Java数组

- What?Why?How?

本章概述

1.一维数组入门

①数组定义、特点、内存分配

②使用一维数组存储数据

③for-each循环

2.一维数组的应用

①查询元素

②数组类型做形参

③查询最大值最小值

④添加元素或删除元素

⑤冒泡排序

⑥Arrays工具类

⑦理解main（String args[]）

⑧可变参数

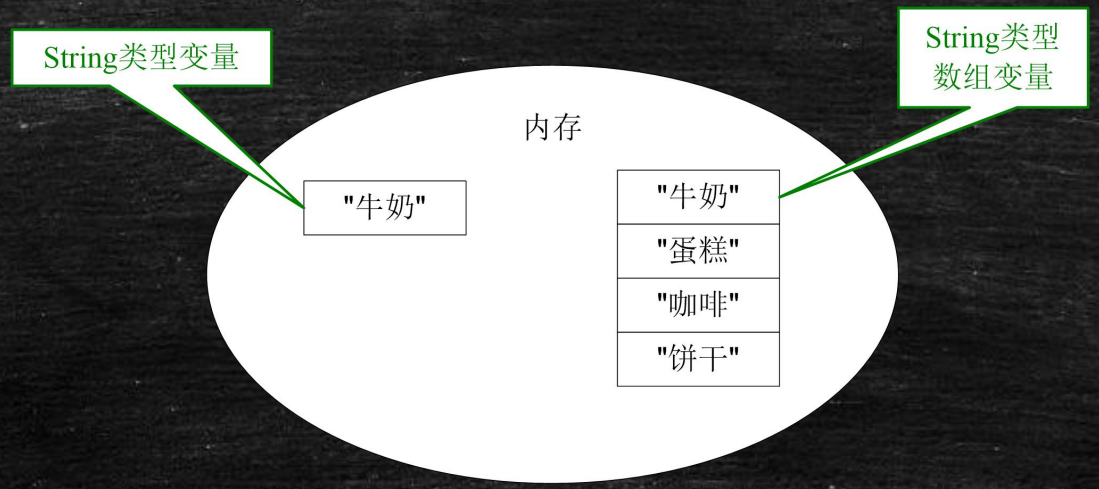
- 3.二维数组：二维数组含义、特点、内存分配、举例

创建数组（1）

- 数组是**相同类型数据**的**有序**集合.
  - 相同类型的若干个数据，按照一定先后次序排列组合而成。
  - 其中，每一个数据称作一个数组元素
  - 每个数组元素可以通过一个下标来访问它们.
- 数组特点：
  - 其长度是确定的。数组一旦被创建，它的大小就是不可以改变的。
  - 其元素必须是相同类型，不允许出现混合类型。
  - 数组中的元素可以是任何数据类型，包括基本类型和引用类型。
- 数组属引用类型
  - length, elements of the array

