# DAY05，一维二维数组，常见操作，基本和引用作为参数

## ██████05.01\_Java语言基础(数组概述和定义格式说明)(了解)

A:为什么要有数组(容器)

为了存储同种数据类型的多个值

B:数组概念

数组是存储**同一种数据类**型多个元素的集合。也可以看成是一个容器。

**数组既可以存储基本数据类型，也可以存储引用数据类型。(集合基本包装成对象,并且集合可以放置不同的类型对象)**

C:数组定义格式(1为动态初始化，2，3为静态初始化)

1, ` 例: int[] arr = new int[5];

变形: 数据类型 数组名字[] = new 数据类型[数组的长度(元素的个数)];

**2，数据类型[] 数组名 = {数据}； （数据之间用逗号进行隔开）声明和赋值同时进行的。这是最常用的。**

int[ ] arr;

arr[ ]={数据}； 这是不可以的,应该是一行。

3，数据类型[ ] 数组名 = new 元素的数据类型[ ]{元素1，元素2，元素3，......};注:第二个[ ]里不能填任何东西，填了就会编译报错。先声明后赋值。掌握这个

int[ ] arr;

**arr = new int[ ]{数据}; 这是可以的 是在匿名内部类的时候运用的.**

D:数组的特点。

1，数组中所有的元素都应该为同一种类型的元素。

2，数组有索引的，最小为0.最大为arr.length-1.

3，数组中的元素个数是固定的，也就是说从定义数组的时候就已经定下来了。

E：数组的属性。

1，数组的长度 = arr.leangth-1.

## ██05.02\_Java语言基础(数组的初始化动态初始化)(掌握)

A:什么是数组的初始化

就是为数组开辟连续的内存空间，并为每个数组元素赋予值

B:如何对数组进行初始化

a:动态初始化 只指定长度，由系统给出初始化值

int[] arr = new int[5];

b:静态初始化 给出初始化值，由系统决定长度。

c：上边两者不能混着使用，是错误的编写格式。

C:动态初始化的格式：

数据类型[] 数组名 = new 数据类型[数组长度];

D:案例演示

输出数组名称和数组元素

int x = 10;

x = 20;

System.out.println("x = " + x);

//数据类型[] 数组名 = new 数据类型[数组的长度];

int[] arr = new int[5]; //可以存储五个int类型的数据

/\*

左边:

int:数据类型

**[]:代表的数组,几个中括号就代表几维数组**

arr:合法的标识符

右边:

**new:创建新的实体或对象**

int:数据类型

[]:代表的数组

5:代表数组的长度

\*/`

----------

**整数类型:byte,short,int,long默认初始化值都是0**

**浮点类型:float,double默认初始化值都是0.0**

**布尔类型:boolean默认初始化值false**

**字符类型:char默认初始化值'\u0000'**

char在内存中占的两个字节,是16个二进制位，默认\u0000,每一个0其实代表的是16进制的0,

那么四个0就是代表16个二进制位引用数据：默认的是Null，也就空的什么都没有。

**[I@19bb25a**

**[ 代表是数组,几个就代表几维**

**I 代表是int类型**

**@ 是固定的**

**19bb25a代表的是数组的地址值，十六进制的地址。哈希地址。**

----------

int[] arr = new int[5]; //动态初始化,在内存中开辟连续的5块空间

5个空间都是有初始值的然后再将地址值给到arr，arr通过索引来访问。

System.out.println(arr[0]); //系统给出默认初始化值,整数类型的都是0

arr[0] = 10; //将值赋值给了索引位置。

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

**System.out.println(arr); //[I@19bb25a，这将打印出地址**。

## ██████05.03\_Java语言基础(Java中的内存分配以及栈和堆的区别)

A:栈(掌握)

存储局部变量 ，局部变量就是定义在方法声明上和方法中的变量。

B:堆(掌握)

