第六章 数组

目录

6.1 数组的概念

1、在程序设计中，为了方便处理数据把具有相同类型的若干变量按有序形式组织起来,这些**按序排列**的**同类数据元素**的集合称为数组

2、在C语言中，数组属于构造数据类型。一个数组可以分解为多个数组元素，这些数组元素可以是基本数据类型或是构造类型。因此按数组元素的类型不同，数组又可分为**数值数组、字符数组、指针数组、结构数组**等各种类别。本章介绍数值数组和字符数组，其余的在以后各章陆续介绍

3、总结

- **数组中的元素是连续的（元素的内存地址连续）**
- **同一个数组所有的成员都是相同的数据类型**

6.2 一维数组

6.2.1 一维数组的定义

1、一维数组的定义语法规则

> 数据类型 数组名[常量表达式];

> 说明：
>
> 1）数组名的命名规范必须满足“标识符的命名规范”
>
> 2）方括号中的常量表达式就是数组的长度，也就是数组中存储元素的个数

例如：

```
int a[10];     说明整型数组a，有10个元素。
float b[10],   c[20];    说明实型数组b，有10个元素，实型数组c，有20个元素
char ch[20];   说明字符数组ch，有20个元素。
```

6.2.2 一维数组的使用

1、使用**下标**访问数组中的元素

- 下标的值必须为**整形常量/变量**
- 下标的值从**0** 开始 到**数组长度-1** 结束
- 注意：**如果使用的下标值大于或者等于数组长度，程序仍然可以编译通过，但是运行结果是未知的**！为何如此呢？请听下回分解！

```
#include <stdio.h>

int main()
{
        int a[10];//定义了一个数组，名字叫a，有10个成员，每个成员都是int类型
        int i = 0;
        for (i = 0; i < 10; i++)
        {
                a[i] = i; //给数组赋值
        }

        //遍历数组，并输出每个成员的值
        for (i = 0; i < 10; i++)
        {
                printf("%d ", a[i]);
        }
        printf("\n");

        return 0;
}
```

2、数组的初始化

- 给数组赋值的方法除了用赋值语句对数组元素逐个赋值外，还可采用初始化赋值和动态赋值的方法。

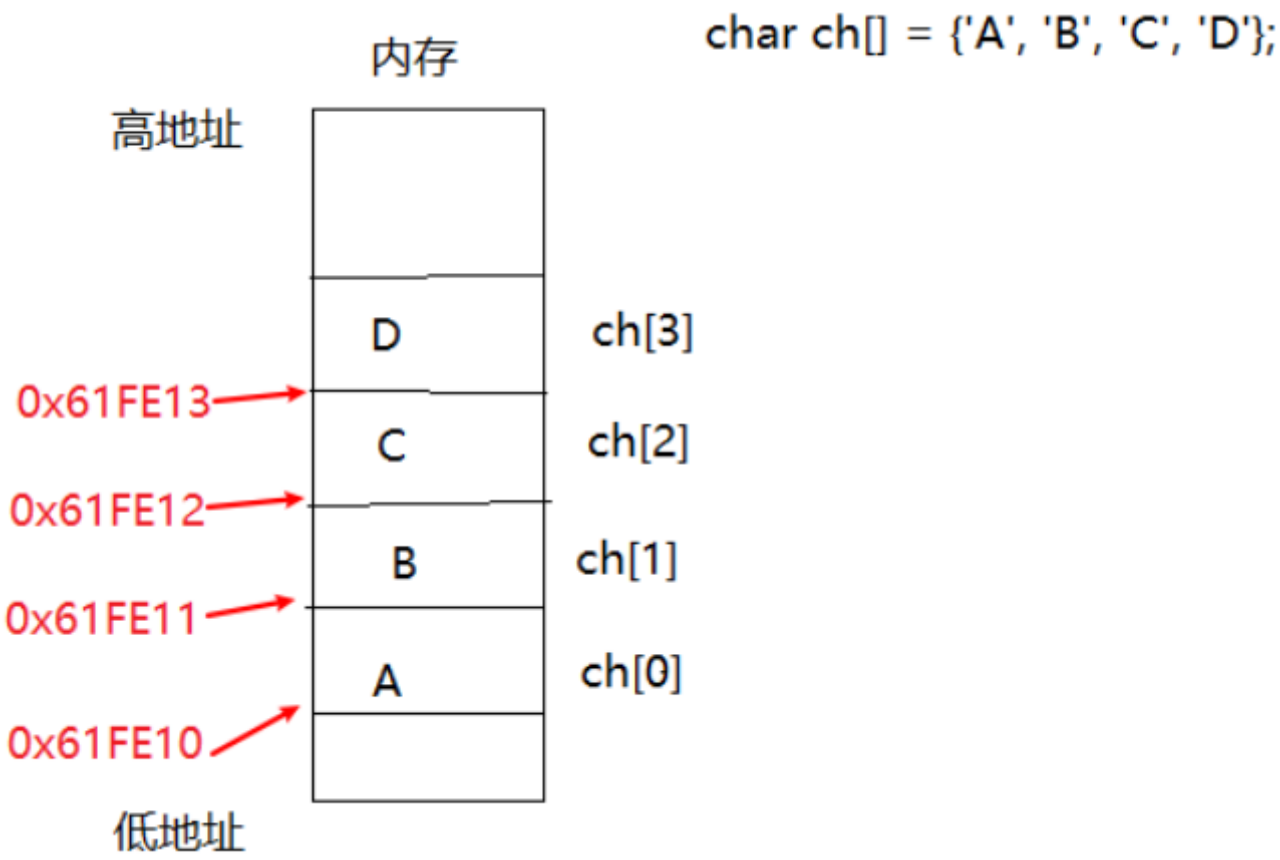
```
int a[10] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };//定义一个数组，同时初始化所有成员变量
int a[10] = { 1, 2, 3 };//初始化前三个成员，后面所有元素都设置为0
int a[10] = { 0 };//所有的成员都设置为0
```

```
//[]中不定义元素个数，定义时必须初始化
int a[] = { 1, 2, 3, 4, 5 };//定义了一个数组，有5个成员
```

