数组指针和指针数组的区别

数组指针（也称行指针）  
定义 int (\*p)[n];  
()优先级高，首先说明p是一个指针，指向一个整型的一维数组，这个一维数组的长度是n，也可以说是p的步长。也就是说执行p+1时，p要跨过n个整型数据的长度。

如要将二维数组赋给一指针，应这样赋值：  
int a[3][4];  
int (\*p)[4]; //该语句是定义一个数组指针，指向含4个元素的一维数组。  
 p=a;        //将该二维数组的首地址赋给p，也就是a[0]或&a[0][0]  
 p++;       //该语句执行过后，也就是p=p+1;p跨过行a[0][]指向了行a[1][]

所以数组指针也称指向一维数组的指针，亦称行指针。

指针数组  
定义 int \*p[n];  
[]优先级高，先与p结合成为一个数组，再由int\*说明这是一个整型指针数组，它有n个指针类型的数组元素。这里执行p+1是错误的，这样赋值也是错误的：p=a；因为p是个不可知的表示，只存在p[0]、p[1]、p[2]...p[n-1],而且它们分别是指针变量可以用来存放变量地址。但可以这样 \*p=a; 这里\*p表示指针数组第一个元素的值，a的首地址的值。  
如要将二维数组赋给一指针数组:  
int \*p[3];  
int a[3][4];  
for(i=0;i<3;i++)  
p[i]=a[i];  
这里int \*p[3] 表示一个一维数组内存放着三个指针变量，分别是p[0]、p[1]、p[2]  
所以要分别赋值。

这样两者的区别就豁然开朗了，数组指针只是一个指针变量，似乎是C语言里专门用来指向二维数组的，它占有内存中一个指针的存储空间。指针数组是多个指针变量，以数组形式存在内存当中，占有多个指针的存储空间。  
还需要说明的一点就是，同时用来指向二维数组时，其引用和用数组名引用都是一样的。  
比如要表示数组中i行j列一个元素：  
\*(p[i]+j)、\*(\*(p+i)+j)、(\*(p+i))[j]、p[i][j]

dasdasdsa