Ch-05

数 组

主要内容：

- **一维数组**
- **二维数组**
- **数组作函数参数**
- **字符数组**
- **C++的string类**

5.1 数组

口 定义

数组是具有相同类型的有序数据的集合。

- 数组用**数组名**进行标识；
- 组成数组的数据单元称为**数组元素**，每个数组元素都是一个变量；
- 数组元素在数组中的位置序号称**下标**，C语言中下标从0开始。

口 常用的数组类型

- 一维数组
- 二维数组

5.2 一维数组

5.2.1 一维数组的的定义和引用

口 定义形式：

数据类型 数组名[常量表达式]

口 引用形式：

数组名[下标]

- **说明：**

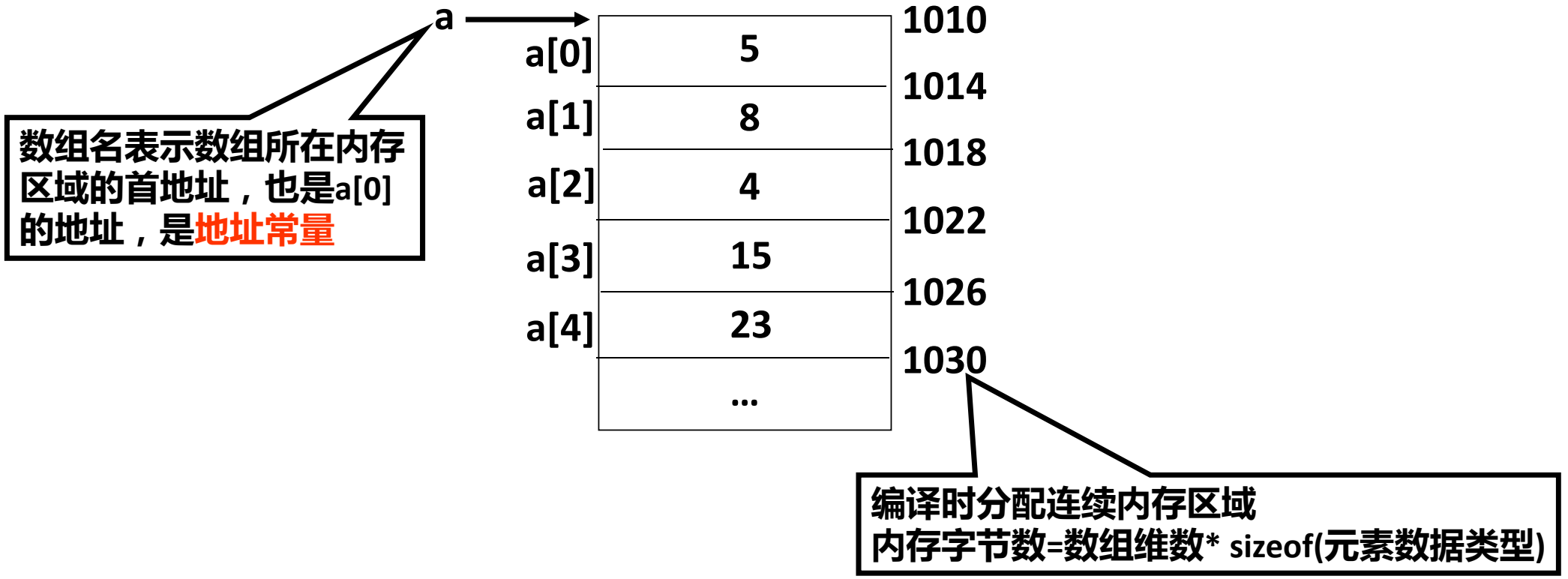
  1、常量表达式的值给出了数组长度，也就是数组元素的个数，因此不能是实数；

  2、下标值从0开始，最大下标值=(数组长度-1)