- **注意：局部数组如果不初始化，内容为随机值。**

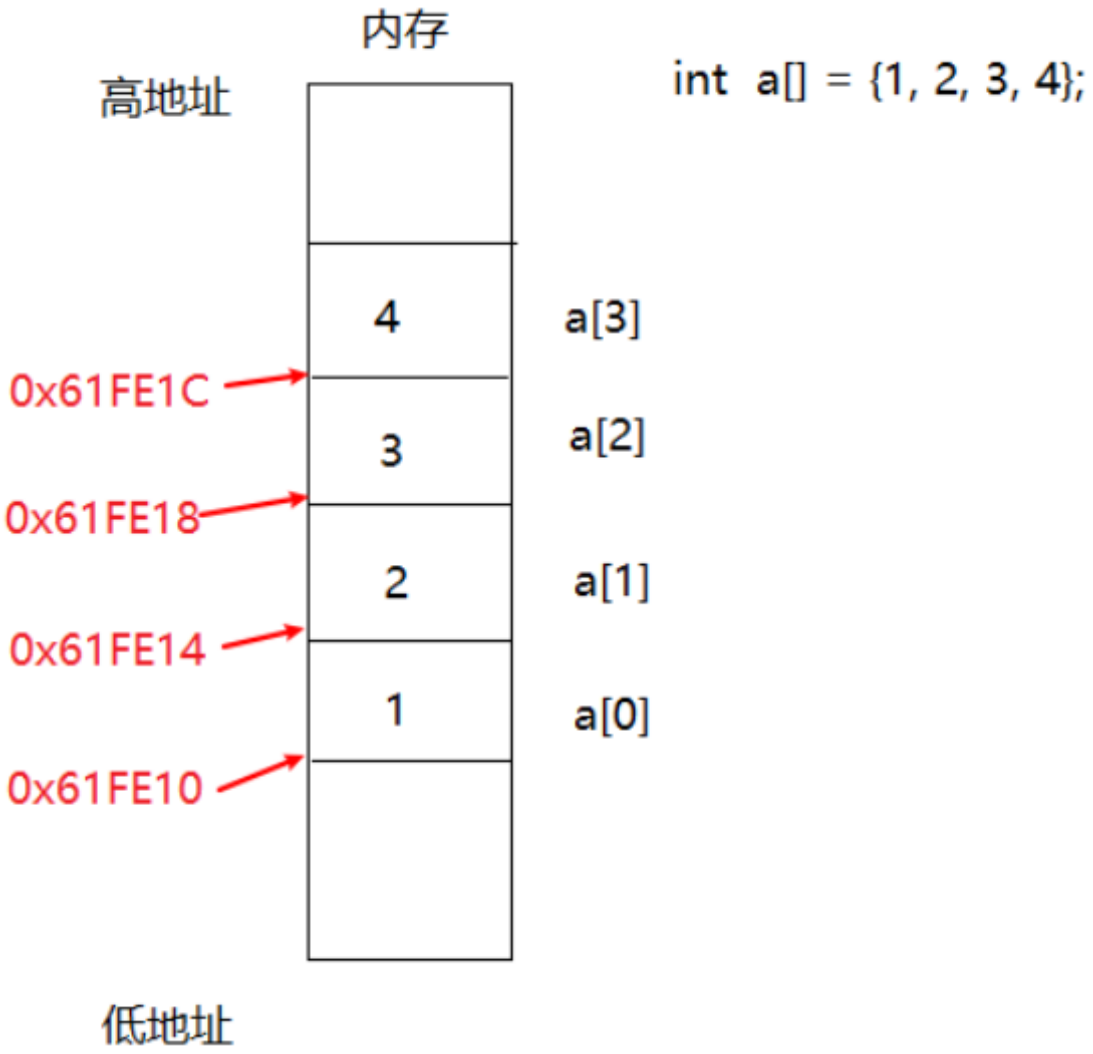
3、一维数组元素在内存中的存储

- 字符数组




```
char ch[] = {'A', 'B', 'C','D'};
int i;
for (i = 0; i < 4; i++)
    printf("%p ", &ch[i]);
printf("\n");
```

- 整型数组




```
int a[] = {1, 2, 3, 4};
int i;
for (i = 0; i < 4; i++)
    printf("%p ", &a[i]);
printf("\n");
```

- 通过**sizeof(数组名)**可以求数组在内存中占用的字节数
- 思考：假如有一维数组a，我们不知道数组中元素的数据类型，求数组a的长度

```
len = sizeof(a)/sizeof(a[0])
```

4、练习

- 求数组中所有元素的和
- 求数组中的最大值和最小值以及平均值
- 定义一个整型数组，长度随意，内容随意，判断该数组是否为升序数组，如果是打印yes，不是打印no

6.2.3 一维数组应用举例

1、一维数组的逆置

```c
#include <stdio.h>

int main()
{
        int a[] = {  1, -2, 3,-4, 5, -6, 7, -8, -9, 10 };//定义一个数组，同时初始化所有成员变量

        int i = 0;
        int j = sizeof(a) / sizeof(a[0]) -1;
        int tmp;

        while (i < j)
        {
                tmp = a[i];
                a[i] = a[j];
                a[j] = tmp;
                i++;
                j--;
        }

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

