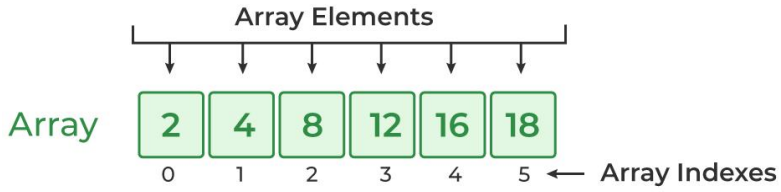
一、数组

（一）什么是数组（Array）？

数组（Array）是一个可以**存放很多相同类型数据的“数据盒子”**，每个小格子用一个“编号”来访问，这个编号就叫**数组下标（Index）**。

就像一个文具盒有很多格子：

- 第 1 格放铅笔
- 第 2 格放橡皮
- 第 3 格放尺子



数组就像是这样的盒子，比如：

```
int box[3] = { 5, 6, 7 };
```

这个数组里放了三个整数，分别是：

位置（下标）	0	1	2
值（内容）	5	6	7

- 注意：C++ 数组的下标是**从 0 开始的**，不是从 1！

☑示例代码：

```
int score[3] = {85, 90, 95};
cout << score[0]; // 输出第一个分数：85
```

考试易错点：

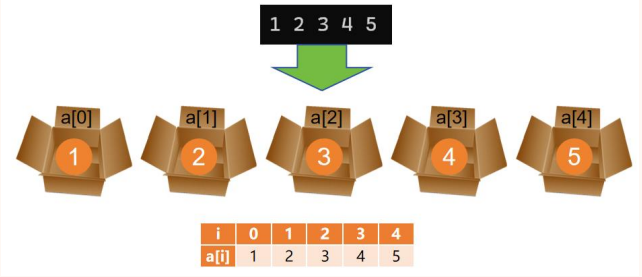
错误点	正确方式
下标从 1 开始	C++下标从 **0** 开始
访问越界	不可以访问不存在的元素，如 score[3]
错写数组名	写 score[i]，不是 score(i)

（二）数组的读入与输出（Input / Output of Array）

☑输入数组（读入）：

你可以使用 for 循环来给数组的每个位置输入值：

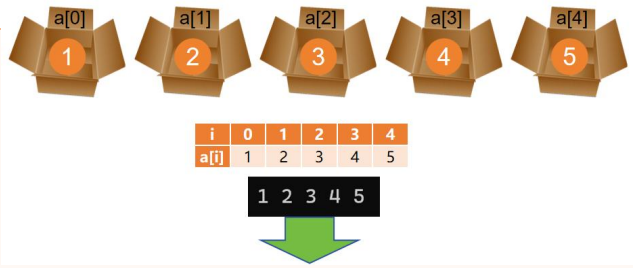
```
int a[5];
for (int i = 0; i < 5; i++) {
    cin >> a[i];
}
```



☑输出数组：

同样用循环来输出每个值：

```
for (int i = 0; i < 5; i++) {
    cout << a[i] << " ";
}
```



类比说明：

想象你在给五个同学发糖果：

- a[0] 是小明的糖
- a[1] 是小红的糖……

你依次记录、再一个个发出来！

考试易错点：

错误类型	正确方法
忘记用循环	输入/输出数组最好用 for 循环
写错下标范围	int a[5] 只能用 a[0] 到 a[4]
忘记初始化	定义数组时最好赋初值或输入赋值

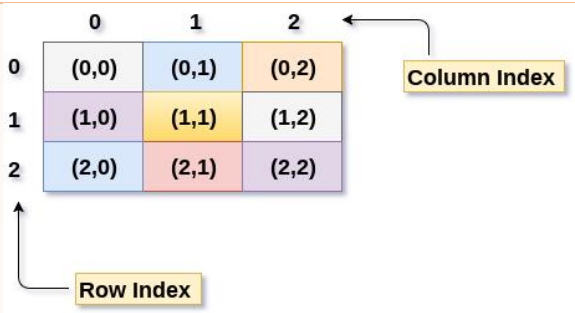
（三）二维数组与多维数组（2D and Multi-dimensional Array）

☑什么是二维数组?

