

Java语言与系统设计

第4讲 数组

- 一维数组
 - 二维数组
 - 数组工具类
-

1.一维数组

数组对象和元素类型：

- ❑ 数组是由同一类型的数据元素构成的一种组合数据结构，被称为数组对象。数组中的每个值称为数组元素。
 - ❑ 数组本身是引用数据类型，而数组中的元素可以是任何数据类型，可以是基本数据类型也可以是类（对象）类型。例如一个数组中的元素可以是标准整型int，字符类型char，也可以是学生等对象类型。
 - ❑ 创建数组对象会在内存中开辟一整块连续的空间，而数组名中引用的是这块连续空间的首地址。
-

1. 一维数组

一维数组定义格式:

<元素类型> []<数组名>; //[]放到数组名后

- 说明：数组名是用户命名的一个标识符，数组名前或后使用一对中括号。
 - 例如：int []a; double b[]; Object []obj;
 - ◆ Java语言中声明数组时不能指定其长度(数组中元素的数)，例如：int a[5]; //非法
-

一维数组创建格式： new <元素类型>[<元素个数>];

- 说明：创建数组使用new运算符，new后面跟着定义数组时使用的元素类型，元素类型后面是一对中括号，其中给出数组将包含的元素个数。
- 执行new运算时，将在内存中分配保存数组中所有元素的存储空间，并把该存储空间的首地址作为运算结果返回，然后需要把该地址赋给被定义的数组对象，以便通过数组对象访问数组中的元素。
- 例如： a=new int[10]; s=new Student[num];

❑ 一维数组的定义和创建可以合并进行，例如：

(1) `int []a=new int[10];`

// 定义和创建包含10个整数元素的数组a

(2) `Object []obj=new Object[num];`

// 定义和创建具有num个元素的数组obj

❑ 数组一经分配空间，其中的每个元素也被按照成员变量同样的方式被隐式初始化

数组元素类型	元素默认初始值
byte	0
short	0
int	0
long	0L
float	0.0F
double	0.0
char	0 或写为:'\u0000'(表现为空)
boolean	false
引用类型	null

□ 创建数组时同时进行元素赋值

□ 例如： `int []a={1,3,5,7};` // 定义并创建数组a，它包含4个元素，每个元素的初值依次为1，3，5和7。

□ 计算机在执行带有初始化表的数组定义语句时，将隐含执行一次new运算，创建一个数组存储对象，该数组的长度（即包含元素的个数）等于初始化表中数据项的个数，接着依次把初始化表中每个数据项的值赋给数组中的每个元素，最后把数组存储对象的首地址赋给被定义的数组引用对象。例如在执行上面第(1)条语句后，数组a包含有4个元素，每个元素的值依次为1、3、5和7。

数组元素的表示：<数组名>[<下标表达式>]

- 数组中的元素从0开始编号，此编号又称为该元素的下标，若数组的长度为 n ，则元素的下标从前向后依次为0、1、2、 \dots 、 $n-1$ ，每个元素用来存储具有元素类型的一个值。
 - 数组中的每个元素又称为下标变量，它用数组名后跟一对中括号表示，中括号内为下标表达式，下标表达式的值就是该元素的下标。例如 $a[k-1]$ 就是 a 数组中的一个元素，又称为下标变量，其下标表达式为 $k-1$ ，对应数组 a 中的第 k 个元素，因为第 k 个元素的下标为 $k-1$ 。
-

数组长度：

- 数组长度就是数组中包含的元素个数，当定义和创建一个数组后，数组长度值被自动保存到数组对象的成员变量length中。
- 数组长度是一个常量成员变量，被创建数组时自动初始化后，以后不允许改变它的值，只允许通过点运算符读取它的值。
- 例如，假定a是一个一维数组，它包含有10个元素，则a.length的值为10。

一维数组的内存存储实现

```
int[] arr = new int[] {1,2,3};  
String[] arr1 = new String[4];  
arr1[1] = "花花";  
arr1[2] = "萧敬腾";  
arr1 = new String[3];  
sysout(arr1[1]); // null
```

栈 (stack)

0x56ef
arr1: 0x34cd
arr: 0x12ab

堆 (heap)

0x56ef



0x34cd



0x12ab



(1) 数据常见算法-数组元素计算

```
int []a=new int[5];  
a[0]=5;  
for(int i=1; i<a.length; i++)  
a[i]=a[i-1]*2+1;  
for(int i=0; i<a.length; i++)  
System.out.print(a[i]+" ");  
  
//5 11 23 47 95
```

(2) 数据常见算法-数组元素求和

```
int []b={3,5,8,10,7,5,9};
```

```
int sum=0;
```

```
for(int i=0; i<b.length; i++)
```

```
    sum+=b[i];
```

```
System.out.println("sum="+sum);
```

```
//输出： sum=47
```

(3) 数据常见算法-数组元素求最大值

```
int []b={3,5,8,10,7,5,9};
```

```
int max=b[0];
```

```
for(int i=1; i<b.length; i++)
```

```
if(b[i]>max) max=b[i];
```

```
System.out.println("max="+max);
```

```
//输出: max=10
```

2.二维数组

- 二维数组是由同一类型的数据元素所构成的组合数据结构，用来存储具有行、列结构的二维矩阵数据。
 - 同一维数组一样，二维数组也被看作为对象。二维数组中的每个元素可以是基本数据类型，也可以是对象类型。
-

2.二维数组

- 对于二维数组的理解，我们可以看成是一维数组array1又作为另一个一维数组array2的元素而存在。
 - 从数组底层的运行机制来看，其实没有多维数组。
-

2.二维数组

格式1（动态初始化）： `int[][] arr = new int[3][2];`

定义了名称为arr的二维数组

二维数组中有3个一维数组

每一个一维数组中有2个元素

一维数组的名称分别为arr[0], arr[1], arr[2]