        return 0;
}
```

2、删除一维数组中指定的元素

```c
#include <stdio.h>
int main()
{
    int i,j,x,a[]={12,5,4,23,8,18,41,34,15,9};
    printf("输入要删除的数: ");
    scanf("%d",&x);
    for(i=0;i<10;++i)
    {
        if(x==a[i])
        {
            for(j=i;j<9;++j)  //删除指定的数
                a[j]=a[j+1];
            a[9]=0;
            break;
        }
    }
    if(i==10)
        printf("没有找到指定的数");
    else
    {
```

```
        for(i=0;i<9;++i)  //输出删除后的数组
            printf("%d    ",a[i]);
    }

    return 0;
}
```

6.2.4 企业笔试题

1、你知道你活了多少天了吗?

2、将一个无序一维数组的所有的奇数放前面，所有的偶数放后面

3、求一个无序一维数组中第二大的值（不能对数组进行整体排序）

4、假如有有序数组int a[10] = {1, 5, 9, 14, 20}; 请编写代码将数组 int b[] = {8,2,11,30,16};合并到数组a中，并且最后数组a依然有序

6.3 二维数组

6.3.1 二维数组的定义

1、在实际应用中有许多数据是二维的，例如棋盘是有多行多列，如果使用C语言描述一个棋盘我们需要使用二维数组。

2、定义二维数组的语法规则

> 数据类型 数组名[常量表达式1][常量表达式2];
>
> 说明:
>
> 1）我们可以将二维数组当作一个有行有列的二维 矩阵
>
> 2）常量表达式1代表矩阵的行数
>
> 3）常量表达式2代表矩阵的列数
>
> 4）二维数组可以理解为由 “常量表达式1”个一维数组所组成的

```
int a[3][4];
数组a是一个有3行4列的二维数组，数组中元素的数据类型为int
```