- **举例：定义整型数组 int a[5];**

  引用数组a中的元素 a[0]、 a[1]、 a[2]、 a[3]、 a[4]。

  - **数组int a[5]的存储示意图**



- 使用数组时要注意一下问题：
  - 定义数组时[ ]中只能是常量表达式，其值只能是整数；
  - 数组必须先定义，后引用；
  - 引用数组元素时，元素下标可以是整型常量或表达式，
    例如: a[1]，a[2*3]；
  - 只能逐个引用数组元素，不能一次引用整个数组；
  - 下标不要超过数组长度范围（编译器不对越界行为进行检查）；
  - 数组元素地址表示：&a[i]，其中&a[0]的值等于a的值；
  - 区分：数组定义int a[10]，数组元素引用t=a[6]。

5.2.2 一维数组的初始化

□ 定义

在定义数组时为数组元素赋初值（在编译阶段使之得到初值）。

□ 说明

- 如果数组不初始化，其元素值为随机数；
- 对static数组元素不赋初值，系统会自动赋以0值。

static int a[5]; 等价于 a[0]=0; a[1]=0; a[2]=0; a[3]=0; a[4]=0;

□ 一维数组初始化的几种形式：

- **在定义数组时对数组元素赋初值**

```
int a[5]={1,2,3,4,5}; 等价于 a[0]=1; a[1]=2; a[2]=3; a[3]=4; a[4]=5;
```

- **只给一部分元素赋值**

```
int a[5]={6,2,3}; 等价于 a[0]=6; a[1]=2;a[2]=3; a[3]=0; a[4]=0;
但是 int a[3]={6,2,3,5,1}; //初值个数超过数组长度，编译错误
```

- **数组元素值全部为0**

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

- **对整个数组元素赋初值时，可以不指定长度。**

```
int a[]={1,2,3,4,5,6}; //编译系统根据初值个数确定数组长度
```

例5.1 求Fibonacci数列

$$\begin{cases} F_1 = 1 & (n=1) \\ F_2 = 1 & (n=1) \\ F_n = F_{n-1} + F_{n-2} & (n \geq 3) \end{cases}$$

```cpp
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    int i;
    int f[20] = {1, 1};

    for(i=2; i<20; i++)
       f[i] = f[i-2] + f[i-1];

    for(i=0; i<20; i++)
    {
        if((i != 0) && (i%5 == 0))
            cout << endl;

        cout << setw(8) << f[i];
    }

    return 0;
}
```

```
       1       1       2       3       5
       8      13      21      34      55
      89     144     233     377     610
     987    1597    2584    4181    6765
```

5.3 二维数组

5.3.1 二维数组的的定义和引用

口 定义形式：

数据类型 数组名[常量表达式1] [常量表达式2]

口 引用形式：

数组名[下标1] [下标2]

- **说明：**
  1、常量表达式1的值表示行的个数，常量表达式2的值表示列的个数，
  二维数组元素的个数=(常量表达式1的值×常量表达式2的值)；
  2、下标值从0开始，最大下标值=(对应常量表达式的值-1)
- **举例：定义整型数组** int b[2][3];
  引用数组b中的元素 b[0][0]、b[0][1]、b[0][2]、b[1][0]、b[1][1]、b[1][2]。

- **二维数组的存储**

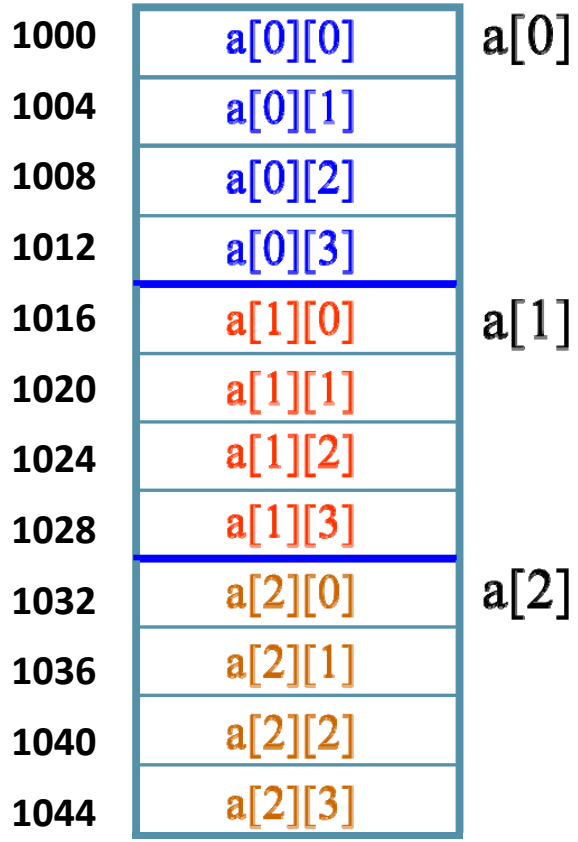
  由于内存是一维的，因此二维数组的各个元素是按照行优先的顺序在内存中连续存放的。