数组概述 B

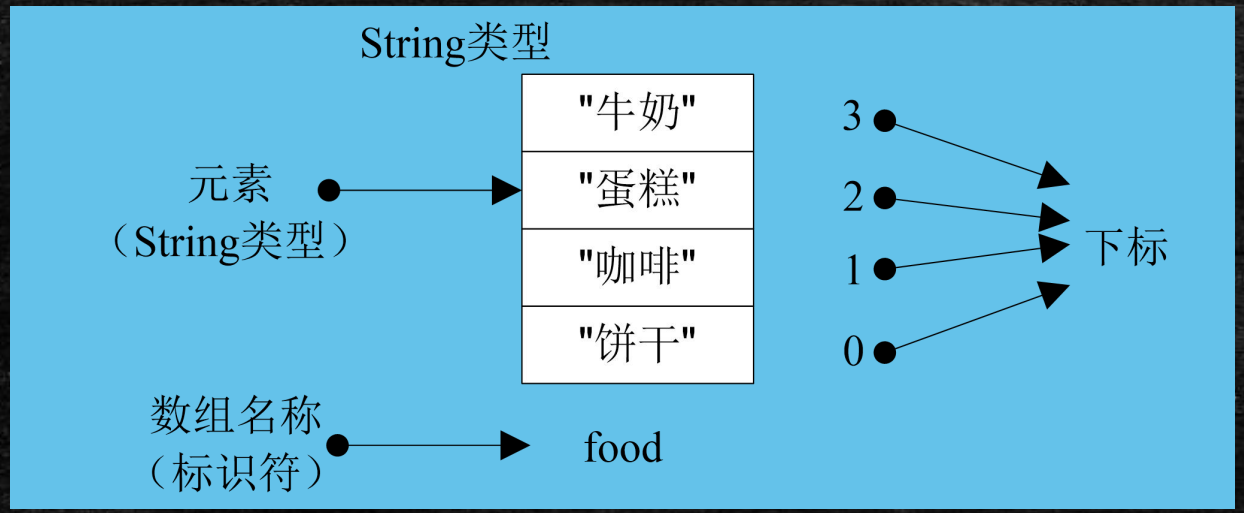
- 数组是一个变量，存储相同数据类型的一组数据



声明一个变量就是在内存空间划出一块合适的空间
声明一个数组就是在内存空间划出一串连续的空间

数组概述 C

- 数组只有一个名称，即标识符
- 元素下标标明了元素在数组中的位置，从0开始
- 数组中的每个元素都可以通过下标来访问
- 数组长度固定不变，避免数组越界



小结

- 下列哪组数据能存储在数组中？数组的类型是什么？
  - “刘星”，“夏雨”，“夏雪”
  - 8，98，“c”，23
  - 98.1，341.2，34.3

如何使用数组

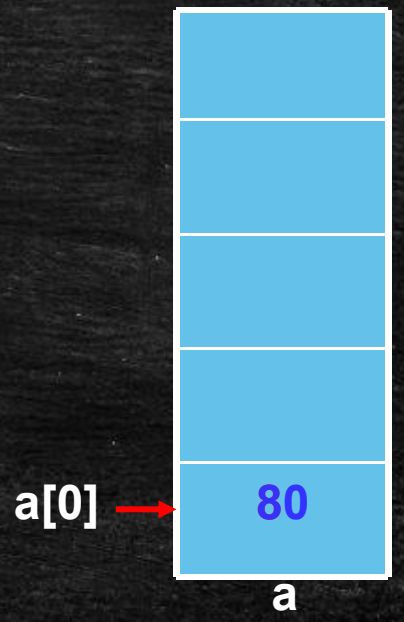
- 使用数组四步走：

1、声明数组 `int[ ] a;`

2、分配空间 `a = new int[5];`

3、赋值 `a [0] = 8;`

4、处理数据 `a [0] = a[0] * 10;`



声明数组

- 声明数组：告诉计算机数据类型是什么

```
int[ ] score1;          //Java成绩
int score2[ ];          //C#成绩
String[ ] name;         //学生姓名
```

声明数组时不规定数组长度

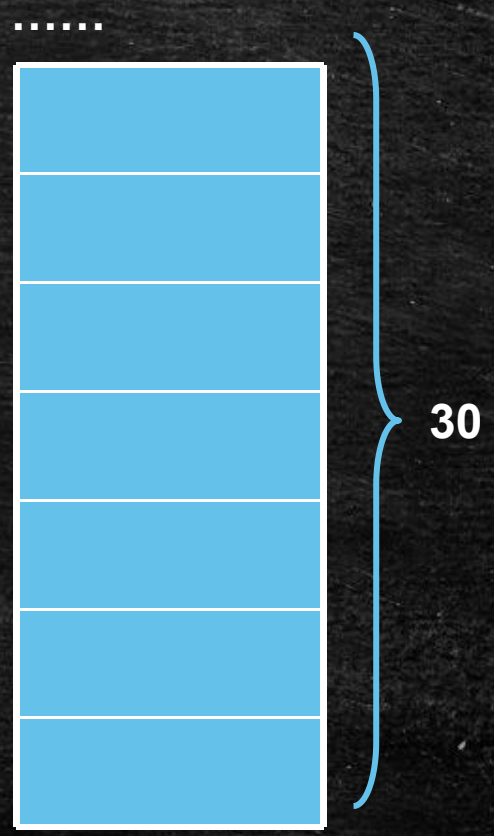
```
数据类型      数组名[ ];
```

```
数据类型[ ]   数组名 ;
```

分配空间

- 分配空间：告诉计算机分配几个连续的空间

```
score = new int[30];
avgAge = new int[6];
name = new String[30];
```