6.3.2 二维数组的使用

1、二维数组也同样通过下标对数组中的元素进行访问，与一维数组不同的是，二维数组元素的访问需要使用两个下标

- 行下标的值从0 开始到“常量表达式1” -1

- 列下标的值从0开始到“常量表达式2”-1
- 通过双重循环访问二维数组中的元素

```
#include <stdio.h>
int main()
{
    int a[2][3];
    int m, n;
    //一行一行访问
    for (m = 0; m < 2; m++)
        {
                for (n = 0; n < 3; n++)
                {
                        printf("%d ", a[m][n]);
                }
                printf("\n");
        }
    //一列一列访问
    for (m = 0; m < 2; m++)
        {
                for (n = 0; n < 3; n++)
                {
                        printf("%d ", a[n][m]);
                }
                printf("\n");
        }
    return 0;
}
```

2、二位数组的初始化

- 分段赋值

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

- 连续赋值

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

- 部分赋值

```
//可以只给部分元素赋初值，未初始化则为0
int a[3][4] = { 1, 2, 3, 4  };
```

- 0值初始化

```
//所有的成员都设置为0
int a[3][4] = {0};
```

- 不指定行数

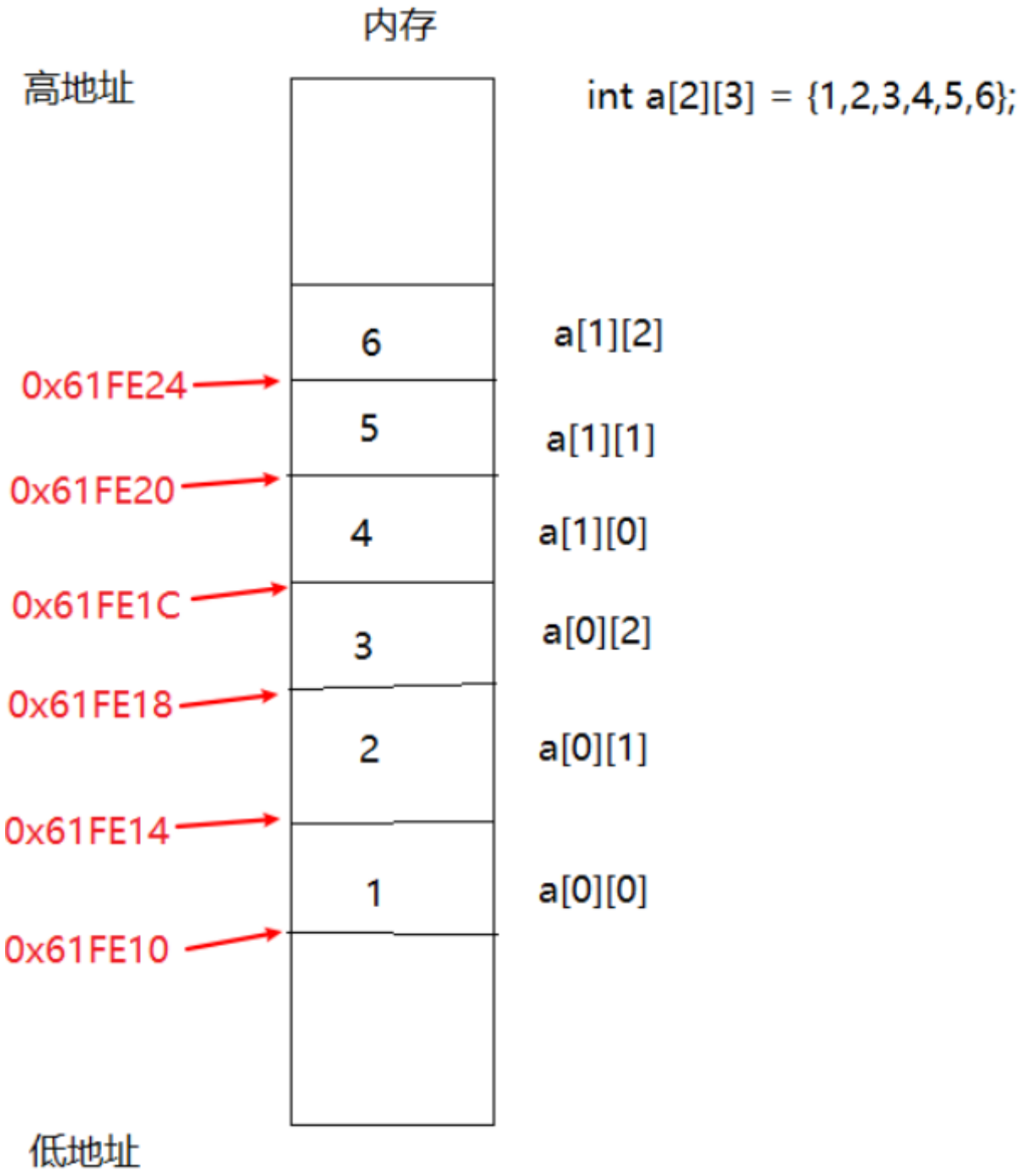
```
//[]中不定义元素个数，定义时必须初始化
int a[][4] = { 1, 2, 3, 4, 5, 6, 7, 8};
```

