今日学习目标

* 能够使用Random生成随机数
* 能够使用两种格式定义数组 int[] arr; int arr[];
* 能够理解数组的动态初始化并使用 举例: int[] arr = new int[3];
* 能够访问数组中的元素 arr[0],arr[1],arr[2] ....
* 能够说出Java中的内存分配
* 能够理解数组内存图解
* 能够理解数组的静态初始化并使用 int[] arr = {1,2,3,4,5};
* 能够理解数组操作的两个常见小问题 : 数组索引越界异常,空指针异常
* 能够完成数组的遍历 获取数组长度: 数组名.length
* 能够完成数组的求最值
* 能够定义二维数组
* 能够遍历二维数组

# 随机数Random

## 概述

* + 用于产生一个随机数
* 使用步骤(和Scanner类似)
  + 导包
    - import java.util.Random;
  + 创建对象
    - Random r = new Random();
  + 获取随机数
    - int number = r.nextInt(10);
    - 产生的数据在0到10之间，包括0，不包括10。
    - 括号里面的10是可以变化的，如果是100，就是0-100之间的数据
* 练习
  + 获取**[**0-10**)**之间的随机数
  + 获取**[**1-100**]**之间的随机数

**public** **class** Demo01Random {

**public** **static** **void** main(String[] args) {

// 创建对象

Random r = **new** Random();

**for** (**int** x = 0; x < 10; x++) {

// 获取随机数

**int** number = r.nextInt(10);

// 输出随机数

System.***out***.println("number:" + number);

}

System.***out***.println("--------------------");

// 如何获取到一个1-100之间的随机数呢?

**int** i = r.nextInt(100) + 1;

System.***out***.println("i:" + i);

}

}

## 猜数字小游戏案例

系统产生一个1-100之间的随机数，请猜出这个数据是多少

**public** **class** Demo02Random {

**public** **static** **void** main(String[] args) {

// 系统产生一个随机数1-100之间的。

Random r = **new** Random();

**int** number = r.nextInt(100) + 1;

**while**(**true**){

// 键盘录入我们要猜的数据

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("请输入你要猜的数字(1-100)：");

**int** guessNumber = sc.nextInt();

// 比较这两个数据(用if语句)

**if** (guessNumber > number) {

System.***out***.println("你猜的数据" + guessNumber + "大了");

} **else** **if** (guessNumber < number) {

System.***out***.println("你猜的数据" + guessNumber + "小了");

} **else** {

System.***out***.println("恭喜你,猜中了");

**break**;

}

}

}

}

# 数组

## 数组概述

* 需求：现在需要统计某公司员工的工资情况，例如计算平均工资、找到最高工资等。假设该公司有80名员工，用前面所学的知识，程序首先需要声明80个变量来分别记住每位员工的工资，然后在进行操作，这样做会显得很麻烦。为了解决这种问题，Java就提供了数组供我们使用。
* 那么数组到底是什么呢?有什么特点呢?通过上面的分析：我们可以得到如下两句话：
  + 数组是存储多个变量(元素)的容器
  + 这多个变量的数据类型要一致

## 数组定义格式

* 数组概念
  + 数组是存储同一种数据类型多个元素的容器。
  + 数组既可以存储基本数据类型，也可以存储引用数据类型。
* 数组的定义格式
  + 格式1：数据类型[] 数组名;
  + 格式2：数据类型 数组名[];
  + 注意：这两种定义做完了，数组中是没有元素值的。如何对数组的元素进行初始化呢?