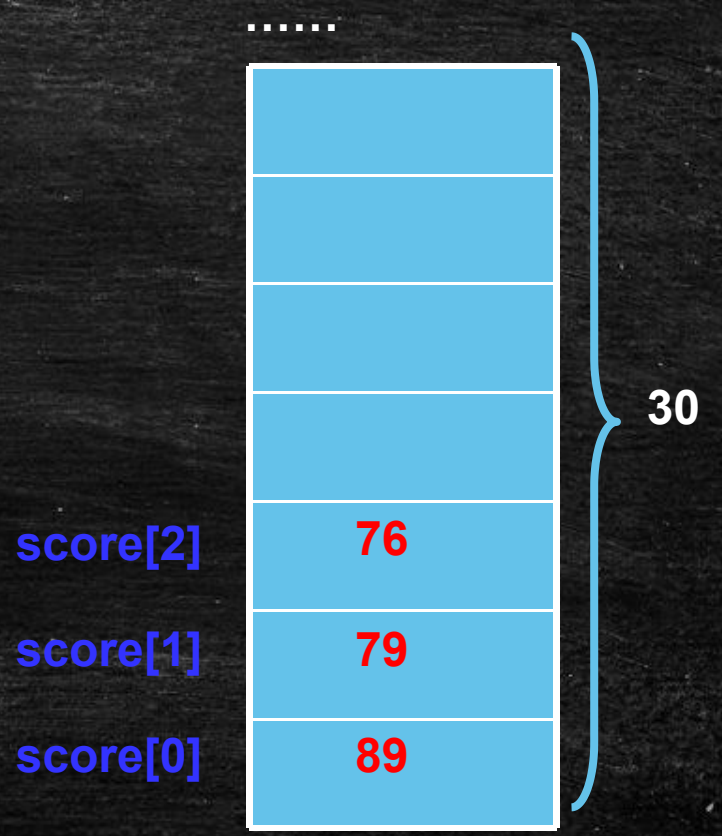
声明数组并分配空间

```
数据类型[ ] 数组名 = new 数据类型[大小] ;
```

数组赋值

- **赋值**：向分配的格子里放数据

```
score[0] = 89;
score[1] = 79;
score[2] = 76;
……
```



太麻烦！能不能一起赋值？

数组赋值

- 方法1: 边声明边赋值

```
int[ ] score = {89, 79, 76};
```

```
int[ ] score = new int[ ]{89, 79, 76};
```

不能指定数组长度

- 方法2：动态地从键盘录入信息并赋值

```
Scanner input = new Scanner(System.in);
for(int i = 0; i < 30; i ++){
    score[i] = input.nextInt();
}
```

处理数据 4

- 对数据进行处理：计算5位学生的平均分

```
int [ ] score = {60, 80, 90, 70, 85};
double avg;
avg = (score[0] + score[1] + score[2] + score[3] + score[4])/5;
```

访问数组成员：使用“标识符[下标]”

```
int [ ] score = {60, 80, 90, 70, 85};
int sum = 0;
double avg;
for(int i = 0; i < score.length; i++){
    sum = sum + score[i];
}
avg = sum / score.length;
```

数组的length属性

访问成员

成绩单
60
80
90
70
85

使用数组求平均分

```
public static void main(String[ ] args) {
    int[ ] scores = new int[5];      //成绩数组
    int sum = 0;                     //成绩总和
    Scanner input = new Scanner(System.in);
    System.out.println("请输入5位学员的成绩：");
    for(int i = 0; i < scores.length; i++){
        scores[i] = input.nextInt();
        sum = sum + scores[i];   //成绩累加
    }
    System.out.println("平均分是：" + (double)sum/scores.length);
}
```

常见错误3-1

```
public class ErrorDemo1 {
    public static void main(String[ ] args){
        int[ ] score = new int[ ];
        score[0] = 89;
        score[1] = 63;
        System.out.println(score[0]);
    }
}
```

编译出错，没有写明数组的大小

常见错误3-2

```
public class ErrorDemo2 {
    public static void main(String[ ] args) {
        int[ ] scores = new int[2];
        scores[0] = 90;
        scores[1] = 85;
        scores[2] = 65;
        System.out.println(scores[2]);
    }
}
```

