# C语言基础Day4-数组

## 一、数组定义

将若干个相同的数据类型的变量存放在一块连续的内存中，同一个数组中所有的成员都是具有相同的数据类型，同时所有的成员在内存中的地址是连续的。

## 二、声明一个数组

- 数组内每一个元素的类型由数组名前面的类型决定
- 数组中的元素个数由[]里面的数值决定
- 定义数组时，[]里面的值不可以是变量，只能是常量
- 使用数组，[]里面的值可以是常量也可以是变量
- 数值数组不可以整体操作
- 数组的每一个元素都是变量

```
int num[10];

int num[10] = {1,2,3,4,5,6,7,8,9,10};

// 如果数组只初始化部分元素，其他元素被初始化为0
int num[10] = {1,2};
int num[10] = {0};// 将数组所有元素都初始化为0
int num[10] = {[5] = 5};// 将下标为5的元素赋值为5

int num[] = {1,2,3};// []中没有值 这个数组的元素个数由{}里面的元素个数决定
```

## 三、数组的长度

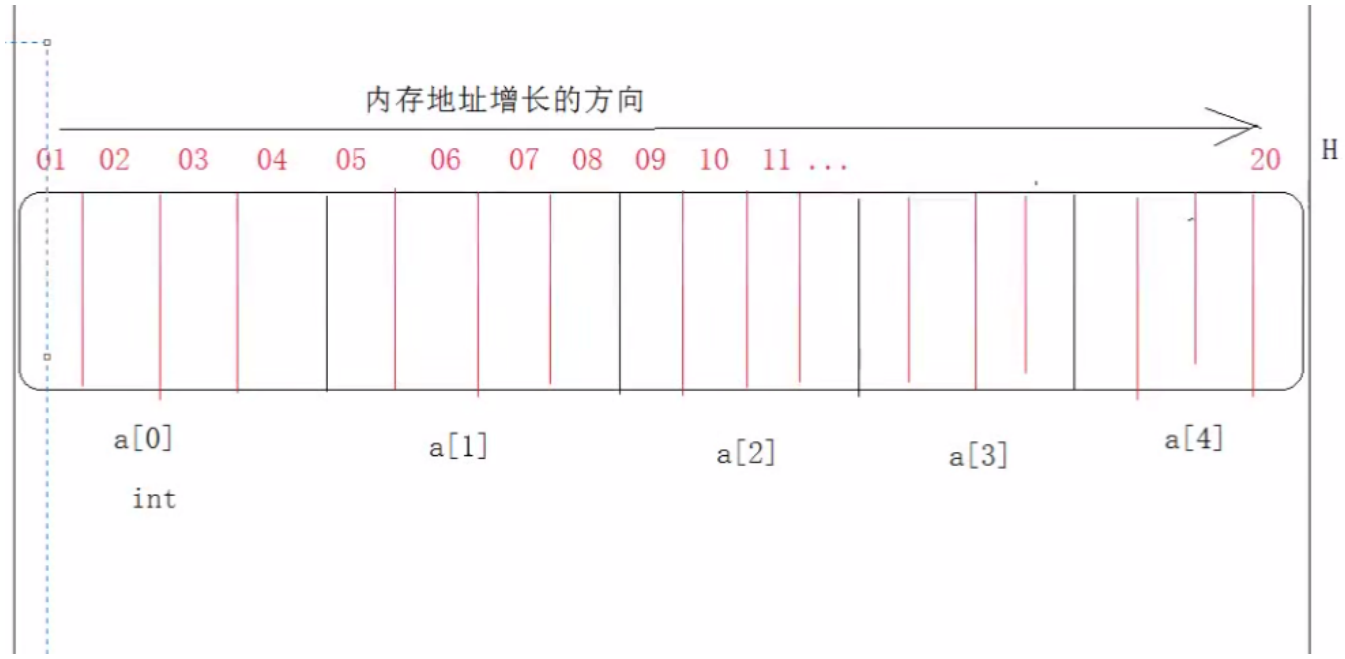
```
int num[10] = {0};
int n = sizeof(num) / sizeof(num[0]);// 数组长度
```

