

数组使用过程可能出现的问题

```
1 public class ArrayDemo03{
2
3     public static void main(String[] args){
4
5
6         int[] scores = {32,45,45,76};
7
8         System.out.println(scores[5]); // 下标越界
9
10        int[] ages = {32,43,444,32,'a'};
11        System.out.println(ages[4]); // 97 报错
12
13        int[] ages1 = {32,43,444,32L};
14        ages1[2] = 100;
15        ages[2] = 100L;
16        System.out.println(ages1[3]); // 报错 32
17
18    }
19 }
```

- 如果在数组中保存的元素可以自动提升(自动类型转化)为数组自己的类型，那是可以保存的
- 数组下标越界

✧ 数组在计算机中的执行原理

好的各位同学，在前面我们已经学习了数组的基本使用，也理解了数组的基本原理。由于数组是一个容器，变量也是一个容器，在理解他们执行原理的时候，有些同学就容易搞混，现在我把他们放在一起带着大家回顾一下他们的会执行原理，顺便带着大家详细理解一下Java程序的执行的内存原理。

数组的执行原理，Java程序的执行原理

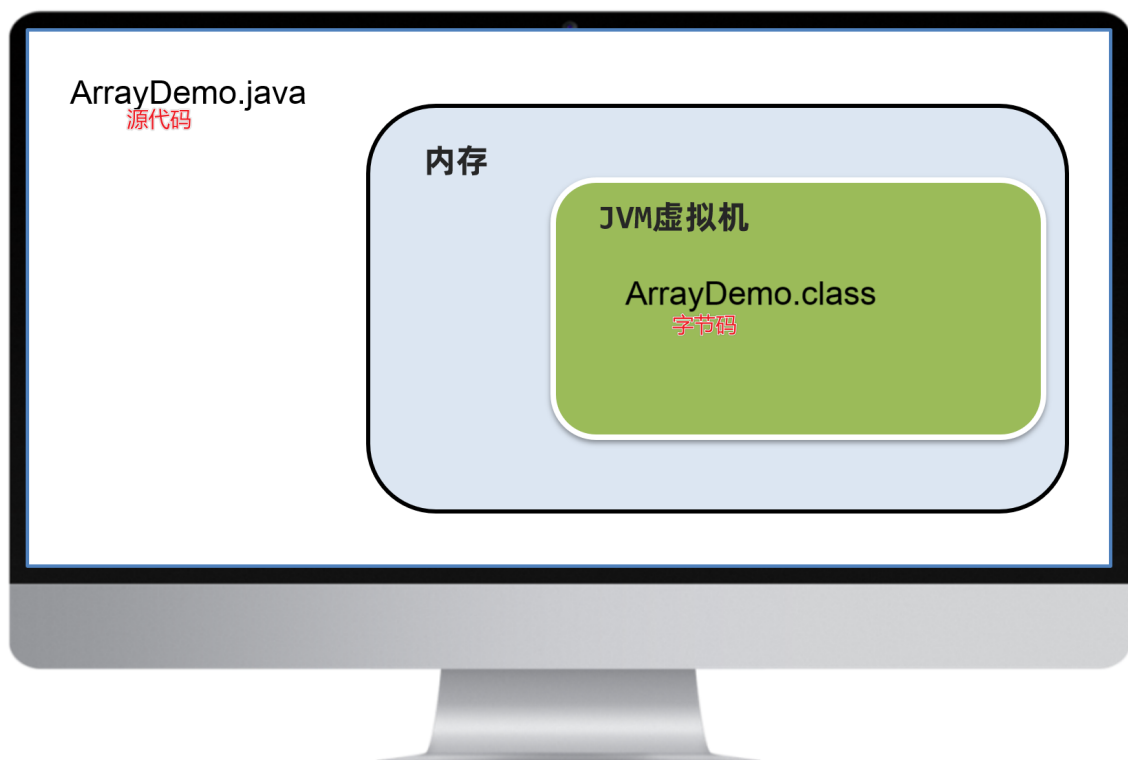
我们以下面的代码，来讲解变量、数组的执原理。

```

1 public class ArrayDemo1 {
2     public static void main(String[] args) {
3         int a = 10;
4         System.out.println(a); // 10
5
6         int[] arr = new int[]{11, 22, 33}; // 静态初始化
7         System.out.println(arr); // 地址
8
9         System.out.println(arr[1]); // 22
10
11        arr[0] = 44;
12        arr[1] = 55;
13        arr[2] = 66;
14
15        System.out.println(arr[0]); // 44
16        System.out.println(arr[1]); // 55
17        System.out.println(arr[2]); // 66
18    }
19 }