存储new出来的数组或对象

C:方法区

面向对象部分讲解

D:本地方法区

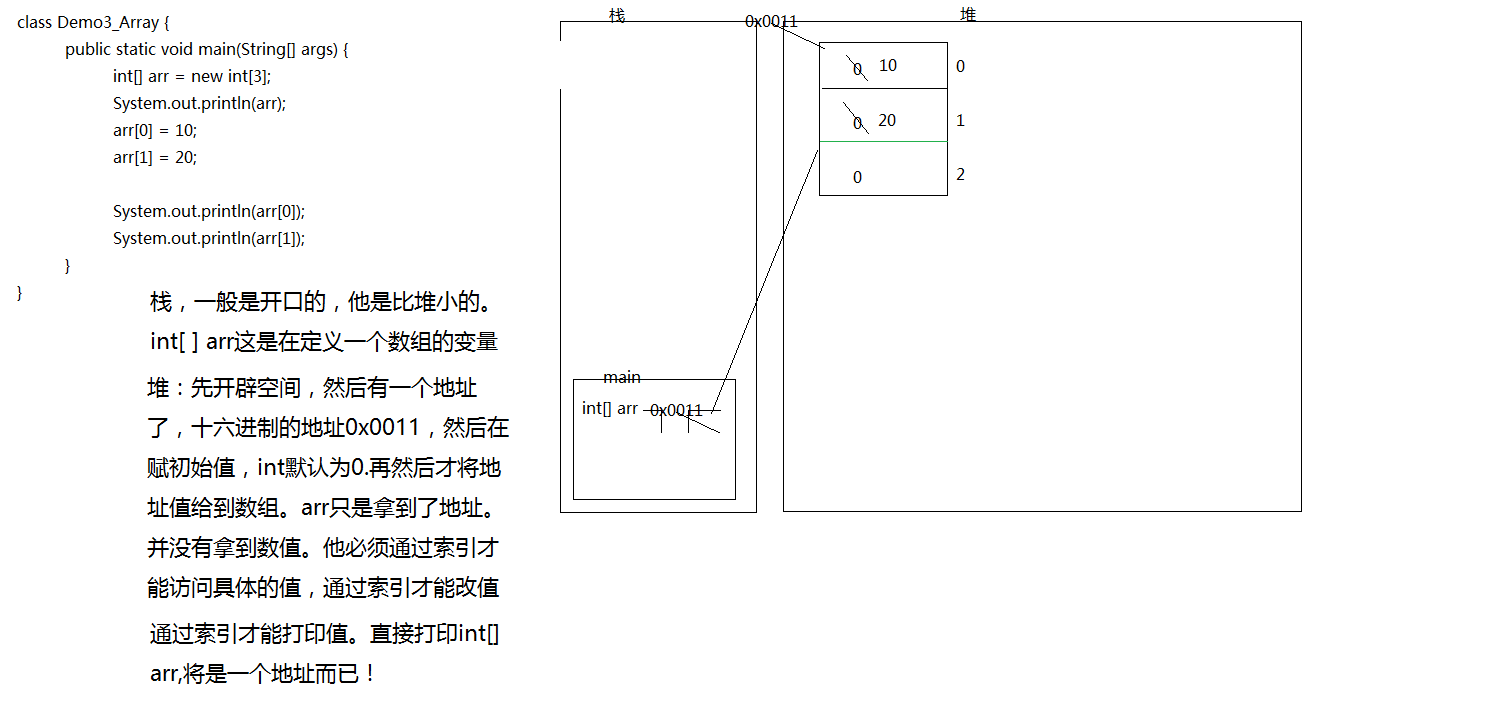
和系统相关 代码的存放。

E:寄存器

给CPU使用

## ██████05.04\_Java语言基础(数组的内存图解1一个数组)(掌握)

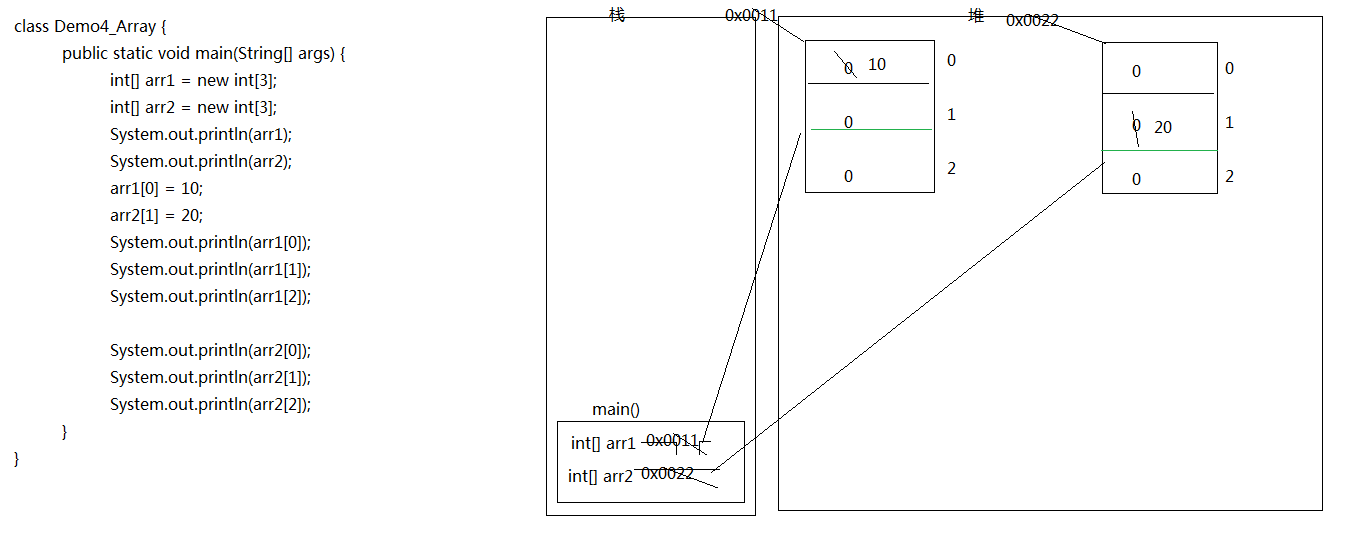
**A:画图演示**



## 05.05\_Java语言基础(数组的内存图解2二个数组)(了解)

**A:画图演示**

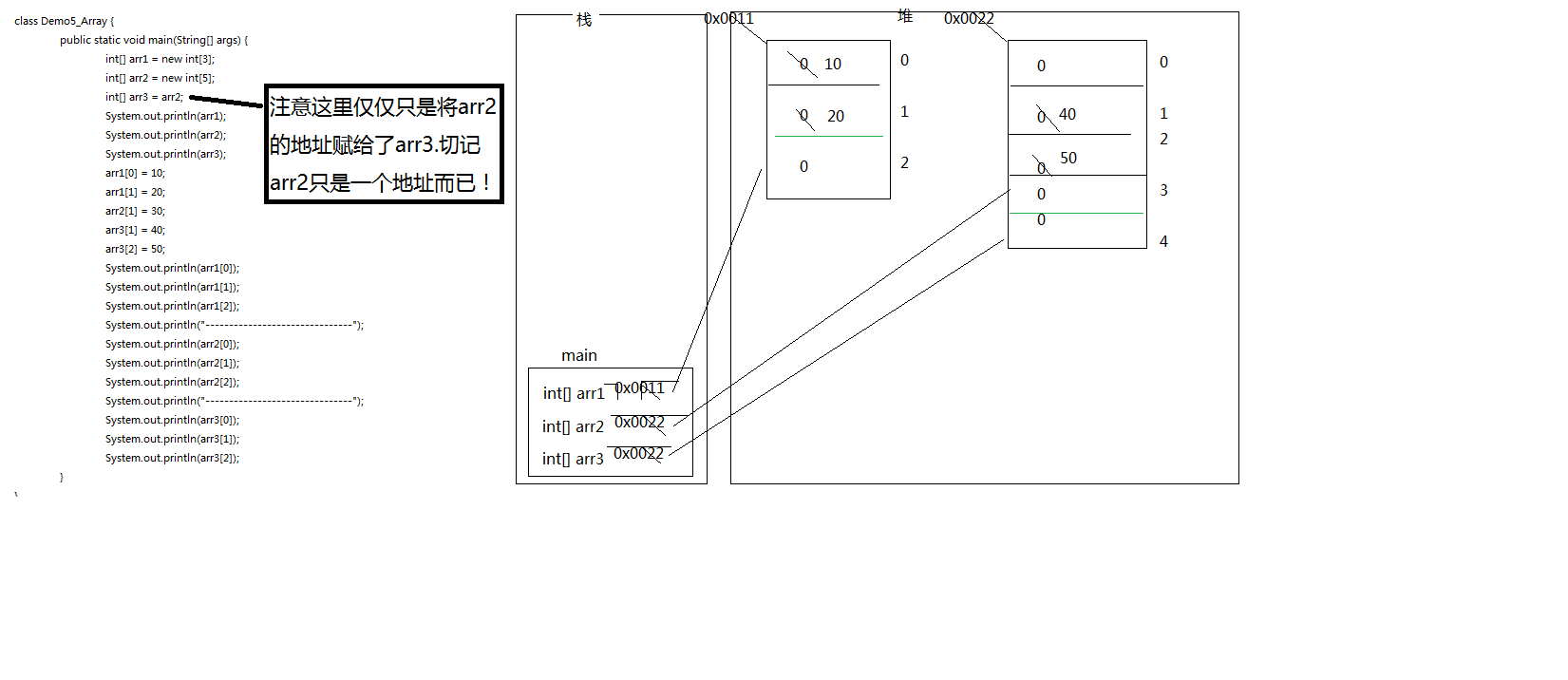
二个不同的数(两个数组的地址是不一样的)



## 05.06\_Java语言基础(数组的内存图解3三个引用两个数组)(了解)

**A:画图演示**

三个引用，有两个数组的引用指向同一个地址



## 05.07\_Java语言基础(数组的初始化静态初始化及内存图)(掌握)

A:静态初始化的格式：

格式：数据类型[] 数组名 = new 数据类型[]{元素1,元素2,…};

简化格式：

数据类型[] 数组名 = {元素1,元素2,…};