例如: int a[3][4]

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]



5.3.2 二维数组的初始化——定义同时赋初值

□ 形式1：分行初始化

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

此方法较直观, 第一对{ }内的数据赋给第一行数组元素, 依次类推。

□ 形式2：按数据排列顺序对数组元素赋初值

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

将数据依次赋给元素 a[0][0] , a[0][1] ……a[2][3] ，但此方法数据无明显的界限, 当数据较多时容易出错。

□ 形式3：对部分元素赋初值，未赋值元素自动取0

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

1	2	0	0
3	0	0	0
4	5	6	0

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

1	2	3	4
5	6	0	0
0	0	0	0

说明：

1、对数组的全部元素赋初值时可以省略第一维的长度（即行的长度），系统会根据数据的个数和第二维的长度自动求出第一维长度，但维下标不可省。

例如：int b[][2] = {1, 2, 3, 4, 5, 6, 7, 8}; //第一维长度等于4

2、如仅对部分元素赋初值，要想省略数组的行数，则必须分行初始化。

例如：int a[][4]={{1, 2}, {0, 3, 4}, {5}}; //第一维长度等于3

例5.2 找到二维数组中值最大的元素及其行列号

```cpp
#include <iostream>
using namespace std;

int main()
{
    int a[3][4]={{1, 2, 3, 4}, {9, 8, 7, 6}, {-10, 10, -5, 2}};
    int i, j, row=0, colum=0, max=a[0][0];

    for(i=0; i<=2; i++)
        for(j=0; j<=3; j++)
            if(a[i][j] > max)
            {
                max = a[i][j];
                row = i;
                colum = j;
            }

   cout << "max = " << max << endl;
   cout << "row = " << row << endl;
   cout << "colum = " << colum << endl;

   return 0;
}
```

```
max = 10
row = 2
colum = 1
```

5.4 数组作为函数参数

□ 数组元素作函数实参——值传递

例5.3 输入10个整数，求值最大的数和该数的位置。

```
#include <iostream>
using namespace std;

int main()
{
    int max(int, int);
    int a[10], m, n, i;

    cout << "Enter 10 integer numbers:" << endl;
    for( i=0; i<10; i++ ) cin >> a[i];

    for( i=0; i<10; i++ )
    {
        if( max(m, a[i]) > m )
        {
            m = max(m, a[i]);
            n = i;
        }
    }
    cout << "The largest number is " << m << endl;
    cout << "It is the " << n+1 << "th number." << endl;

    return 0;
}
```

```
Enter 10 integer numbers:
1 34 65 5 19 22 7 48 90 105
The largest number is 105
It is the 10th number.
```

□ 数组名作函数参数——地址传递

实参是数组名，形参必须是数组名或指针变量 。

- **说明：**
  - **地址传递；**
  - **形参和实参类型必须一致；**
  - **形参数组长度（多维数组的第一维长度）可不指定，即在数组名面跟一个空的方括号，如a[]；**

    *** C编译系统不检查形参数组大小，因此指定其大小不起作用。**
  - **形参数组名是地址变量。**

例5.4 输入10名学生的成绩，求平均值。

```cpp
#include <iostream>
using namespace std;

int main()
{
    float average(float array[10]);
    float score[10], aver;
    int i;

    cout << "Input 10 scores:" << endl;
    for( i=0; i<10; i++ )
        cin >> score[i];
    cout << endl;

    aver = average( score );
    cout << "Average is: " << aver << endl;

    return 0;
}

float average(float array[])
{
    int i;
    float aver, sum=array[0];
    for( i=1; i<10; i++ )
        sum = sum+array[i];
    aver = sum/10;

    return aver;
}
```

```
Input 10 scores:
54 62 60 75 89 43 96 88 61 70

Average is: 69.8
```