## -数组初始化

* 概述
  + Java中的数组必须先初始化,然后才能使用。
  + 所谓初始化：就是为数组中的数组元素分配内存空间，并为每个数组元素赋值。
* 数组的初始化方式
  + 第一种初始化（动态）：初始化时只指定数组长度，由系统为数组分配初始值。
  + 第二种初始化（静态）：初始化时指定每个数组元素的初始值，由系统决定数组长度。

## 动态初始化

* 初始化时只指定数组长度，由系统为数组分配初始值。
  + 格式：数据类型[] 数组名 = new 数据类型[数组长度];
  + 数组长度其实就是数组中元素的个数。
  + 举例：
    - int[] arr = new int[3];
    - 解释：定义了一个int类型的数组，这个数组中可以存放3个int类型的值。

## 访问数组元素

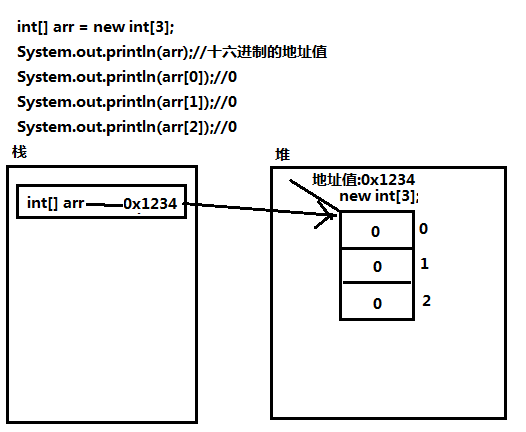
* 直接打印数组名,打印的是数组堆内存中的地址
* 数组中的每个元素都是有编号的，编号是从0开始的，最大的编号就是数组的长度-1
* 用数组名和编号的配合我们就可以获取数组中的指定编号的元素
* 这个编号的专业叫法：索引
* 格式：数组名[编号] -- 数组名[索引]
  + 数组名称为arr,索引为0,1,2
  + 获取数组中元素的格式:
    - arr[0]获取数组中的第一个元素
    - arr[1]获取数组中的第二个元素
    - arr[2]获取数组中的第三个元素

## Java中的内存分配

Java 程序在运行时，需要在内存中的分配空间。

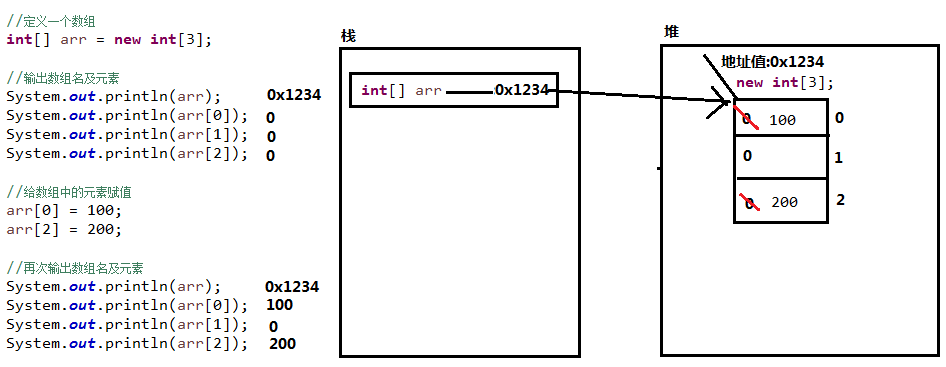
为了提高运算效率，就对空间进行了不同区域的划分，因为每一片区域都有特定的处理数据方式和内存管理方式。

* 栈 存储局部变量
  + 局部变量是定义在方法中或者语句中的变量
  + 局部变量使用完毕，立即回收
* 堆 存储new出来的东西
  + 每一个对象都有首地址值,是一个十六进制的地址
  + 每一个对象内的数据都有默认值
    - byte,short,int,long 0
    - float,double 0.0
    - char ‘\u0000’,一个空格
    - boolean false
    - 引用类型 null
  + 使用完毕后，会被垃圾回收器空闲的时候回收。
* 方法区 (面向对象进阶讲)
* 本地方法区 (和系统相关)
* 寄存器 (给CPU使用)
* 画图说明:数组的内存图

