

C语言 06 数组

[一维数组](#)

[二维数组](#)

[多维数组](#)

基本概念

为什么需要数组

解决大量同类型数据的存储和使用问题

模拟现实世界

分类

一维数组

```
int a[25];
```

如何定义一维数组

为n个变量连续分配存储空间

所有的变量数据类型必须相同

所有变量所占的字节大小必须相等

操作

初始化

完全初始化

```
int a[5] = {1,2,3,4,5};
```

不完全初始化，未初始化的元素自动为零

```
int a[5] = {1,2,3};
```

不初始化，所有元素是垃圾值

```
int a[5];
```

清零

```
int a[5] = {0};
```

错误写法

```
int a[5];
```

`a [5] = {1,2,3,4,5};` //只有在定义数组的时候才能整体赋值，其他情况下整体赋值都是错误的。

赋值

把 a 数组的值赋给 b 数组

错误写法：

```
int a[5] = {1,2,3,4,5};
```

```
int b[5];
```

```
b = a;
```

正确写法：

```
for ( i=0; i<5; i++)
```

```
    b[i] = a[i];
```

二维数组

```
int a[3][4];
```

总共是12个元素，可以当做3行4列看待，这12个元素的名字依次是

`a[0][0]` `a[0][1]` `a[0][2]` `a[0][3]`

`a[1][0]` `a[1][1]` `a[1][2]` `a[1][3]`

`a[2][0]` `a[2][1]` `a[2][2]` `a[2][3]`

`a[i][j]` 表示第*i*+1行第*j*+1列的元素

`int a[m][n];` 该二维数组右下角位置的元素只能是`a[m-1][n-1]`

初始化

```
int a[3][4] {
    {1, 2, 3, 4},
    {5, 6, 7, 8},
    {9, 10, 11, 12}
};
```

输出

用多层 for 嵌套

```
int i, j;

for (i = 0; i < 3; i++)
{
    for (j = 0; j < 4; j++)
    {
        printf("%d", a[i][j]);
    };
    printf("\n");
};
```

多维数组

是否存在多维数组

——》不存在

——》因为内存是线性一维的

n维数组可以当做每个元素是n-1维数组的一维数组

比如：int a[3][4];

该数字是含有3个元素的一维数组

但每个元素又含有4个元素