给第一个一维数组1脚标位赋值为78写法是：arr[0][1] = 78;

格式2（动态初始化）： `int[][] arr = new int[3][];`

二维数组中有3个一维数组。

每个一维数组都是默认初始化值null (注意：区别于格式1)

可以对这个三个一维数组分别进行初始化

arr[0] = new int[3]; arr[1] = new int[1]; arr[2] = new int[2];

2.二维数组

格式3（静态初始化）： `int[][] arr = new int[][]{{3,8,2},{2,7},{9,0,1,6}};`

定义一个名称为arr的二维数组，二维数组中有三个一维数组

每一个一维数组中具体元素也都已初始化

第一个一维数组 `arr[0] = {3,8,2};`

第二个一维数组 `arr[1] = {2,7};`

第三个一维数组 `arr[2] = {9,0,1,6};`

第三个一维数组的长度表示方式：`arr[2].length;`

二维数组的内存存储实现

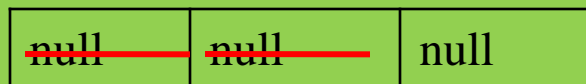
```
int[][] arr4= new int[3][];  
sysout(arr4[0]); //null  
sysout(arr4[0][0]); //报错  
arr4[0] = new int[3];  
arr4[0][1] = 5;  
arr4[1] = new int[] {1,2};
```

栈 (stack)

arr4:0x23ab

堆 (heap)

0x23ab



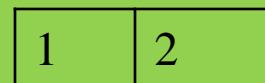
0x56ef

0x34cd

0x56ef



0x34cd



3.数据工具类Array

- ❑ `java.util.Arrays`类即为操作数组的工具类，包含了用来操作数组（比如排序和搜索）的各种方法。

1	<code>boolean equals(int[] a,int[] b)</code>	判断两个数组是否相等。
2	<code>String toString(int[] a)</code>	输出数组信息。
3	<code>void fill(int[] a,int val)</code>	将指定值填充到数组之中。
4	<code>void sort(int[] a)</code>	对数组进行排序。
5	<code>int binarySearch(int[] a,int key)</code>	对排序后的数组进行二分法检索指定的值。

3.数据工具类Array

例如，java.util.Arrays类的sort()方法提供了排序功能

```
import java.util.Arrays;

public class SortTest{
    public static void main(String[] args){
        int[] numbers={5,900,1,5,77,30,64,700};
        Arrays.sort(numbers);
        for(int i=0;i<numbers.length;i++){
            System.out.println(numbers[i]);
        }
    }
}
```

3.数据工具类Array

□ 数组操作常见异常。

数组脚标越界异常(ArrayIndexOutOfBoundsException)

```
int[] arr = new int[2];  
System.out.println(arr[2]);  
System.out.println(arr[-1]);
```

访问到了数组中的不存在的脚标时发生。

空指针异常(NullPointerException)

```
int[] arr = null;  
System.out.println(arr[0]);
```

arr引用没有指向实体，却在操作实体中的元素时。

◆ 编译时不报错

例题：

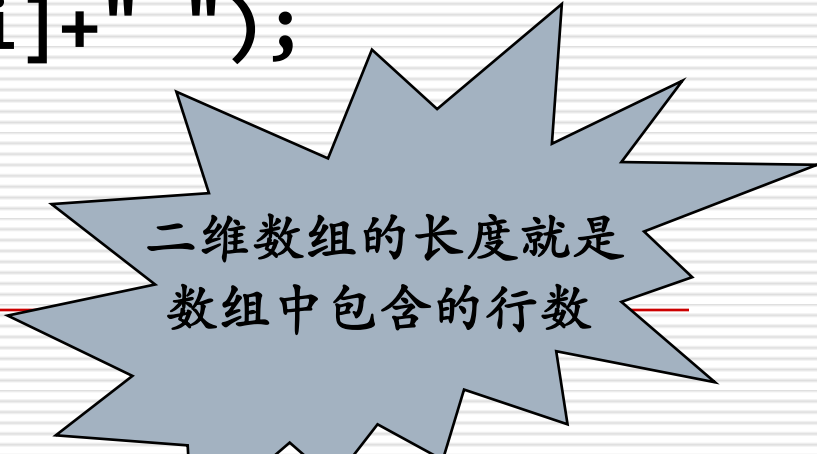
在给 `int [] x, int [][] y` 变量赋值以后，
以下选项允许通过编译的是：

- a) `x[0] = y;`
- b) `y[0] = x;`
- c) `y[0][0] = x;`
- d) `x[0][0] = y;`
- e) `y[0][0] = x[0];`
- f) `x = y;`

//b和e能通过编译

例题：

```
int [][]b={{1,3,5,8},{2,4,6,9},{3,4,5,6}};  
int []a=new int[b.length]  
for(int i=0; i<b.length; i++)  
    for(int j=0; j<b[i].length; j++)  
        a[i]+=b[i][j];  
for(int i=0; i<a.length; i++)  
    System.out.print(a[i]+" ");  
//17 21 18
```



二维数组的长度就是
数组中包含的行数

实例 二维矩阵的转置

```
public class test
{
    public static void main(String[] args)
    {
        int a[][]={{ 1,2,3,4},{2,3,4,5},{3,4,5,6}};
        int b[][]= new int[4][3];
        int i,j;
        for(i=0;i<3;i++)           //利用两重循环进行转置
            for(j=0;j<4;j++)
                b[j][i]=a[i][j];
    }
}
```

```
for(i=0;i<4;i++)  
{ for(j=0;j<3;j++)  
    System.out.print(b[i][j]+" ");  
    System.out.println();  
}  
}
```