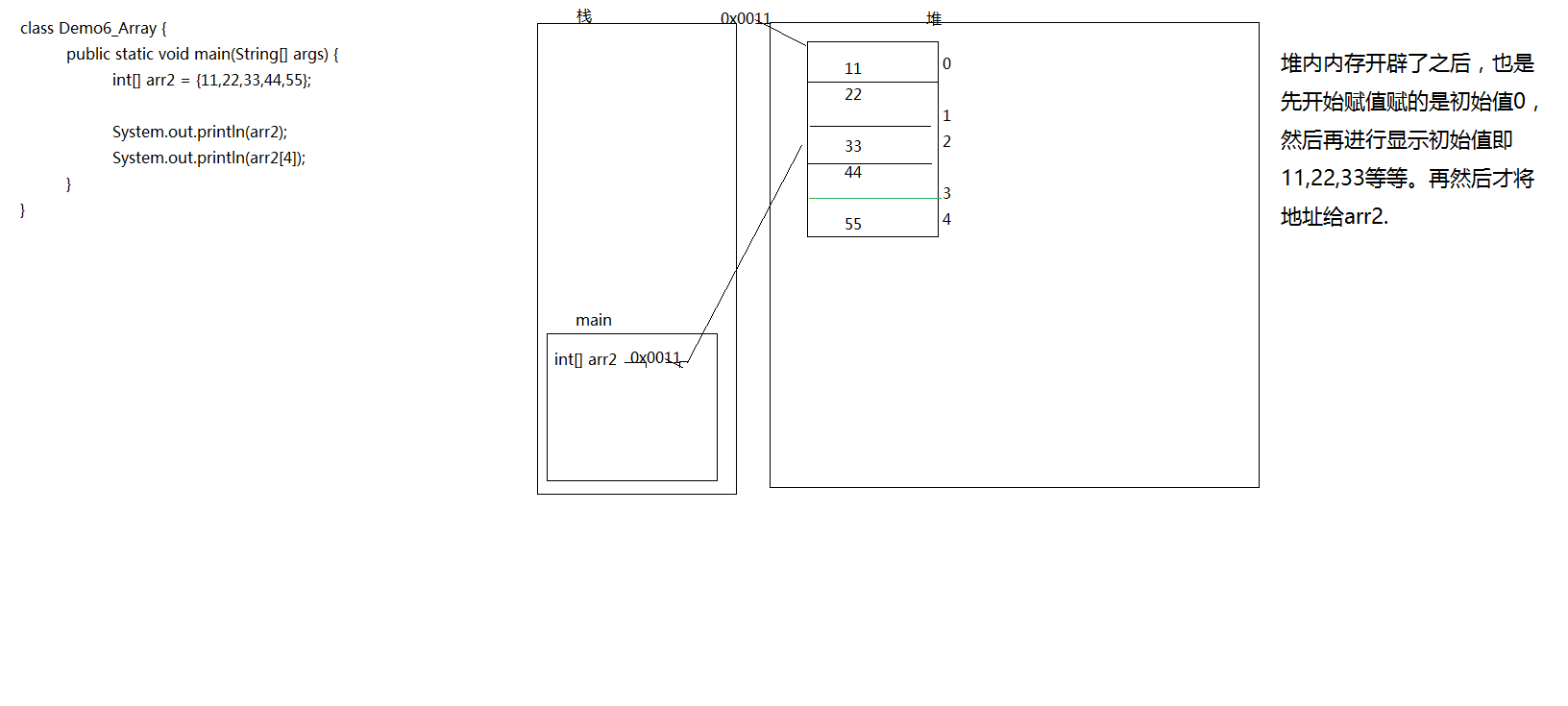
B:案例演示

对数组的解释

输出数组名称和数组元素

**C:画图**演示

一个数组

05.08\_Java语言基础(数组操作的两个常见小问题越界和空指针)(掌握)

A:案例演示

a:ArrayIndexOutOfBoundsException:数组索引越界异常

原因：你访问了不存在的索引。

System.out.println(arr[-1]);//当访问数组中不存在的索引,会出现索引越界异常

b:NullPointerException:空指针异常

原因：数组已经不在指向堆内存了。而你还用数组名去访问元素。

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

arr = null;

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

c：一个数组名字只可以指向一个数组实例，因为变量名只能存放一个值，所以只能指向一个数组实例但一个数组实例可以被多个数组名字指向，因为多个变量都存放同一个值（数组实例的地址值）。

## ██数组的常见操作

### 05.09\_Java语言基础(数组的操作1遍历)(掌握)

A:案例演示

数组遍历：就是依次输出数组中的每一个元素。

数组的属性:arr.length数组的长度

数组的最大索引:arr.length - 1;

-------------------------------------------------------------------------------------------------------- public static void print(int[] arr) { 返回值类型void，参数列表为int[] arr.

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

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

}

}

--------------------------------------------------------------------------------------------------

### 05.10\_Java语言基础(数组的操作2获取最值)(掌握)

A:案例演示

数组获取最值(获取数组中的最大值最小值)

--------------------------------------------------------------------------------------------------- public static int getMax(int[] arr) { //

int max = arr[0];

for (int i = 1;i < arr.length ;i++ ) { //从数组的第二个元素开始遍历

if (max < arr[i]) { //如果max记录的值小于的数组中的元素

max = arr[i]; //将大值赋值给max

}

}

return max;

}

------------

角标式做法:

public static int getMax(int[] arr) {

int max = 0;

for (int i = 1;i <= arr.length - 1;i++ ) {

if (arr[i] > arr[max]) {

max = i;

}

}

return arr[max];

}

----------

三元运算符的取最大值做法：

int[] arr = {2,54,46,2345,4,99,90,88,98,67};

//int max = getMax(arr);

//System.out.print(max);