- 不能不指定列数

```
int a[4][] = { 1, 2, 3, 4, 5, 6, 7, 8};//错误的初始化
```

3、二维数组中元素在内存中的存储

- 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。



4、练习

- 求二维数组每一行的最大值并且打印出最大值的列号, 求每一列的最大值并且打印所在的行号

```
int a[2][3] = {{1,2,3}, {4,5,6}};
int max;
int index;
for (m = 0; m < 2; m++){
    max = a[m][0];
    index = 0;
    for (n = 0; n < 3; n++){
        if (a[m][n] > max){
            max = a[m][n];
            index = n;
        }
```

```
    }
    printf("max: %d, index: %d\n", max, index);
}

for (m = 0; m < 3; m++){
    max = a[0][m];
    for (n = 0; n < 2; n++){
        if (a[n][m] > max){
            max = a[n][m];
            index = n;
        }
    }
    printf("max: %d, index: %d\n", max, index);
}
```

- 求二维数组a[4][4]的对角线之和

```
#include <stdio.h>

int main()
{
    int a[4][4] = {{1, 2, 3, 4},
                   {5, 6, 7, 8},
                   {9 ,10,11,12},
                   {13,14,15,16}};
    int i,j;
    int sum =0;
    for (i = 0; i < 4; i++)
    {
        for (j = 0; j < 4; j++)
        {
            if (j == i || j == 3-i)
            {
                sum += a[i][j];
            }
        }
    }
    printf("%d\n", sum);
    return 0;
}
```

6.3.3 二维数组应用举例

1、一个学习小组有5个人， 每个人有三门功课。求全组分科的平均成绩

	张	王	李	赵	周
Math	80	61	59	85	76
C	75	65	63	87	77
Foxpro	92	71	70	90	85

可设一个二维数组a[5][3]存放五个人三门课的成绩。再设一个一维数组v[3]存放所求得各分科平均成绩。

```
#include <stdio.h>
int main()
{
    int a[5][3] = {{80,75,92},{61,65,71},{59,63,70},{85,87,90},{76,77,85}};
    int v[3];
    int i, j, sum = 0;
    for (i = 0; i < 3; i++)
    {
        for (j = 0; j < 5; j++)
        {
            sum += a[j][i];
        }
        v[i] = sum / 3;
        sum = 0;
    }
    return 0;
}
```

6.3.4 企业笔试题

1、求二维数组中的鞍点（有一个数满足：行中最大，列中最小）

```
for (m = 0; m < 4; m++)
{
    max = a[m][0];
    index_col = 0;
    //求出当前行的最大值并且记住最大值的列号
    for(n = 0; n < 4; n++)
    {
        if (a[m][n]>max)
        {
            max = a[m][n];
            index_col = n;
        }
    }
    min = a[0][index_col];index_row = 0;
    // 求最大值所在的列上的最小值，并且记住最小值所在的行号
    for(k = 0; k < 4; k++)
    {
```

```
            if (a[k][index_col] < min)
            {
                min = a[k][index_col];
                row = k;
            }
        }
        //如果列上的最小值所在的行号和当前行是相等的
        if (index_row == m)
        {
            printf("下标为 [%d, %d] 的 %d 是鞍点！\n", m, index_col, min);
            break;
        }
    }
```