西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

C++程序设计
Programming in C++

1011018

主讲：魏英，计算机学院

数组的定义和使用

- 3、多维数组的定义、初始化
- 4、多维数组的引用

- 1. 多维数组的定义
- C++允许定义多维数组，其定义形式为：

元素类型　数组名[常量表达式1][常量表达式2];

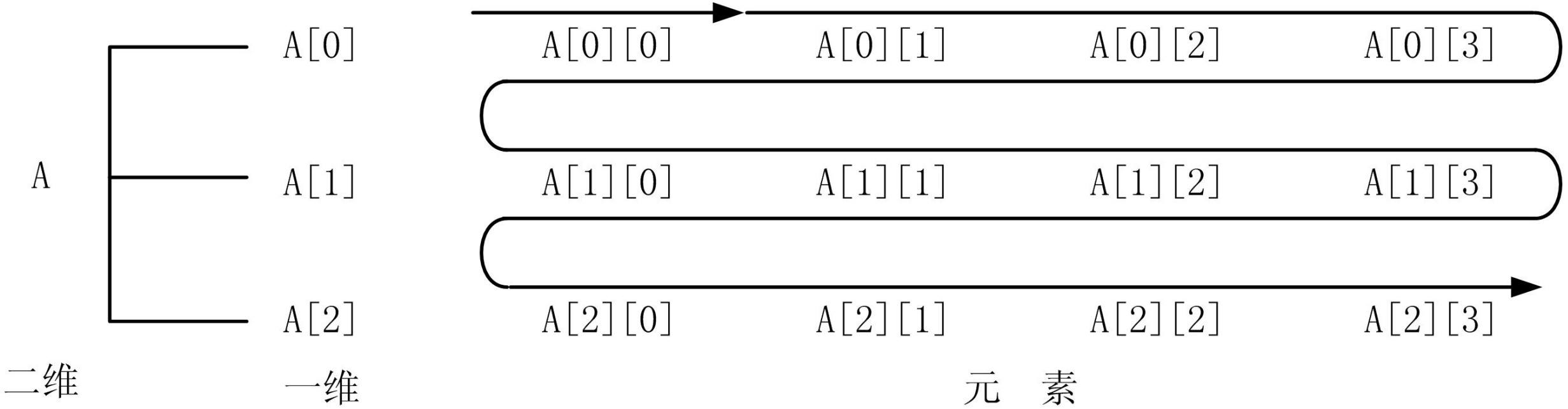
元素类型　数组名[常量表达式1][常量表达式2] ...[常量表达式n] ;

```
int A[3][4]; //定义二维数组
int B[3][4][5]; //定义三维数组
int C[3,4,5,6]; //错误!
```

- 多维定义实际上是反复递归一维定义：即一维数组的每一个元素又是一个一维数组，就构成了二维数组。

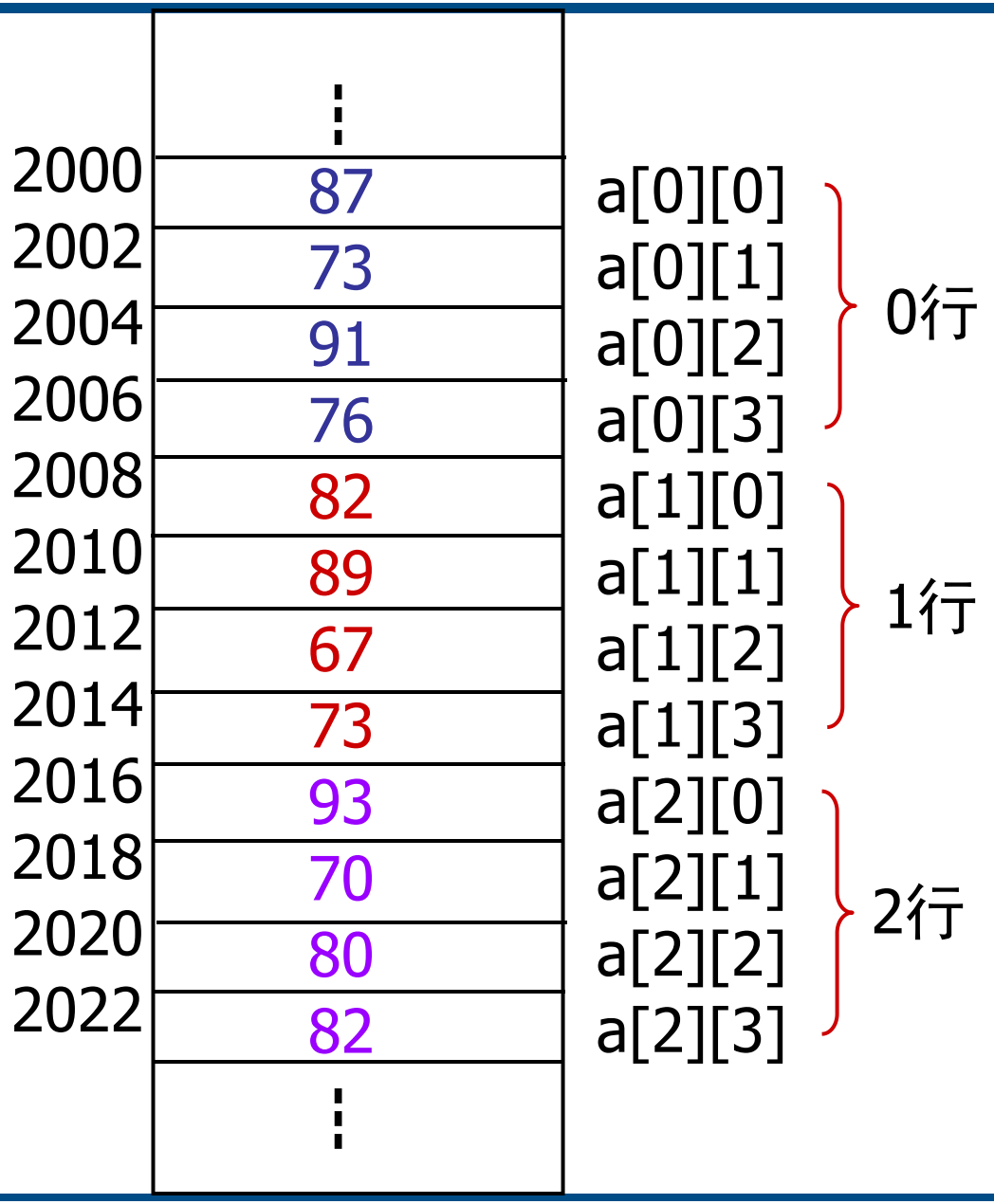
- 本质上，C++的多维数组都是一维数组，这是由内存形式的线性排列决定的。因此，不能按几何中的概念来理解多维，多维数组不过是借用“维”的数学说法表示连续内存单元。