System.out.print(getMax(arr)); //两种输出方式

}

public static int getMax(int[] arr){

int max = arr[0];

for (int i = 0 ;i < arr.length -1;i++ ) {

max = (max< arr[i+1])? arr[i+1]:max; //用三元运算符得出的代码，注意上下的层层对比关系。

}

return max;

}

### 05.11\_Java语言基础(数组的操作3反转)(掌握)

A:案例演示

数组元素反转(就是把元素对调)

---------------------------------------------------------------------------------------------------------------- public static void reverseArray(int[] arr) {

for (int i = 0;i < arr.length / 2 ; i++) { //记得除以2，否就会原反转回去。奇数和偶数都是除以2，中间的不交换。

//arr[0]和arr[arr.length-1-0]交换

//arr[1]和arr[arr.length-1-1]交换

**//arr[2]和arr[arr.lentth-1-2]，上边这么写是有用意的就是发现规律。自己书写的时候一定是内外发生关系，多发生关系才能出规律**。

//...

int temp = arr[i]; //第三方变量永远是最好理解且最好开发的。

arr[i] = arr[arr.length-1-i];

arr[arr.length-1-i] = temp;

}

}

----------

完整的打印程序性应该是

class Demo10\_Array {

public static void main(String[] args) {

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

reverseArray(arr); //进行数组的调用传值。数组已经反转但是没有任何的输出。

print(arr); //数组答打印输出。

}

public static void reverseArray(int[] arr) { //进行数组的反转

for (int i = 0;i < arr.length / 2 ; i++) {

int temp = arr[i];

arr[i] = arr[arr.length-1-i];

arr[arr.length-1-i] = temp;

}

}

public static void print(int[] arr) {

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

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

}

}

}

### 05.12\_Java语言基础(数组的操作4查表法)(掌握)

A:案例演示

数组查表法(根据键盘录入索引,查找对应星期)

import java.util.Scanner;

class Demo11\_Array {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("请输入对应的星期范围在1-7");

int week = sc.nextInt();

System.out.println("星期" + getWeek(week));

}

public static char getWeek(int week) {

char[] arr = {' ','一','二','三','四','五','六','日'};//定义了一张星期表

return arr[week]; //通过索引获取表中的元素，是元素！

}

}

### 05.13\_Java语言基础(数组的操作5基本查找)(掌握)

A:案例演示

数组元素查找(查找指定元素第一次在数组中出现的索引) //仅仅只是第一次？后边的怎么输入？

返回值类型为int型，参数列表为int[] arr,value.

public static int getIndex(int[] arr,int value) {

for (int i = 0;i < arr.length ;i++ ) { //数组的遍历

if (arr[i] == value) { //如果数组中的元素与查找的元素匹配

return i; //这是内部的返回，直接跳出方法

}

}

return -1; //这是本来的返回。若是进不去for，记得写上坏情况的return。

}

## 05.14\_Java语言基础(二维数组概述和格式1的讲解)(了解)

A:二维数组概述

B:二维数组格式1

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

**定义了一个二维数组，有三个二维数组，每个二维数组里边有两个元素。这种格式已经规定了每一个一维数组里的元素的个数是相同的**

C:二维数组格式1的解释

D:注意事项

a:以下格式也可以表示二维数组

1:数据类型 数组名[][] = new 数据类型[m][n];

2:数据类型[] 数组名[] = new 数据类型[m][n];

B:注意下面定义的区别

int x;

int y;

int x,y;

int[] x;

int[] y[];

