20180402_二维指针和二维数组

其实二维数组名是一个数组指针，那什么是数组指针？数组指针是指向一个数组首地址的指针，它实际上也是一种指针类型，类似于函数指针。

```
int (*pArr)[3]
```

它说明pArr是一个数组指针，它指向的是一个数组元素为int类型并且数组元素的个数为3的一个数组指针，奇怪，中间的怎么还有一个括号是啥玩意？呵呵，这个括号还真是不可少的。少了它就变为另外一种类型了：指针数组。指针数组是数组类型，代表数组的每一个元素是指针类型，它声明如下：int *pArr[3]。

```
#include <bits/stdc++.h>
using namespace std;

void TestFun(int *pArr,int nlength){

}

void TestFun2(void *pArr,int nlength){}

int main(){

    /// 二维数组的定义和初始化
    int iArr[2][3]={0,1,2,3,4,5};

    /// 上面的iArr[0]就是代表第一个数组的首地址，由于二维数组在内存中的存储也是先行后列的方式，所以第二行也紧跟第一行之后，这样就可以用p来访问数组的元素值了，访问的方式有下标和指针方式。
    int * p = iArr[0];

    // 下标和指针两种访问方式
    printf("%d,",p[3]);
    printf("%d\n",*(p+3));


    /// 二维数组指针
    // int **pp = iArr; /// cannot convert 'int (*)[3]' to 'int**'
    int (*pArr)[3] = iArr;
    cout<< *(*(pArr+1) + 2) << endl; // 访问第一行第二列的元素
    cout << *(*(iArr+1) + 2) << endl;// 也可以用数组名来访问

    /// 传递函数参数
```

```
    ///TestFun(iArr,6);         //“TestFun”: 不能将参数 1 从“int [2][3]”转换为“int *”         // 因为数组名是数组指针，而函数的参数是int*，两者的类型化完全不一样，所以不能转换。
    TestFun(&iArr[0][0],6);      // 数组首元素的地址显然是int*类型
    TestFun(iArr[0],6);          // 这样就可以

    // 2-
    TestFun2(iArr,6);
    TestFun2(&iArr[0][0],6);

    return 0;
}
```

附上一个动态创建二维数组的方法：

```
    int **p = new int*[3];  /// 3 行
    for( int i=0; i< 3; i++ )
        p[i] = new int[2];  /// 2 列

    int arr[3][2] = {1,2,3,4,5,6};
    for( int i=0; i< 3; i++ )
        for(int j =0; j < 2; j++){
            p[i][j] = arr[i][j];

        }

    delete[] p;
```