编译出错，数组越界

常见错误3-3

```
public static void main(String[ ] args){
        int[ ] score = new int[5];
        score = {60, 80, 90, 70, 85};

        int[ ] score2;
        score2 = {60, 80, 90, 70, 85};
}
```

编译出错，创建数组并赋值的方式必须在一条语句中完成

一维数组的声明 A

- **一维数组的声明方式**有两种：

```
type[]  arr_name;
type  arr_name[];
```

- **例如：**

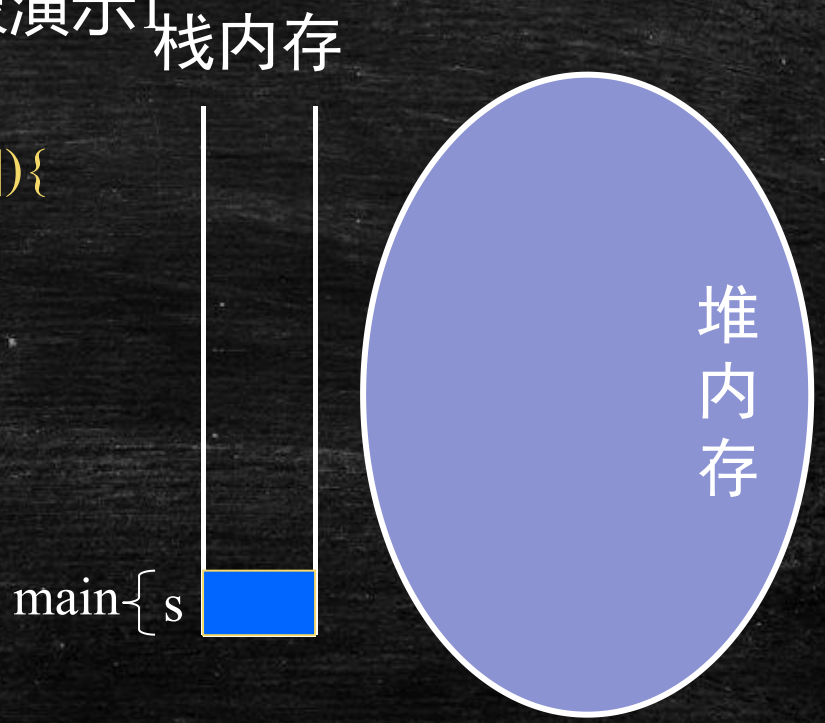
```
int[] intArrays;    int intArrays[];

double[] doubleArrays;

Person[] pArrays;

String[] strArrays;
```

创建数组（1）

- Java中使用关键字new 创建数组对象
- 创建基本数据类型一维数组对象演示1

```
public class Test{
    public static void main(String args[]){
        int[] s = null;
        s = new int[10];
        for ( int i=0; i<10; i++ ) {
            s[i] =2*i+1;
            System.out.println(s[i]);
        }
    }
}
```