```

前面我们给大家讲过，程序是在内存中执行的。实际上Java程序是把编译后的字节码加载到Java虚拟机中执行的。



Java为了便于虚拟机执行Java程序，将虚拟机的内存划分为 方法区、栈、堆、本地方法栈、寄存器 这5块区域。同学们需要重点关注的是 方法区、栈、堆。

下面把每一个块内存区域作用介绍一下，我们大致只需要知道每一部分存储什么内容就行。

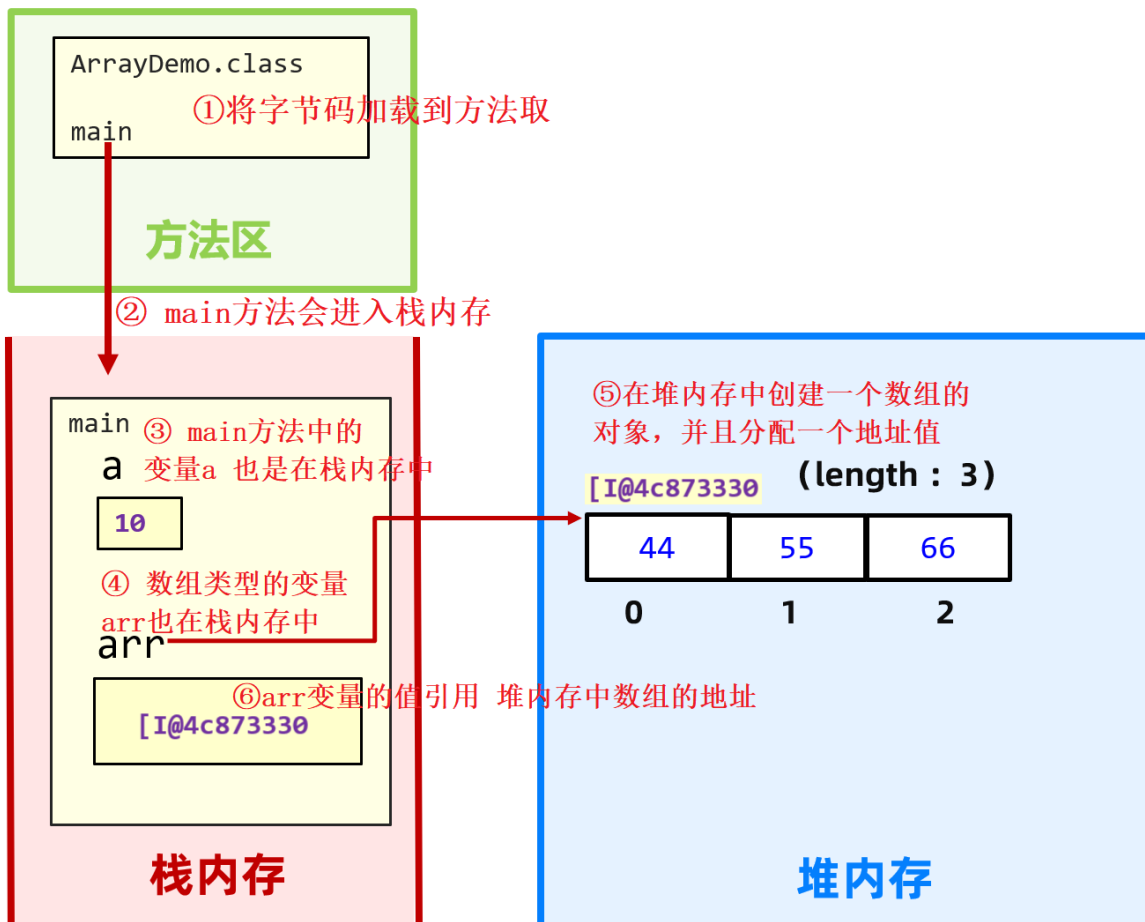
- 方法区：字节码文件先加载到这里

- 1 Random rand = new Random();
- 2 Scanner sc = new Scanner(System.in);

○

- 栈：方法运行时所进入的内存区域，由于变量在方法中，所以变量也在这一块区域中
- 堆：存储new出来的东西，并分配地址。由于数组是new 出来的，所以数组也在这块区域。

下面是上面案例执行的内存原理如下图所示，按照① ② ③ ④ ⑤ ⑥ 的标记的顺序来看



总结一下 `int a = 10` 与 `int[] arr = new int[]{11,22,33}`的区别

- `a`是一个变量，在栈内存中，`a`变量中存储的数据就是10这个值。
- `arr`也是一个变量，在栈中，存储的是数组对象在堆内存中的地址值

```
1 // 这里的int a是一个基本类型变量，存储的是一个数值
2 int a = 10 ;
3 //这里的int[] arr是一个引用类型的变量，存储的是一个地址值
4 int[] arr = new int[]{44,55,66};
```

多个变量指向同一个数组的问题

各位同学，我们了解了数组在内存中的执行原理。我们知道数组类型的变量，指向的是堆内存中数组对象的地址。但是在实际开发中可能存在一种特殊情况，就是多个变量指向同一个数组对象的形式。

讲解这个知识点的目的，是让同学们注意多个变量指向同一个数组对象存在什么问题？

我们先看一段代码

```
1 public class ArrayDemo2 {
2     public static void main(String[] args) {
3         // 目标：认识多个变量指向同一个数组对象的形式，并掌握其注意事项。
4         int[] arr1 = new int[]{11, 22, 33};
5
6         // 把int类型的数组变量arr1赋值给int类型的数组变量arr2
7         int[] arr2 = arr1;