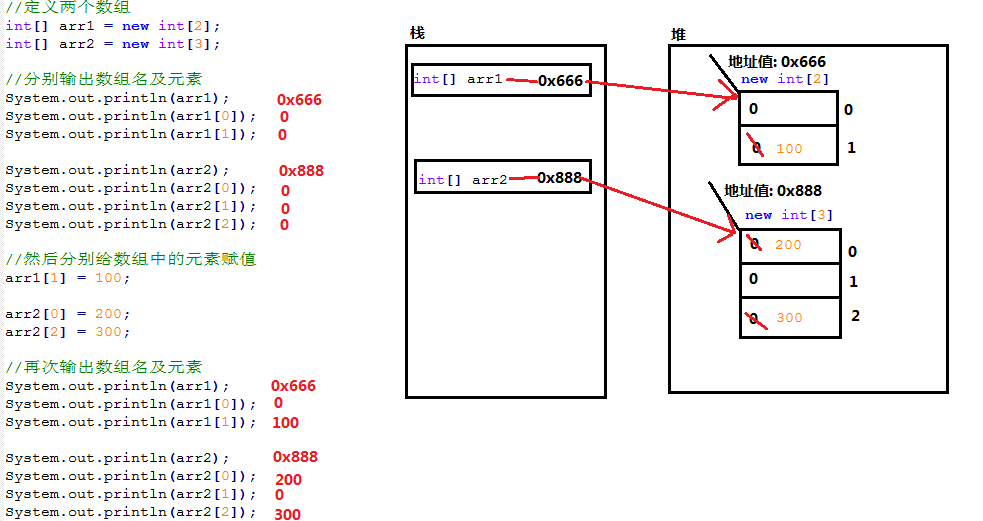


## 数组内存图解

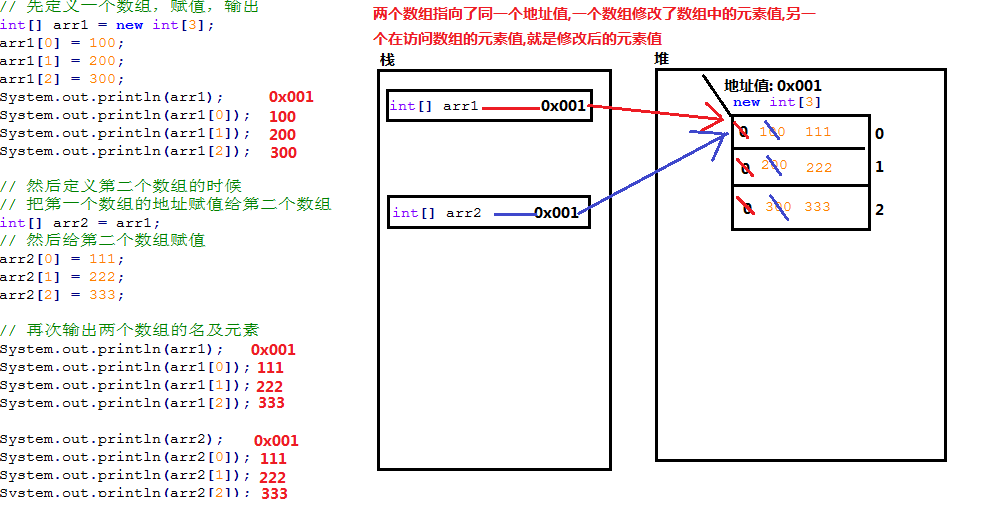
* 定义一个数组，输出数组名及元素。然后给数组中的元素赋值，再次输出数组名及元素



* 定义两个数组，分别输出数组名及元素。然后分别给数组中的元素赋值，分别再次输出数组名及元素。



* 定义两个数组，先定义一个数组赋值，输出。然后定义第二个数组的时候把第一个数组的地址赋值给第二个数组。然后给第二个数组赋值，再次输出两个数组的名及元素。



就相当于两个人合租,一个人把房间打扫了,另一个人住的也是干净的.