**int[] x,y[]; x是一维数组,y是二维数组**

E:案例演示

定义二维数组，输出二维数组名称，一维数组名称，一个元素

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

/\*

这是一个二维数组

这个二维数组中有3个一维数组

每个一维数组中有2个元素

**[[I@19bb25a //二维数组的地址值**

**[I@da6bf4 //一维数组的地址值**

**0 //元素值**

\*/

System.out.println(arr); //二维数组

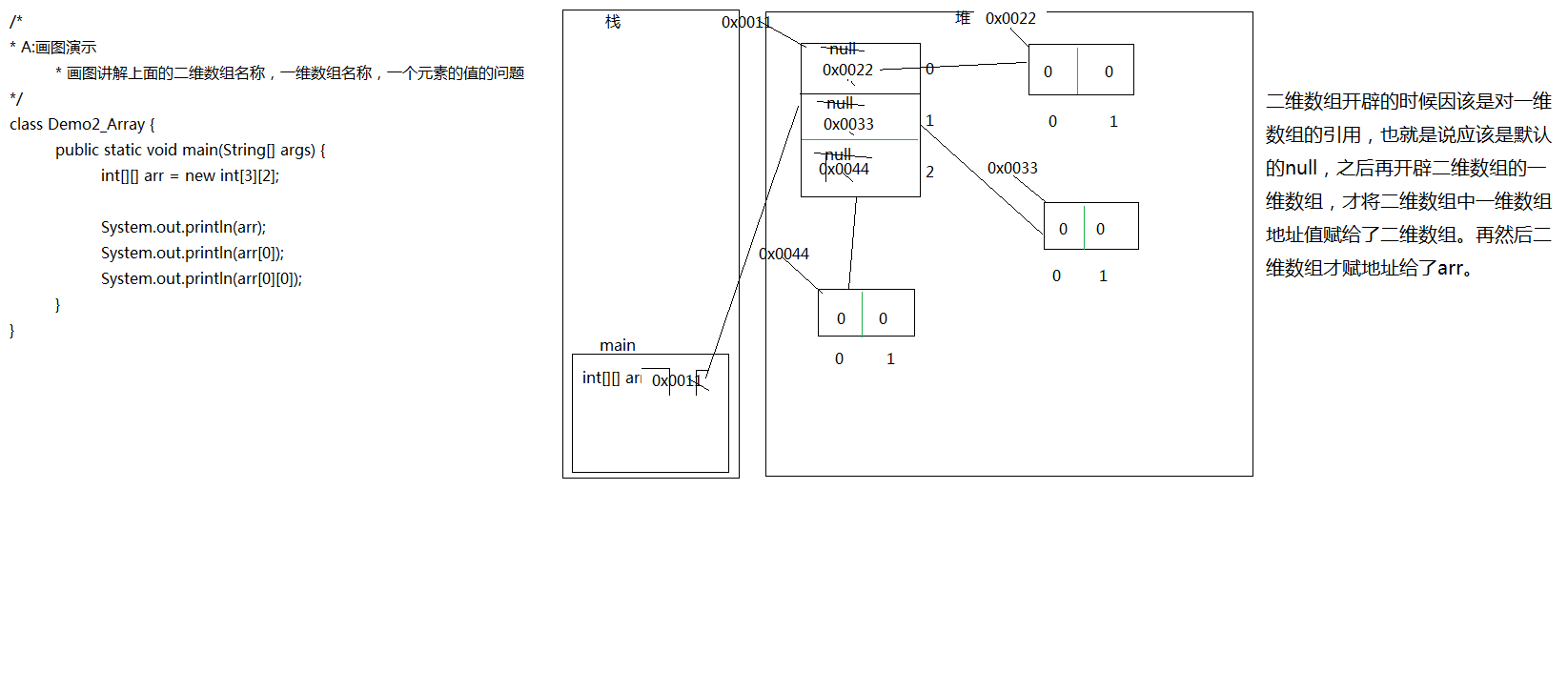
System.out.println(arr[0]); //二维数组中的第一个一维数组

System.out.println(arr[0][0]); //二维数组中的第一个一维数组的第一个元素

## 05.15\_Java语言基础(二维数组格式1的内存图解)(了解)

**A:画图演示**

画图讲解上面的二维数组名称，一维数组名称，一个元素的值的问题



## 05.16\_Java语言基础(二维数组格式2的讲解及其内存图解)(了解)

A:二维数组格式2

int[][] arr = new int[3][];

B:二维数组格式2的解释

C:案例演示

讲解格式，输出数据，并画内存图

int[][] arr = new int[3][]; //这是一个二维数组,这个二维数组中有三个一维数组,三个一维数组都没有被赋值

System.out.println(arr[0]); //全打印都是null

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

System.out.println(arr[2]); //数组的默认都是null。

arr[0] = new int[3]; //第一个一维数组中可以存储三个int值

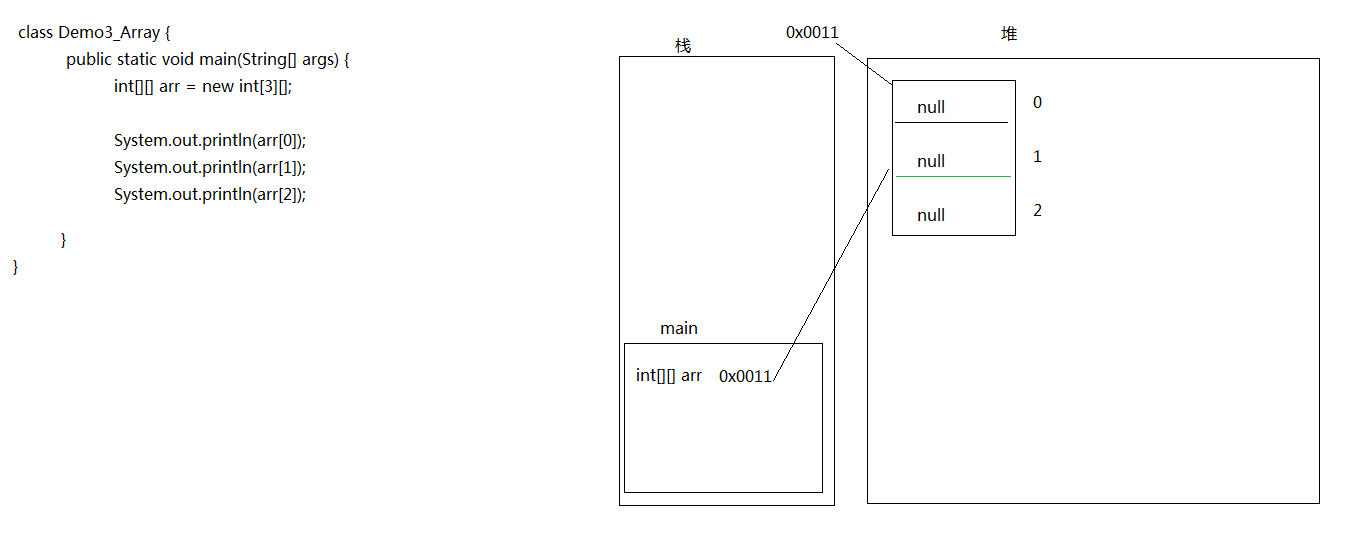
arr[1] = new int[5]; //第二个一维数组中可以存储五个int值