8         int[] arr3 = new int[]{11, 22, 33};
9
10        System.out.println(arr1); // 地址
11        System.out.println(arr2); // 地址 1 == 2 都不一样
12        System.out.println(arr3); // 地址 3 不一样
13
14        arr2[1] = 99;
15        System.out.println(arr1[1]);
16
17        arr2 = null; // 拿到的数组变量中存储的值是null
18        System.out.println(arr2);
19
20        //System.out.println(arr2[0]);
21        //System.out.println(arr2.length);
22    }
23 }
```

我们先看一段代码

```
1 public class ArrayDemo2 {
2     public static void main(String[] args) {
3         // 目标: 认识多个变量指向同一个数组对象的形式, 并掌握其注意事项。
4         int[] arr1 = {11, 22, 33};
5
6         // 把int类型的数组变量arr1赋值给int类型的数组变量arr2
7         int[] arr2 = arr1;
8         int[] arr3 = {11, 22, 33};
9
10        System.out.println(arr1); // 地址
11        System.out.println(arr2); // 地址 1 == 2 都不一样
12        System.out.println(arr3); // 地址 3 不一样
13        int[] arr3;
14        int[] arr2 = {11, 22, 33};
15        System.out.println(arr1[1]);
16        int[] arr1 = {11, 22, 33};
17        arr2 = null; // 拿到的数组变量中存储的值是null
18        System.out.println(arr2);
19
20        //System.out.println(arr2[0]);
```