## 四、数组在内存中如何存储

启动一个程序，系统会给这个程序分配一块内存空间；内存的最小单位是一个字节，内存中的每一个字节都会有编号，这个编号就是内存中的地址。

数据在内存中的地址，就是它在内存中的起始地址。

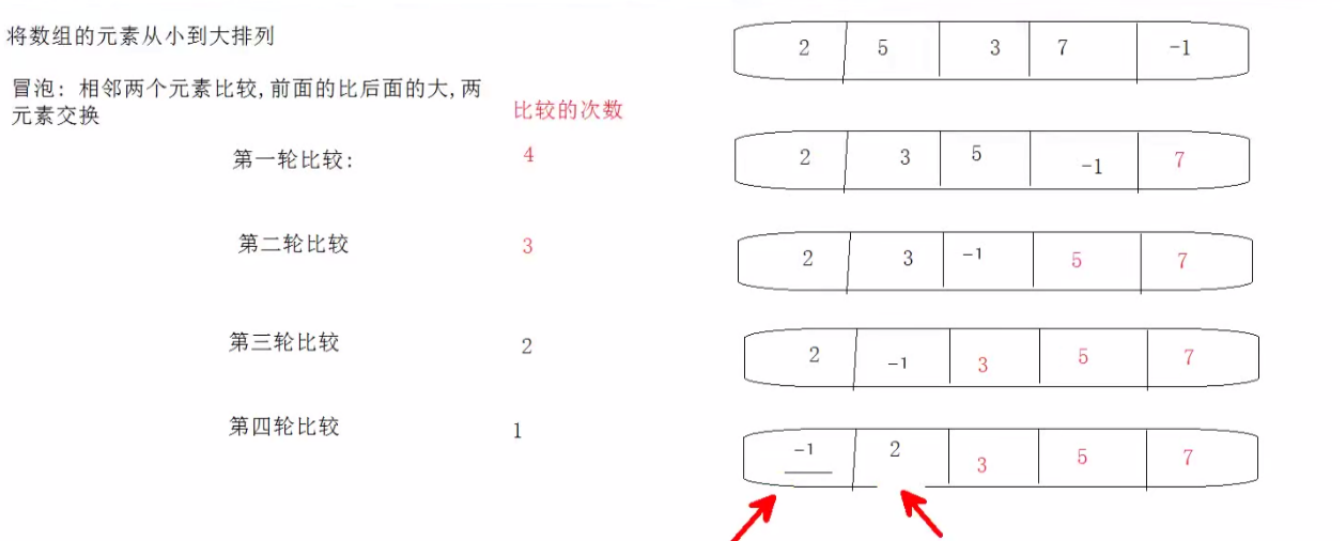
比如，int b,占用内存四个字节，那么这四个字节占用的地址分别是：0x1001、0x1002、0x1003、0x1004，那么b的内存地址就是起始地址:0x1001。




- 说明:
  - a[0] 第0个元素
  - &a[0] 第0个元素的地址 == 01
  - 数组名a 代表数组，也代表第0个元素的地址 a == &a[0] == 01,所以说数组名是一个常量，是一个地址。
  - &a 取得是整个数组的地址 == 01 在数值上 &a[0],a,&a, 相等，但是意义不相同
  - **&a[0] + 1 元素的地址加一 跨过一个元素 == 05**
  - **a+1 元素的地址加一 跨过一个元素 == 05**
  - &a + 1 整个数组的地址加一，跨过整个数组 == 21

```c
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    int a[5];
    printf("%u\n",&a[0]);
    printf("%u\n",a);// 数组名 就是第一个元素的地址
    printf("%u\n",&a);// 取出 数组的地址

    printf("%u\n",&a[0] + 1);//数组第一个元素 加一 跨过一个元素  + 4
    printf("%u\n",a + 1);// 元素的地址加一 跨过一个元素
    printf("%u\n",&a + 1);// 整个数组的地址加一  跨过整个数组

    return 0;
}
```