图13.1 二维数组的内存结构



例：int a[3][4];

假设每个元素占2个字节的存储空间，则存储结构图示：



- 2. 多维数组的初始化
- 可以在多维数组定义时对它进行初始化，这里以二维数组来说明，初始化有两种形式。
- （1）初值按多维形式给出：

元素类型　数组名[常量表达式1][常量表达式2] ={{初值列表1},{初值列表2},...};

```
int A[2][3]={ {1,2,3},{4,5,6}};//初值按二维形式
```

- （2）初值按一维形式给出：

```
元素类型    数组名[常量表达式1][常量表达式2] ={初值列表};
```

```
int A[2][3]={ 1,2,3,4,5,6 };//初值按一维形式
```

- 初值列表提供的元素个数不能超过数组长度，但可以小于数组长度。如果初值个数小于数组长度，则只初始化前面的数组元素；剩余元素初始化为0。这个规则两种初始化形式都适用，例如：

```
//只对每行的前若干个元素赋初值
int A[3][4]={{1},{1,2},{1,2,3}};
```

A

	[0]	[1]	[2]	[3]
[0]	1	0	0	0
[1]	1	2	0	0
[2]	1	2	3	0

```
//一维形式部分元素赋初值
int A[3][4]={1,2,3,4,5};
```

A

	[0]	[1]	[2]	[3]
[0]	1	2	3	4
[1]	5	0	0	0
[2]	0	0	0	0

- 在提供了初值列表的前提下，多维数组定义时可以不用指定第1维的数组长度，但其余维的长度必须指定，编译器会根据列出的元素个数自动确定第1维的长度。例如：

```
int A[][2][3]={1,2,3,4,5,6,7,8,9,10,11,12};
//正确
int B[2][][3]={1,2,3,4,5,6,7,8,9,10,11,12};
//错误，只能省略第1维
int C[2][2][]={1,2,3,4,5,6,7,8,9,10,11,12};
//错误，只能省略第1维
```

- 多维数组元素的引用与一维类似，也只能逐个引用数组元素的值而不能一次引用整个数组，引用的一般形式为：

数组名[下标表达式1][下标表达式2] ...[下标表达式n]

```
int b[3][4], i, j, sum=0;
b[0][0]=10;
b[1][2]=b[0][0]+10;
b[2-1][2*2-1]= 5
```

```
int A[3][4]={1,2,3},x;
x = A[0][1]; //x=2
A[2][2] = 50; //则数组A变为
```

A

[0]	1	2	3	0
[1]	0	0	0	0
[2]	0	0	50	0
	[0]	[1]	[2]	[3]

【例13.3】给二维数组输入数据，并以行列形式输出

```
#include <iostream>
using namespace std;
int main()
{
    int A[3][4],i,j;
    for (i=0;i<3;i++)
        for (j=0;j<4;j++) cin >> A[i][j];
    for (i=0;i<3;i++) {
        for (j=0;j<4;j++)  //内循环输出一行
            cout<<A[i][j]<<" ";
        cout<<endl; //每输出一行换行
    }
    return 0;
}
```

【例13.4】求矩阵的转置矩阵

$$A=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{bmatrix} \qquad A^T=\begin{bmatrix}1 & 4\\ 2 & 5\\ 3 & 6\end{bmatrix}$$

```
#include <iostream>
using namespace std;
int main()
{
    int A[2][3]={{1,2,3},{4,5,6}},AT[3][2], i, j;
    for (i=0; i<2; i++) //求矩阵A的转置
      for(j=0;j<3;j++) AT[j][i]=A[i][j];
    cout<<"A="<<endl;
    for (i=0; i<2; i++) { //输出矩阵A
      for(j=0;j<3;j++) cout<<A[i][j]<<" ";
        cout<<endl;
    }
    cout<<"AT="<<endl;
    for (i=0; i<3; i++) { //输出转置矩阵AT
      for(j=0;j<2;j++) cout<<AT[i][j]<<" ";
        cout<<endl;
    }
    return 0;
}
```

C程序设计