堆

栈

我们重点关注这一段代码

这里arr1记录的是数组的地址值

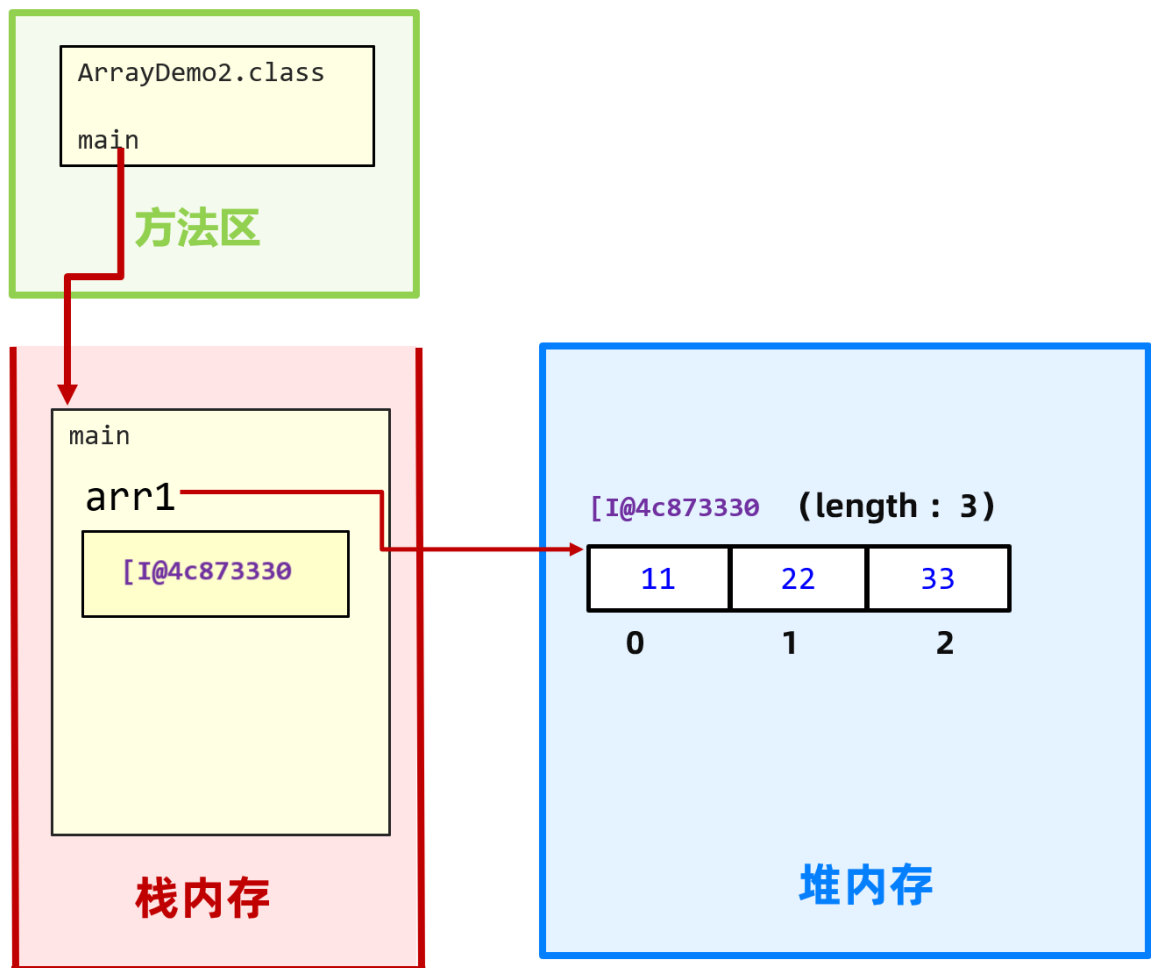
```
int[] arr1 = {11, 22, 33};