# 五、冒泡排序

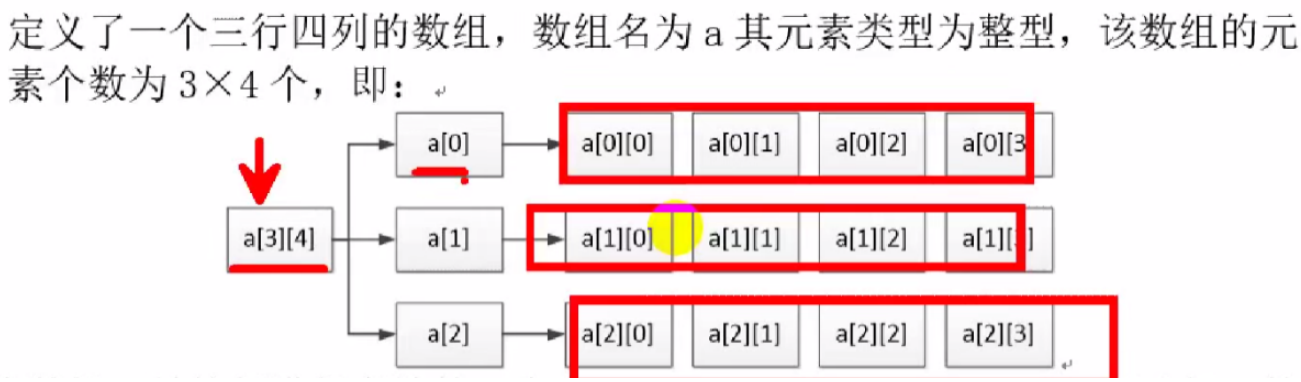



冒泡排序：相邻的两个元素进行比较，前面的比后面的大，两个元素进行交换，如果数组有n个元素，那么总共需要交换n - 1轮，每i轮交换n - i次。

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    int a[5] = {2,5,6,3,-1};
    /*printf("%u\n",&a[0]);
    printf("%u\n",a);// 数组名 就是第一个元素的地址
    printf("%u\n",&a);// 取出 数组的地址

    printf("%u\n",&a[0] + 1);//数组第一个元素 加一 跨过一个元素  + 4
    printf("%u\n",a + 1);// 元素的地址加一 跨过一个元素
    printf("%u\n",&a + 1);// 整个数组的地址加一  跨过整个数组*/

    int n = sizeof(a) / sizeof(a[0]);// 计算数组有多少个元素  将数组所有字节长度除以单个元素的长度

    for (int i = 0; i < n - 1; i++)
    {
        // 外层循环 主要是控制循环次数
        // 内层循环主要是控制每一次循环的比较次数
        for (int j = 0; j < n - i - 1; j++)
        {
            if (a[j] > a[j+1])
            {
                // 交换元素
                int t = a[j];
                a[j] = a[j+1];
                a[j+1] = t;
            }
        }
    }
```

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

    return 0;
}
```

# 六、二维数组




- 二维数组在概念上是二维的：其下标在两个方向上变化，对其访问一般需要两个下标
- 在内存中并不存在二维数组，二维数组实际的硬件存储器是连续编址的，也就是说内存中只有一维数组，即放完一行之后顺次放入第二行，和一维数组存放方式是一样的。
- int a[2][4] 定义了一个二维数组，二维数组有两个一维，每个一维数组有四个元素，那么二维数组的a[0]代表第0行

## 6.1 数组的初始化

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    int a[3][4] = { {1,2,3,4},{5,6,7,8},{9,10,11,12} };// 二维数组的初始化

    // 如果给二维数组的部分元素初始化，其他元素为0
    int b[3][4] = { 1,2,3 };

