C 语言指针数组（每个元素都是指针）

如果一个数组中的所有元素保存的都是指针，那么我们就称它为指针数组。指针数组的定义形式一般为：

```
dataType *arrayName[length];
```

`[]`的优先级高于`*`，该定义形式应该理解为：

```
dataType *(arrayName[length]);
```

括号里面说明 `arrayName` 是一个数组，包含了 `length` 个元素，括号外面说明每个元素的类型为 `dataType *`。

除了每个元素的数据类型不同，指针数组和普通数组在其他方面都是一样的，下面是一个简单的例子：

```
#include <stdio.h>
int main(){
    int a = 16, b = 932, c = 100;
    //定义一个指针数组
    int *arr[3] = {&a, &b, &c};//也可以不指定长度，直接写作 int *parr[]
    //定义一个指向指针数组的指针
    int **parr = arr;
    printf("%d, %d, %d\n", *arr[0], *arr[1], *arr[2]);
    printf("%d, %d, %d\n", **(parr+0), **(parr+1), **(parr+2));

    return 0;
}
```

运行结果：

16, 932, 100

16, 932, 100

arr 是一个指针数组，它包含了 3 个元素，每个元素都是一个指针，在定义 arr 的同时，我们使用变量 a、b、c 的地址对它进行了初始化，这和普通数组是多么地类似。

parr 是指向数组 arr 的指针，确切地说是指向 arr 第 0 个元素的指针，它的定义形式应该理解为 `int *(*parr)`，括号中的`*`表示 parr 是一个指针，括号外面的 `int *`表示 parr 指向的数据的类型。arr 第 0 个元素的类型为 int *，所以在定义 parr 时要加两个 *。

第一个 printf() 语句中，arr[i] 表示获取第 i 个元素的值，该元素是一个指针，还需要在前面增加一个 * 才能取得它指向的数据，也即 *arr[i] 的形式。

第二个 printf() 语句中，parr+i 表示第 i 个元素的地址，*(parr+i) 表示获取第 i 个元素的值（该元素是一个指针），**(parr+i) 表示获取第 i 个元素指向的数据。

指针数组还可以和字符串数组结合使用，请看下面的例子：

```
#include <stdio.h>
int main(){
    char *str[3] = {
        "c.biancheng.net",
        "C 语言中文网",
        "C Language"
    };
    printf("%s\n%s\n%s\n", str[0], str[1], str[2]);
    return 0;
}
```

运行结果：

c.biancheng.net

C 语言中文网

C Language

需要注意的是，字符数组 str 中存放的是字符串的首地址，不是字符串本身，字符串本身位于其他的内存区域，和字符数组是分开的。

也只有当指针数组中每个元素的类型都是 char *时，才能像上面那样给指针数组赋值，其他类型不行。

为了便于理解，可以将上面的字符串数组改成下面的形式，它们都是等价的。

```
#include <stdio.h>
int main(){
    char *str0 = "c.biancheng.net";
    char *str1 = "C 语言中文网";
    char *str2 = "C Language";
    char *str[3] = {str0, str1, str2};
    printf("%s\n%s\n%s\n", str[0], str[1], str[2]);
    return 0;
}
```