## 静态初始化

* 概述

初始化时指定每个数组元素的初始值，由系统决定数组长度。

* 格式

数据类型[] 数组名 = new 数据类型[]{元素值1,元素值2,…};

* 举例

int[] arr = new int[]{1,2,3};

解释：定义了一个int类型的数组，这个数组中可以存放3个int类型的值，并且值分别是1,2,3。

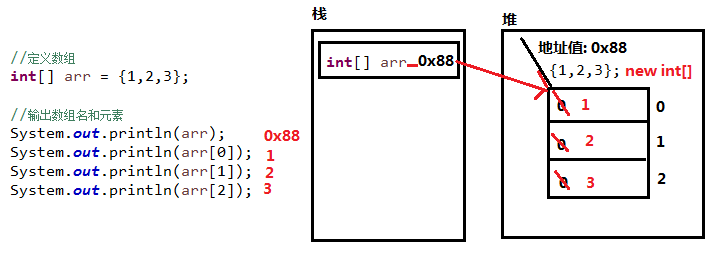
* 简化格式

数据类型[] 数组名 = {元素1,元素2,...};

* 简化格式举例

int[] arr = {1,2,3};

* 内存图



* 注意:
  + 动态初始化可以先定义再初始化.



* + 静态初始化必须定义的同时初始化.



## 数组操作的两个常见小问题

* 数组索引越界异常ArrayIndexOfBoundsException

访问到了数组中的不存在的索引时发生。

* 空指针异常NullPointerException

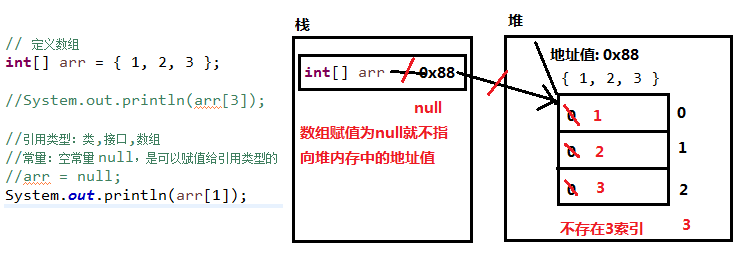
数组引用没有指向对象，却在操作对象中的元素时。

* 为什么要记住这些小问题呢?

因为写程序不仅仅是把代码写出来，还包括出现问题后要能够及时的把问题解决。

而排错也应该是我们的基本功。所以，我们也应该记住一些常见的问题，以及出错原因和解决方案。

* 内存图



## 练习

### 遍历数组

* + 依次输出数组中的每一个元素
  + 获取数组长度：数组名.length

**public** **class** Demo07ArrayTest {

**public** **static** **void** main(String[] args) {

// 定义数组

**int**[] arr = { 11, 22, 33, 44, 55 };

// 原始做法

System.***out***.println(arr[0]);

System.***out***.println(arr[1]);

System.***out***.println(arr[2]);

System.***out***.println(arr[3]);

System.***out***.println(arr[4]);

System.***out***.println("--------------------");

// 用for循环改进

**for** (**int** x = 0; x < 5; x++) {

System.***out***.println(arr[x]);

}

System.***out***.println("--------------------");

//为了解决我们去数数组中元素个数的问题，数组就提供了一个属性：length

//用于获取数组的长度

//格式：数组名.length

System.***out***.println("数组共有："+arr.length+"个元素");

System.***out***.println("--------------------");

**for**(**int** x=0; x<arr.length; x++) {

System.***out***.println(arr[x]);