    int c[3][4] = {1,2,3,4,5,6,7,8,9,10,11,12};// 二维数组本质上还是一维数组  可以自动计算下标

    int d[][3] = { 1,2,3,4,5 };// 二维数组定义时 不能省略列的下标 可以省略行的下标 可以自动计算行的下标

    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 3; j++)
        {
```

```
            printf("%d ",d[i][j]);
        }
    }

    return 0;
}
```

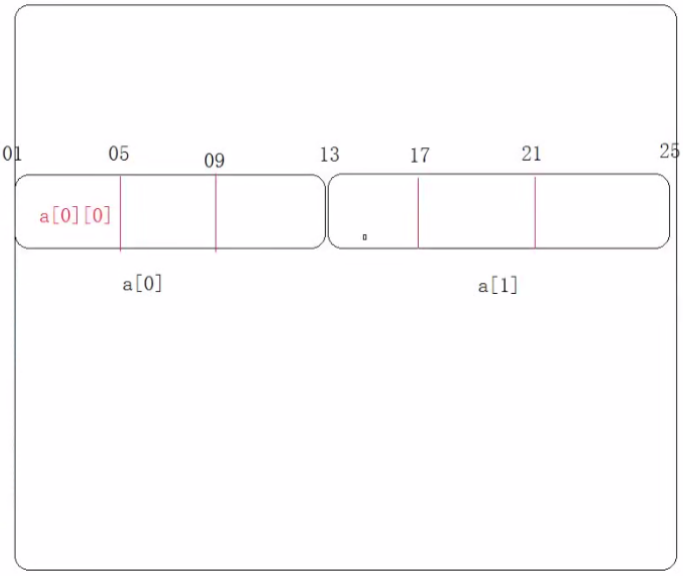
## 6.2 计算数组的行数和列数

- 计算数组元素的总个数：sizeof(a) / sizeof(a[0][0]);
- 计算数组的行数：sizeof(a) / sizeof(a[0]); a[0]代表一行元素的大小
- 计算数组的列数：sizeof(a[0]) / sizeof(a[0][0]);一行元素的字节数除以一个元素的字节数

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    int a[3][4] = { {1,2,3,4},{5,6,7,8},{9,10,11,12} };// 二维数组的初始化

    int n = sizeof(a) / sizeof(a[0][0]);// 计算元素的总个数
    int row = sizeof(a) / sizeof(a[0]);// 计算数组的行数  二维数组的大小除以一行的大小
    int column = sizeof(a[0]) / sizeof(a[0][0]);// 计算数组的列数  一行的大小除以一个元素的大小

    printf("%d %d %d\n",n,row,column);

    return 0;
}
```

## 6.3 数组名

```
int  a[2][3];

a[0][0]    第0行第0个元素
&a[0][0]   第0行第0个元素的地址 = 01
a[0]     代表第0行一维数组的数组名,  a[0] = &a[0][0]  01
&a[0]    第0行的地址 = 01
a     二维数组数组名,代表二维数组,也代表首行地址 &a[0]
&a    二维数组的地址                          int a[2][3]
```




```
int a[2][3];

* a[0][0]   第0行第0个元素
* &a[0][0]  第0行第0个元素的地址
* a[0]  代表第0行一维数组的数组名，也代表第0行一维数组的地址a[0] = &a[0][0]
* &a[0] 第0行的地址 = 01
* a 二维数组的数组名 代表二维数组，也代表首行地址&a[0]
* &a 二维数组的地址

* &a[0][0] + 1  元素地址加1，跨过一个元素
* a[0] + 1      元素地址加一，跨过一个元素
* &a[0] + 1     行地址加一，跨过一行
* a + 1         行地址加一，跨过一行
* &a + 1        二维数组地址加一，跨过整个数组
```

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    int a[3][4] = { {1,2,3,4},{5,6,7,8},{9,10,11,12} };// 二维数组的初始化

    int n = sizeof(a) / sizeof(a[0][0]);// 计算元素的总个数
    int row = sizeof(a) / sizeof(a[0]);// 计算数组的行数  二维数组的大小除以一行的大
小
    int column = sizeof(a[0]) / sizeof(a[0][0]);// 计算数组的列数  一行的大小除以一
个元素的大小

    printf("%d %d %d\n",n,row,column);

    printf("%u\n",&a[0][0]);
    printf("%u\n",a[0]);
```

```
    printf("%u\n",&a[0]);
    printf("%u\n",a);
    printf("%u\n",&a);

    printf("%u\n",&a[0][0] + 1);// 只加上一个元素地址
    printf("%u\n",a[0] + 1);// 只加上一个元素的地址
    printf("%u\n",&a[0] + 1);// 加上一行元素的地址
    printf("%u\n",a + 1);// 加上一行元素的地址
    printf("%u\n",&a + 1);// 加上整个数组的元素地址


    return 0;
}
```

# 七、多维数组

int num[2][3][4]; 定义了一个三维数组，有两个二维数组，每一个二维数组都有三个一维数组，每一个一维数组有四个元素。

# 八、字符数组

字符串就是字符数组中有\0字符的数组，因为有\0字符的字符数组，操作起来方便，**数字0（字符'\0'的ACSII码）结尾的char数组就是一个字符串，但是如果chaR数组没有以数字0结尾，那么就不是一个字符串，只是普通的字符数组，所以字符串是一种特殊的char数组。**

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    char a[5] = {'a','b','c','d','\0'};// 字符数组中含有\0字符的 就是字符串
    char b[5] = "abcd";// 定义了一个字符数组 存的是abcd\0
    char c[] = "world";// 数组的长度是6  字符串
    char d[100] = "abc";// 定义了一个字符数组 有100个元素 其他元素都是\0
    char e[100] = {0};// 将一个字符数组清0 0是字符'\0' 的acsii码

    for (int i = 0; i < sizeof(a) / sizeof(a[0]); i++)
    {
        printf("%d ",a[i]);// 单个打印字符  十进制形式打印
    }
    printf("\n");
    // 字符数组也可以当作字符串进行打印 只要直到数组的首地址
    printf("%s\n",a);
    printf("%s\n",b);
    printf("%s\n",c);
    printf("%s\n",d);
    printf("%s\n",e);// 全部都是空格

    return 0;
```

```
}
```