5.5 字符数组

5.5.1 字符数组的定义和初始化

口 定义形式

```
char  数组名[常量表达式]
char  数组名[常量表达式][常量表达式]
```

例如：char c[10], ch[3][4];

口 初始化

- 用逐个元素赋值的方法初始化字符数组
- 用字符串常量初始化字符数组

- **用逐个元素赋值的方法初始化字符数组**

  char ch[5]={‘H’,’e’,’l’,’l’,’o’};

  char ch[5]={‘B’,’o’,’y’}; //ch[3]=‘\0’, ch[4]=‘\0’

  char ch[ ]={‘B’,’o’,’y’}; //编译系统自动判定数组长度等于3

- **用字符串常量初始化字符数组（注意字符串结束标志’\0’的保存）**

  char ch[6]={“Hello”}; 或 char ch[6]=“Hello”; //ch[5]=‘\0’

  char ch[5]=“boy”; //ch[3]=‘\0’, ch[4]=‘\0’

  char ch[]=“boy”; //编译系统自动判定数组长度等于4

  char fruit[][7]={”Apple”, ”Orange”, ”Grape”, ”Pear”, ”Peach”}

5.5.2 字符数组的输入/输出

口 逐个字符的I/O ： 以数组元素作为操作对象；

- **cin会将输入的空白符（空格、回车、Tab）当做有效输入（非空白符）之间的分隔符进而过滤掉；**
- **若输入的字符数量大于数组长度，则缓冲区中的多余字符将作为下一个cin的输入。**

例5.5.1

```
#include <iostream>
using namespace std;

int main()
{
    char str[3];
    int i;

    cout << "cin >> str" << endl;
    for(i=0; i<3; i++)
        cin >> str[i];

    cout << endl << "cout << str" << endl;
    for(i=0; i<3; i++)
        cout << str[i];

    return 0;
}
```

```
cin >> str
a  b
c

cout << str
abc
```

例5.5.2

```
#include <iostream>
using namespace std;

int main()
{
    char str[3], ch[2];
    int i;

    cout << "cin >> str" << endl;
    for(i=0; i<3; i++)
        cin >> str[i];

    cout << endl << "cin >> ch" << endl;
    for(i=0; i<2; i++)
        cin >> ch[i];

    cout << endl << "cout << str" << endl;
    for(i=0; i<3; i++)
        cout << str[i];

    cout << endl << "cout << str" << endl;
    for(i=0; i<2; i++)
        cout << ch[i];

    return 0;
}
```

```
cin >> str
a b cd  e

cin >> ch

cout << str
abc
cout << str
de
```

- 整个字符串的I/O：以数组名作为操作对象。
  - 输入串长度要小于数组维数，否则可能造成数据破坏；
  - 读取输入流时如果遇到空白符，则认为输入结束，并在读取的字符串尾部自动加 ‘\0’。
  - 输出字符串时，遇到第一个’\0’即停止输出。

例5.5.3

```
#include <iostream>
using namespace std;

int main()
{
    char str[3];

    cout << "cin >> str" << endl;
    cin >> str;

    cout << endl << "cout << str" << endl;
    cout << str << endl;

    return 0;
}
```

```
cin >> str
  b cd

cout << str
b
```

```
cin >> str
abcdefghijklmn

cout << str
abcdefghijklmn
```

□ C字符串的处理

C和C++都有字符串常量，但只有C++中才有字符串变量（用string类实现）。C对字符串的处理都是通过字符数组进行的，因此C字符串其实是以'\0'结束的字符数组。

C++中保留了C字符串及其处理函数：

注意：所有字符串处理函数的实参都是字符数组名！

- C字符串的输入输出函数（#include <cstdio>）

（1）输出函数puts

函数原型：int puts(const char* str)

功能：向显示器输出C字符串。

说明：1、字符串输出后自动换行；2、返回一个非负值。

（2）输入函数gets

函数原型：char* gets(char* str)