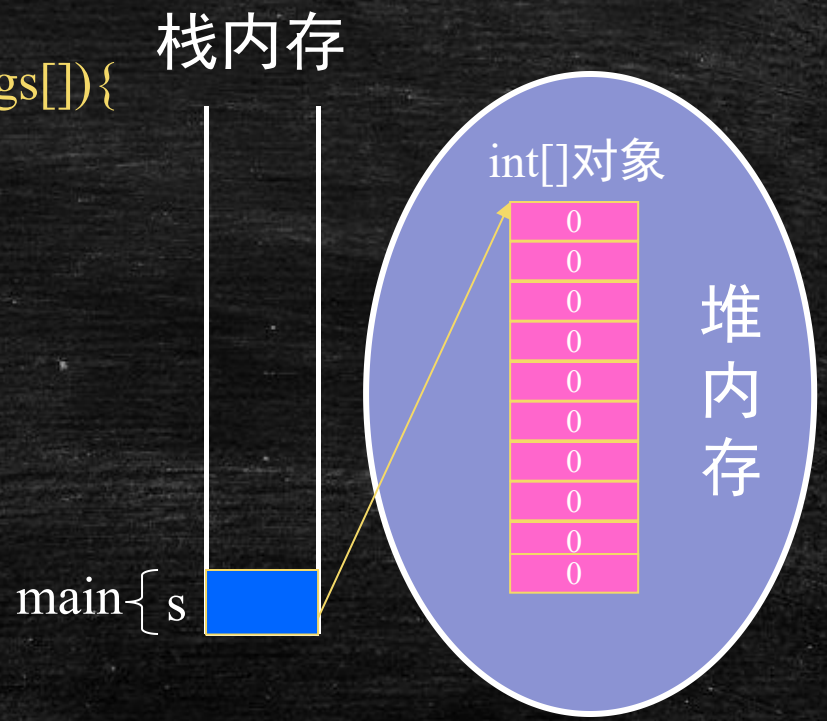
处内存状态



创建数组（2）

- 创建基本数据类型一维数组对象演示2

```
public class Test{
    public static void main(String args[]){
        int[] s = null;
        s = new int[10];
        for ( int i=0; i<10; i++ ) {
            s[i] =2*i+1;
            System.out.println(s[i]);
        }
    }
}
```

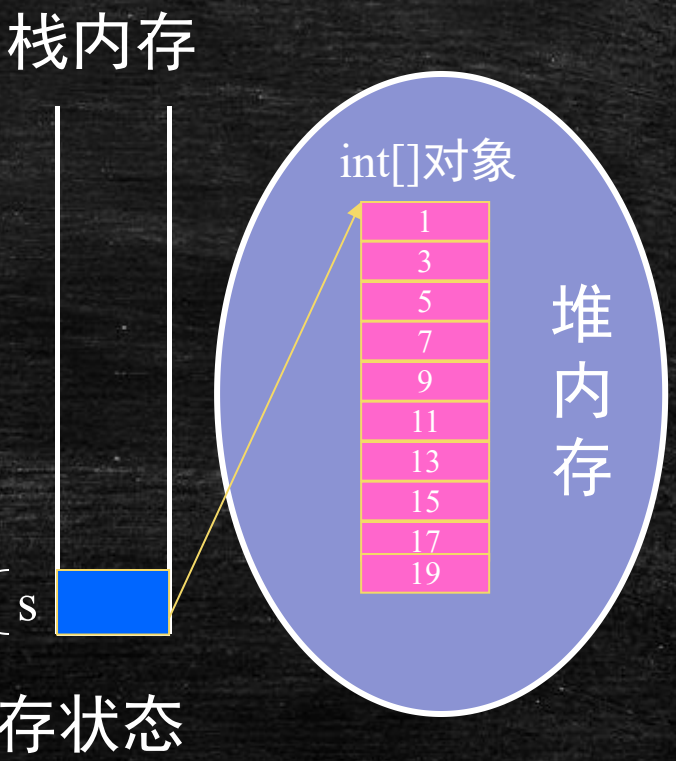


处内存状态

创建数组（3）

- 创建基本数据类型一维数组对象演示3

```
public class Test{
    public static void main(String args[]){
        int[] s = null;
        s = new int[10];
        for ( int i=0; i<10; i++ ) {
            s[i] =2*i+1;
            System.out.println(s[i]);
        }
    }
}
```



处内存状态

数组初始化

- 动态初始化
- 数组定义与为数组元素分配空间并赋值的操作分开进行

```
int  a[] = null;
a  = new int[3];
a[0] = 3;
a[1] = 9;
a[2] = 8;
```

数组初始化

- 静态初始化：
- 除了用new关键字来产生数组以外，还可以直接在定义数组的同时就为数组元素分配空间并赋值。
  - 格式：类型 [] 数组名 = {元素1[, 元素2 ……]};
    - int [] a = {1, 2, 3, 4, 5};

```
public class Test {
    public static void main(String args[]) {
        int [] a = { 3, 5, 7 };
}
}
```

数组元素的默认初始化 A

- 数组是引用类型，它的元素相当于类的实例变量，因此数组一经分配空间，其中的每个元素也被按照实例变量同样的方式被隐式初始化。

```
public class ArrayTest3 {
    public static void main(String args[]) {
         int a[] = new int[2];
        boolean [] b = new boolean[2];
        String[] s = new String[2];
        for(int i = 0; i < 2; i++)
                System.out.println(a[i]);
        for(int i = 0; i < 2; i++)
                System.out.println(b[i]);
        for(int i = 0; i < 2; i++)
                System.out.println(s[i]);
    }
}
```

```
0
0
false
false
null
null
```