二维数组就像一个**表格**，有**行（row）**和**列（column）**。

定义方式：

```
int grid[2][3] = {
    {1, 2, 3},
    {4, 5, 6}
};
```



这个数组就像：

行 \ 列	0	1	2
0	1	2	3
1	4	5	6

☑ 访问方式：

```
cout << grid[1][2]; // 输出 6（第 2 行第 3 列）
```

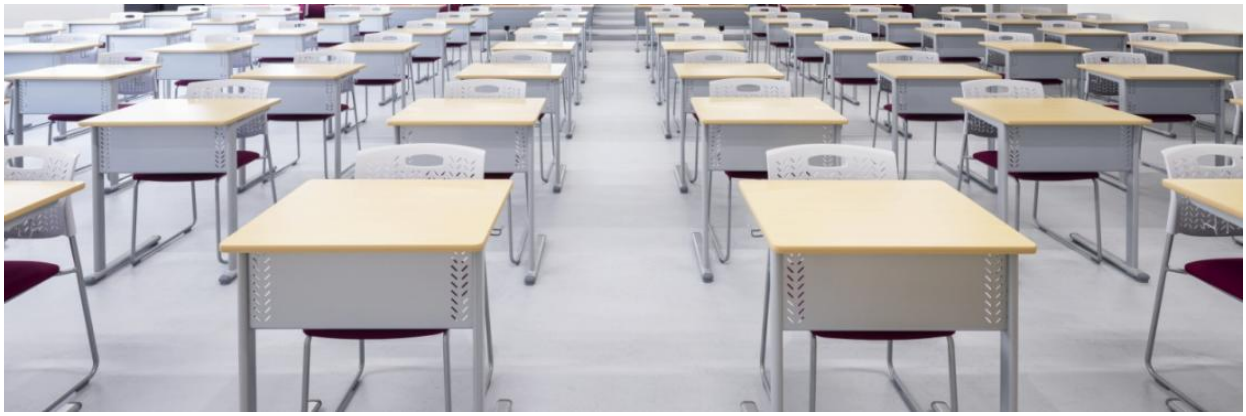
☑ 输入/输出二维数组：

```
int a[2][3];
// 输入
for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 3; j++) {
        cin >> a[i][j];
    }
}
// 输出
for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 3; j++) {
        cout << a[i][j] << " ";
```

```
        }
        cout << endl;
    }
```

举例理解：

二维数组就像教室座位表：

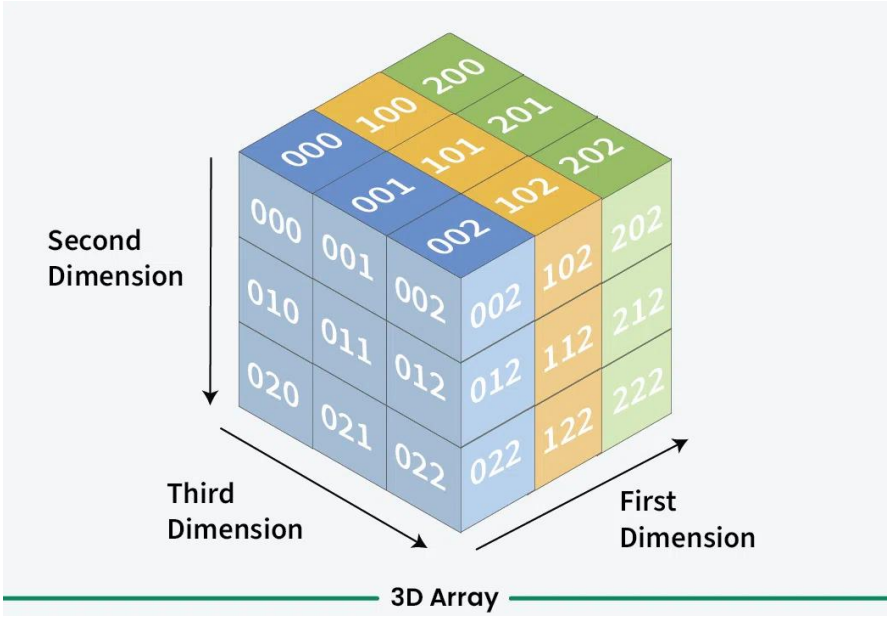


☑多维数组：

可以类比三维魔方或多层楼的座位表：

```
    int cube[2][3][4]; // 2 层楼，每层 3 行，每行 4 列
```

- 一般最多使用二维数组，多维数组理解更复杂，初学掌握二维就足够！



考试易错点：

错误点	正确写法
写成一维下标	必须写 [行][列]，如 a[2][3]
忘记初始化或循环	读写二维数组要用**双重 for 循环**
顺序搞反了	行是第一维，列是第二维（先行后列）