间的时候需要释放内存空间；**

**（注意：以上函数的声明在stdlib.h头文件中，使用这些函数之前需要包含stdlib.h头文件）**

**开辟的内存空间大小size一般由sizeof(数据类型);来进行计算**

## 结构体指针

### 1、指向结构体变量的指针

**所谓的结构体指针就是指向结构体变量的指针，一个结构体变量的起始地址就是这个结构体变量的指针。如果把一个结构体变量的起始地址存放在一个指针变量中，那么这个指针变量就指向此结构体变量。**

### **结构体指针的定义**

**如有以下结构体：**

**struct student**

**{**

**int id;**

**char \*name;**

**char sex[4];**

**float score;**

**}s1,s2,s3;**

**则可以定义指向struct student类型结构体的指针：**

**struct student \*sp;**

**sp = &s1;//用struct student类型的结构体指针sp指向struct student类型的结构体变量s1**

### **3、通过结构体指针引用结构体变量及其结构体成员**

**指向结构体成员运算符：’->’**

**通过结构体指针引用结构体成员的基本格式：结构体指针名->结构体成员名**

**如：sp->id=100; //通过结构体指针引用结构体成员用指向结构体成员运算符’->’**

**printf(“%s”,sp->name);**

## 内存四区

**在系统为程序开辟内存时，将内存区域划分为4块，分别为：**

**栈区：存放局部变量，由系统自动申请和释放**

**堆区：动态分配内存区域，由自己手动申请和释放，用到指针管理**

**全局静态常量区：存放常量（一般是字符串常量），全局变量和静态变量**

**代码区：存放可执行的代码**