数组的界限

- 定义并用运算符new为之分配空间后，才可以引用数组中的每个元素；
- 数组元素的引用方式：arrayName[index]
  - ✓index为数组元素下标，可以是整型常量或整型表达式。如a[3]，b[i]，c[6*i];
  - ✓数组元素下标从0开始；长度为n的数组合法下标取值范围： 0 ~ n-1；

- 每个数组都有一个属性length指明它的长度，例如：a.length 指明数组a的长度(元素个数)；
  - ✓ 数组的长度：数组名.length

- 起点和终点
  - ✓起点： 数组名[0]
  - ✓终点： 数组名[length-1]

```
int[] i = {4, 56, 78, 9,
    34};
i.length → 5
i[0] → 4
i[length-1]=i[4]→34
i[a] 若a>4 则???
```

课堂练习（20分钟）

1. 编写一应用程序实现下述功能：创建一基本(primitive)数据类型的数组并输出各数组元素的值。例如：

```
char[] s;
 s = new char[26];
 for ( int i=0; i<26; i++ ) {
        s[i] = (char) ('A' + i);
        System.out.println(s[i]);
        // System.out.println("s[" + i + "]=" + s[i]);
 }
```

2. 编写一应用程序练习数组对象的两种初始化方式，并输出各元素的值。
3. 编写程序，练习使用数组类型对象的**length**属性，测试并体会数组元素的默认初始化机制；
4. 有一个数列，8,4,2,1,23,344,1，（1）循环输出数列的值（2）求数列中所有数值的和（3）猜数游戏：从键盘上任意输入数字，判断数组中是否包含此数

二维数组

二维数组举例：

int [][] a = {{1,2},{3,4,0,9},{5,6,7}};

Java中多维数组不必须是规则矩阵形式

i \ j	j = 0	j = 1	j = 2	j = 3
i = 0	1	2		
i = 1	3	4	0	9
i = 2	5	6	7	

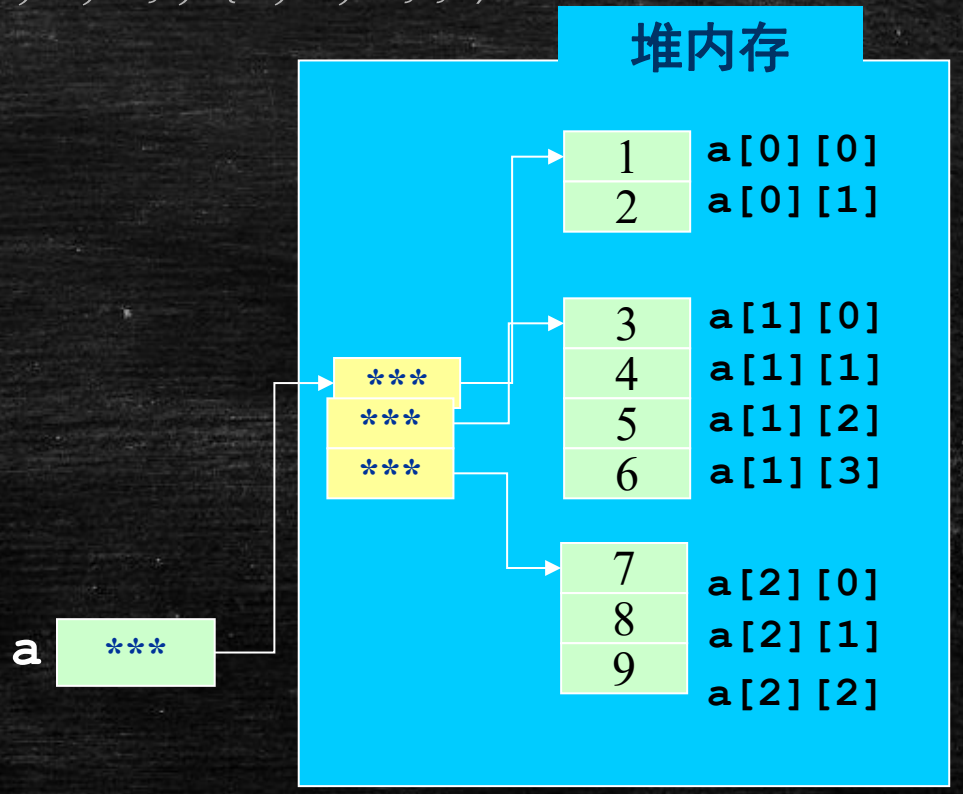
二维数组

- 二维数组可以看成以数组为元素的数组。例如：

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

- Java中多维数组的声明和初始化应按从高维到低维的顺序进行，例如：

```
int [][]  a= new int[3][];
a[0] = new int[2];
a[1] = new int[4];
a[2] = new int[3];
int t1[][] = new int[][4]; //非法
```