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

System.out.println(arr[0]);  **//现在打印出来的都是地址了！**

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

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



## 05.17\_Java语言基础(二维数组格式3的讲解及其内存图解)(了解)

A:二维数组格式3

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

B:二维数组格式3的解释

C:案例演示

讲解格式，输出数据，并画内存图

class Demo4\_Array {

public static void main(String[] args) {

int[][] arr = {{1,2,3},{4,5},{6,7,8,9}}; //这是一个二维数组,这个二维数组中每个大括号都代表一个一维数组

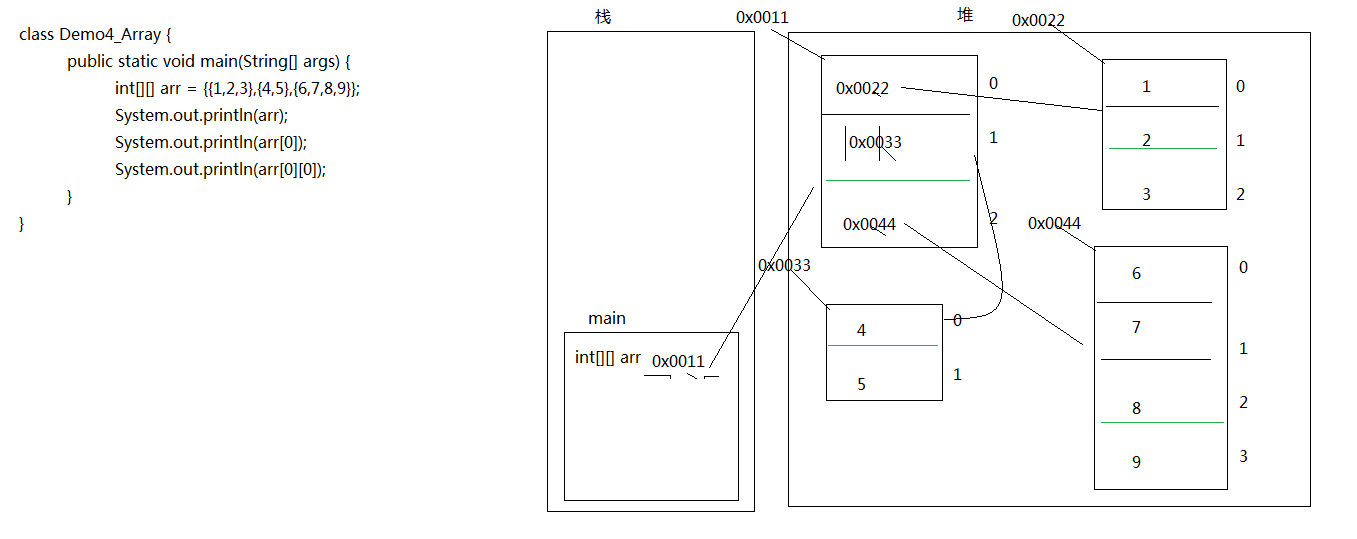
System.out.println(arr); //[[I@19bb25a,二维数组的地址值

System.out.println(arr[0]); //[I@da6bf4,一维数组的地址值

System.out.println(arr[0][0]); //1,一维数组中的元素值

}

}



## 05.18\_Java语言基础(二维数组练习1遍历)(掌握)

A:案例演示

需求：二维数组遍历

外循环控制的是二维数组的长度，其实就是一维数组的个数。

内循环控制的是一维数组的长度。

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

for (int i = 0;i < arr.length ;i++ ) { //获取到每个二维数组中的一维数组

for (int j = 0;j < arr[i].length ;j++ ) { //获取每个一维数组中的元素

System.out.print(arr[i][j] + " ");

}

System.out.println();

}

## 05.19\_Java语言基础(二维数组练习2求和)(掌握)

A:案例演示

需求：公司年销售额求和

某公司按照季度和月份统计的数据如下：单位(万元)

第一季度：22,66,44

第二季度：77,33,88

第三季度：25,45,65

第四季度：11,66,99

int[][] arr = {{22,66,44},{77,33,88},{25,45,65},{11,66,99}};

int sum = 0; //定义变量,记录每次相加的结果

for (int i = 0;i < arr.length ;i++ ) { //获取每一个一维数组

for (int j = 0;j < arr[i].length ;j++ ) { //获取每一个一维数组中的元素

sum = sum + arr[i][j]; //累加

}

}

System.out.println(sum);