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    char a[128] = "abcd\0def\0";

    printf("%d\n",sizeof(a));// 128
    printf("%s\n",a);// 打印abcd


    return 0;
}
```

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    char a[] = {'a','b','c'};// 普通的字符数组
    printf("%d\n",sizeof(a));// 输出3
    printf("%s\n",a);// 没有字符0 结尾  输出abc烫烫 乱码
    return 0;
}
```

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    char a[] = {'a','b',0,'c'};// 中间有0 表示'\0'
    char b[] = {'a','b','0','c',0};// 中间的0字符会被打印 因为acsii是48
    char c[] = {'a','b','c','\0','v'};// 打印abc

    printf("%s\n",a);// 打印ab
    printf("%s\n",b);// 打印ab0c
    printf("%s\n",c);// 打印abc

    return 0;
```

```
}
```

## 九、scanf输入字符串

**遇到回车结束，遇到空格也结束，同时，如果存放读取字符的空间不足，继续想后面存放字符，没有'\0'字符，那么打印的时候会直接将所有的字符都打印，即使超过字符数组的长度，这就造成了内存污染**

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    char num[128] = "";// 字符数组清0
    scanf("%s",num);// 从键盘中获取一个字符串  遇到回车结束 不需要取地址符
    printf("%s\n",num);// %s 要的是打印字符数组的首元素地址

    return 0;
}
```

## 十、gets函数

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    // gets函数遇到\n结束  但是遇到空格不会结束 读取空格  但是也会造成内存污染
    char num[128] = "";// 字符数组清0
    gets(num);// 参数是存放读取字符串的地址
    printf("num = %s\n",num);

    return 0;
}
```

## 十一、fgets()函数

库函数：从键盘读取一个字符串；char num[128]; fgets(num,sizeof(num),stdin);// fgets从stdin（标准输入-键盘）读取字符串到num数组中，最大可以读取sizeof(num) - 1个字符，因为后面需要加上'\0';

- 如何将最后一个字符重置成'\0'

```

```

```c
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)

int main()
{
    char buf[128] = "helloA";

    // 将字符串最后一个字符置为'\0'
    // 首先需要找到最后一个字符的下标
    // 求的是字符数组的有效字符个数

    int i = 0;
    while (buf[i] != '\0')
    {
        i++;
    }
    // 最后i 落了下标6的位置
    printf("%d\n",i);

    buf[i - 1] = '\0';// 将最后一个字符置为'\0'
    printf("%s\n",buf);// 打印hello

    return 0;
}
```

**使用strlen()函数计算字符数组的有效字符个数，然后将最后一个字符置为'\0'**

```c
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)
#include<stdlib.h>
#include<string.h>

int main()
{
    char buf[128] = "helloA";

    // 将字符串最后一个字符置为'\0'
    // 首先需要找到最后一个字符的下标
    // 求的是字符数组的有效字符个数

    int i = strlen(buf);// strlen() 计算字符数组有效字符的个数

    buf[i - 1] = '\0';// 将最后一个字符置为'\0'


    printf("%s\n",buf);// 打印hello

    return 0;
```

```
}
```

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)
#include<stdlib.h>
#include<string.h>

int main()
{
    //fgets会把回车键读取
    char buf[1024] = "";
    fgets(buf,sizeof(buf),stdin);// 读取换行符
    buf[strlen(buf) - 1] = '\0';// 将换行符直接置为'\0'
    printf("%s\n",buf);// 打印hello

    return 0;
}
```

# 十二、字符数组的输出

- puts、fputs函数

```
#include<stdio.h>
#define _CRT_SECURE_NO_WARNINGS
#pragma warning(disable:4996)
#include<stdlib.h>
#include<string.h>

int main()
{
    char buf[1024] = "helloworld";
    puts(buf);// 自动输出一个换行
    fputs(buf,stdout);// 第一个参数，数组首元素地址 stdout标准输出（屏幕）


    return 0;
}
```