二维数组初始化

- Declare, create and initiate in the same time：

```
int intA[][] = {{1,2},{2,3},{3,4,5}};
int intB[3][2] = {{1,2},{2,3},{4,5}};//非法
```

- Declare, create and initiate separately：

```
int a[][] = new int[3][5];

int b[][] = new int[3][] ;

b[0] = new int[2];

b[1] = new int[3];

b[2] = new int[5];
```

课堂练习

编写一应用程序实现下述功能：创建一基本(primitive)数据类型的二维数组并输出各数组元素的值。例如：

```
    ……
int a[][] = {{1,2},{2,3,4,5},{5,6,7}};
for(int i=0;i<a.length;i++) {
        for(int j=0;j<a[i].length;j++) {
            System.out.println(intArray1[i][j]);
        }
}
```

数组的拷贝

- 使用java.lang.System类的静态方法

```
public static void arraycopy
            (Object src,int srcPos,Object dest,
                        int destPos,int length)
```

- 可以用于数组src从第srcPos项元素开始的length个元素拷贝到目标数组从destPos项开始的length个位置。
- 如果源数据数目超过目标数组边界会抛出

  IndexOutOfBoundsException 异常。

数组的拷贝举例

```
public class ArrayTest7 {
  public static void main(String args[]) {
    String[] s = {"Mircosoft","IBM","Sun","Oracle","Apple"};
    String[] sBak = new String[6];
    System.arraycopy(s,0,sBak,0,s.length);
        for(int i=0;i<sBak.length;i++){
                System.out.print(sBak[i]+" ");
        }
        System.out.println();
    int[][] intArray = {{1,2},{1,2,3},{3,4}};
    int[][] intArrayBak = new int[3][];
    System.arraycopy(intArray,0,intArrayBak,0,intArray.length);
    intArrayBak[2][1] = 100;
    for(int i = 0;i<intArray.length;i++){
        for(int j =0;j<intArray[i].length;j++){
            System.out.print(intArray[i][j]+"  ");
        }
        System.out.println();
    }
  }
}
```

命令行参数

- JAVA应用程序的主方法(程序的入口)
  - -public static void main (String args[]) {…}
  - -public static void main (String[] args) {…}
- 命令行参数
  - -在启动Java应用程序时可以一次性地向应用程序中传递0~多个参数----命令行参数
  - -命令行参数使用格式:

```
java ClassName  lisa  "bily"  "Mr Brown“
```

  由参数args接收
  - -空格将参数分开
  - -若参数包含空格，用双引号引起来

命令行参数用法举例

```
public class Test {
    public static void main(String[] args) {
    for ( int i = 0; i < args.length; i++ ) {
        System.out.println("args[" + i + "] = " + args[i]);
    }
    }
}
```

```
//运行程序
java Test  lisa  "bily"  "Mr Brown"
//输出结果:
```

- args[0] = lisa
- args[1] = bily
- args[2] = Mr Brown

Java.uitl.Arrays

- 该类提供了关于数组操作的API.
  - 打印数组----toString方法。
  - 比较两个数组是否相同----equals方法。
  - 数组排序----sort方法。
  - 数组查找----binarySearch 方法

总结

- 一维数组入门
  - 数组的特点：长度固定，连续空间，存储同一种类型数据
  - 数组内存分配图
  - for-each循环：简单、主要用于遍历操作
- 一维数组的应用
  - 数组优缺点
    - 优点：按照索引查询效率高
    - 缺点：添加删除元素效率低；按照内容查询效率低（无序）
  - 冒泡排序：基本的排序算法，理解排序规则，实现并完善排序代码
  - 数组类型做形参
- 二维数组：
  - 实质是每个元素是一维数组的一维数组；二维数组内存分配图