## ████05.20\_Java语言基础(思考题Java中的参数传递问题及图解)(掌握面试题)

A:案例演示

看程序写结果，并画内存图解释

public static void main(String[] args) {

int a = 10;

int b = 20;

System.out.println("a:"+a+",b:"+b);

change(a,b);

System.out.println("a:"+a+",b:"+b);//基本数据类型的值传递,不改变原

值,因为调用后就会弹栈,局部变量随之消失。

int[] arr = {1,2,3,4,5};

change(arr); //引用数据类型的值传递,改变原值,因为即使方法弹栈,但是堆

内存数组对象还在,可以通过地址继续访问。但是String是个例外

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

}

public static void change(int a,int b) {

System.out.println("a:"+a+",b:"+b);

a = b;

b = a + b;

System.out.println("a:"+a+",b:"+b);

}

public static void change(int[] arr) {

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

if(arr[x]%2==0) {

arr[x]\*=2;

}

}

}

**Java中到底是传值还是传址**

**1,既是传值,也是传地址,基本数据类型传递的值,引用数据类型传递的地址**

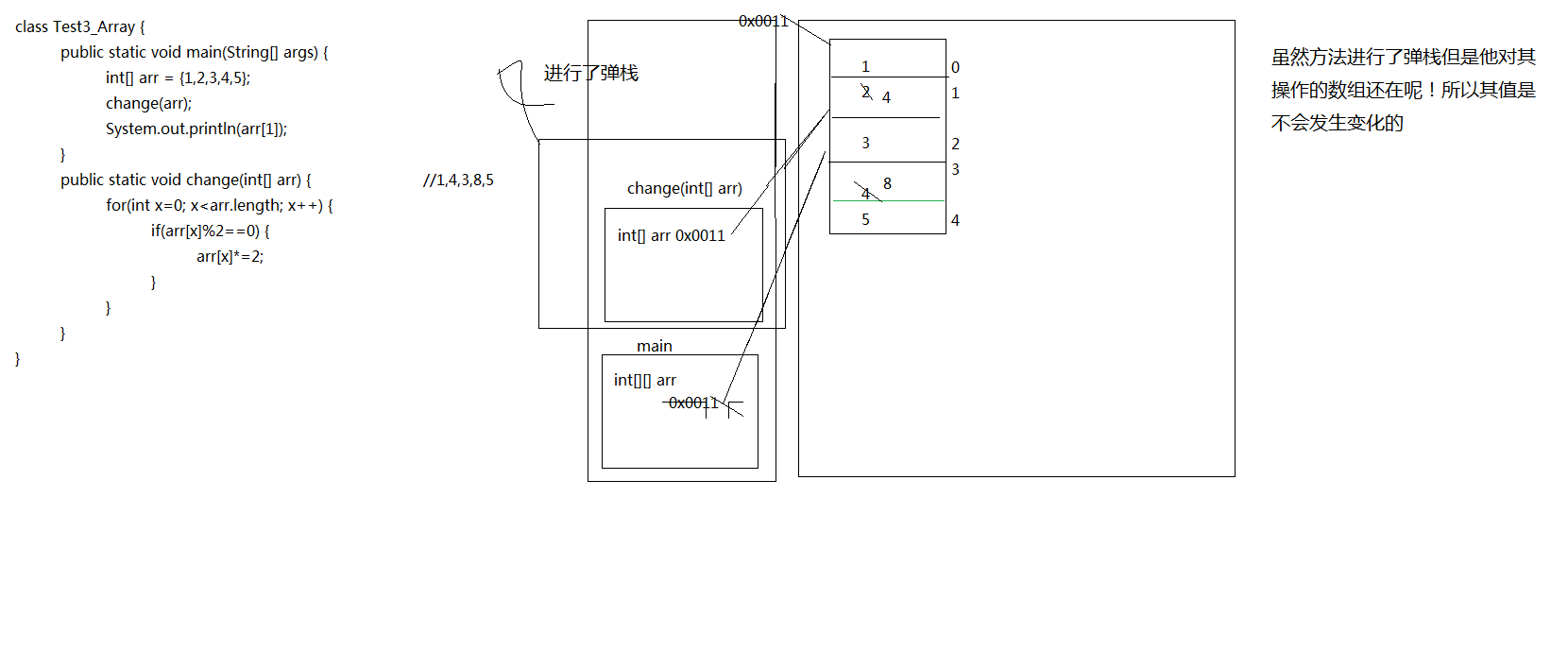
**2,java中只有传值,因为地址值也是值(出去面试都说这种,支持者是高司令(java之父))**

null //代表什么都没有，空格也没有

" " //代表字符串空格

' ' //代表字符空格

\u0000 //Unicode编码，char类型的默认值



05.21\_day05总结

把今天的知识点总结一遍。