功能：将键盘输入的C字符串存入字符数组中，并在结尾自动加'\0'。

说明：1、输入的字符串长度应小于字符数组维数；2、键入回车符则结束输入。

例5.6

```
1  #include <cstdio>
2  using namespace std;
3
4  int main( )
5  {
6      char string[80];
7
8      printf("Input a string: ");
9      gets(string);
10     puts(string);
11
12     return 0;
13 }
```

```
Input a string: C String
C String
```

- **C字符串的处理函数（#include <cstring>）**

（3）字符串连接函数strcat

函数原型：char* strcat(char* destination, const char* source)

功能：把字符串source连接到字符串destination后面。返回值：返回destination。

说明：1、字符数组destination必须足够容纳连接后的新字符串；2、连接后，只保留最后的'\0'。

（4）字符串拷贝函数strcpy

函数原型：char* strcpy(char* destination, const char* source)

功能：strcpy将字符串2/strncpy将字符串2的前n个字符，拷贝到字符数组1中去。

返回值：返回destination。

说明： 1、字符数组destination必须足够容纳拷贝过来的新字符串； 2、拷贝时'\0'一同拷贝；3、不能使用赋值语句为一个字符数组赋值。

（5）字符串比较函数strcmp

函数原型：int strcmp(const char* str1, const char* str2)

功能：比较两个字符串，比较规则：从左向右逐个字符比较（ASCII码），直到遇到不同字符或'\0'为止。

返回值：若字符串1 < 字符串2，返回负整数；若字符串1 > 字符串2，返回正整数；若字符串1 == 字符串2，返回零。

说明：字符串比较不能用“==”，必须用strcmp函数。

（6）字符串长度函数strlen

函数原型：size_t strlen(const char* str)

功能：计算字符串长度。

返值：返回字符串实际长度，不包括' \0'在内。

例5.7 strlen函数与sizeof运算符的区别

```c
#include <cstring>
#include <cstdio>

int main()
{
    int m, n, l;

    char str[20] = "A\0BC\0D";
    char str1[] = "A\0BC\0D";

    m = strlen(str);
    n = sizeof(str); //sizeof是运算符，不是函数
    l = sizeof(str1);

    printf("m is %d\nn is %d\nl is %d\n", m, n, l);

    return 0;
}
```

```
m is 1
n is 20
l is 7
```

例5.8 判断字符串的长度

```c
#include <cstring>
#include <cstdio>

int main()
{
    char a[10] = {'A', '\0', 'B', 'C', '\0', 'D'};
    char b[] = "\t\v\\\0will\n";
    char c[] = "\x69\082\n";

    printf("strlen(a) is %d\nstrlen(b) is %d\nstrlen(c) is %d\n",
           strlen(a), strlen(b), strlen(c));

    return 0;
}
```

```
strlen(a) is 1
strlen(b) is 3
strlen(c) is 1
```

5.6 C++的字符串

C++标准库中定义了一个string类，把string类当做一个基本类型，可以实现C++的字符串变量（本质是string类对象）。

string类支持对字符串的运算符操作，同时可以自动地管理内存分配。

使用string类必须#include <string>。

5.6.1 字符串变量的定义和引用

□ 字符串变量的定义及初始化

```
string str1;
string str2 = “C++”;
string str3(“C++”);
string str4 = string(“C++”);
sting str5 = str2;
```

□ 字符串变量的赋值

```
string str1, str2 = "C++";
str1 = "CPlusPlus";        //用字符串常量给str1赋值
str1 = str2;        //用另一个字符串变量str2给str1赋值
str2[0] = 'A';    //可以用下标法引用字符串中的某一个字符，str2="A++"
```

5.6.2 字符串变量的运算

- **赋值运算　=**
- **连接运算　+**
- **关系运算　==　!=　>　<　>=　<=**

5.6.3 字符串数组

string name[5] = {“Zhang”, “Li”, “Hu”, “Wang”, “Tan”};

说明：1、每个元素相当于一个字符串变量；2、每个元素的长度可以不同；3、每个元素只包含字符串的有效字符，而不包括’\0’。