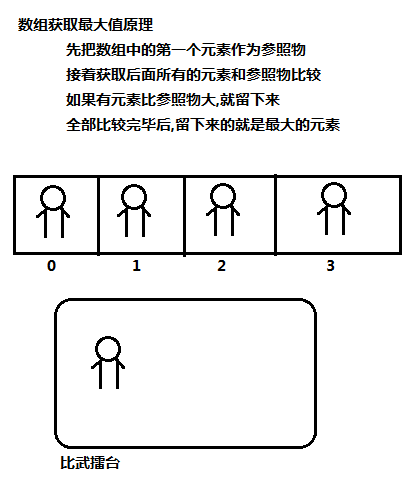
}

}

}

### 获取最值

* + 获取数组中的最大值与最小值



**public** **class** Demo08ArrayTest {

**public** **static** **void** main(String[] args) {

//定义数组

**int**[] arr = {12,98,45,73,60};

//定义参照物

**int** max = arr[0];

//遍历数组，获取除了0以外的所有元素，进行比较

**for**(**int** x=1; x<arr.length; x++) {

**if**(arr[x] > max) {

max = arr[x];

}

}

System.***out***.println("数组中的最大值是："+max);

}

}

## 二维数组

### 概述

* 我们传智播客的Java基础班每个班有很多个学生，所以，可以用数组来存储，而我们又同时有很多个Java基础班。这个也应该用一个数组来存储。如何来表示这样的数据呢?Java就提供了二维数组供我们使用。
* 由此可见：其实二维数组其实就是一个元素为一维数组的数组。

### 基本使用

* 二维数组定义格式
  + 数据类型[][] 数组名;
  + 数据类型 数组名[][]; （不推荐）
  + 数据类型[] 数组名[]; （不推荐）
* 初始化方式
  + 数据类型[][] 变量名 = new 数据类型[m][n];
    - m： 二维数组的长度（二维数组中一维数组的个数）
    - n： 二维数组中一维数组的长度
  + 数据类型[][] 变量名 = new 数据类型[][]{{元素…},{元素…},{元素…}};
  + 简化版格式：数据类型[][] 变量名 = {{元素…},{元素…},{元素…}};

**public** **class** Demo01ArrayArray {

**public** **static** **void** main(String[] args) {

// 数据类型[][] 数组名 = {{元素...},{元素...},{元素...},...};

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

System.***out***.println(arr); // [[I@104c575

System.***out***.println(arr.length); // 二维数组中的一维数组的个数

System.***out***.println(arr[0]);// [I@3fa5ac

System.***out***.println(arr[0].length);

System.***out***.println(arr[1]);// [I@95cfbe

System.***out***.println(arr[2]);// [I@179dce4

//我如何获取到一个二维数组的元素呢?

System.***out***.println(arr[0][0]);

System.***out***.println(arr[1][1]);

System.***out***.println(arr[2][0]);

}

}

### 二维数组的遍历

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

**public** **class** Demo02ArrayArrayTest {

**public** **static** **void** main(String[] args) {

// 定义二维数组

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

// 二维数组中的一维数组名称：二维数组名[索引]

// arr[0] 其实就是二维数组中的第一个一维数组的名称

// arr[1] 其实就是二维数组中的第二个一维数组的名称

// arr[2] 其实就是二维数组中的第三个一维数组的名称

// for (int x = 0; x < arr[0].length; x++) {

// System.out.println(arr[0][x]);

// }

// System.out.println("hello");//输出内容并换行

// System.out.println("world");//输出内容并换行

// System.out.print("hello");//输出内容不换行

// System.out.print("world");//输出内容不换行

/\*

// 第一个一维数组的元素

for (int x = 0; x < arr[0].length; x++) {

System.out.print(arr[0][x] + " ");

}

System.out.println();

// 第二个一维数组的元素

for (int x = 0; x < arr[1].length; x++) {

System.out.print(arr[1][x] + " ");

}

System.out.println();

// 第三个一维数组的元素

for (int x = 0; x < arr[2].length; x++) {

System.out.print(arr[2][x] + " ");

}

System.out.println();

\*/

/\*for(int y=0; y<3; y++) {

for (int x = 0; x < arr[y].length; x++) {

System.out.print(arr[y][x] + " ");

}

System.out.println();

}\*/

**for**(**int** y=0; y<arr.length; y++) {

**for** (**int** x = 0; x < arr[y].length; x++) {

System.***out***.print(arr[y][x] + " ");

}

System.***out***.println();//换行

}

}

}