```

// 把int类型的数组变量arr1赋值给int类型的数组变量arr2

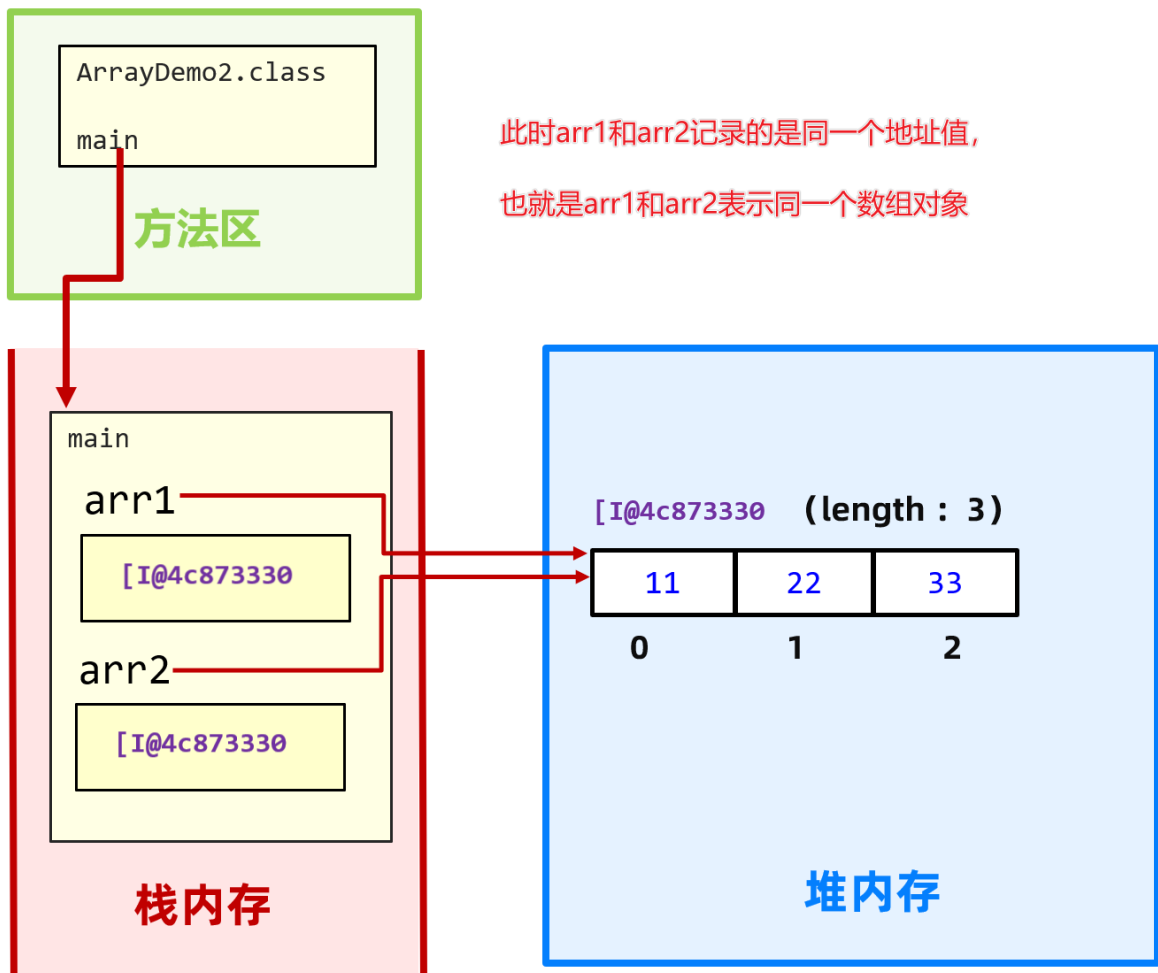
```
int[] arr2 = arr1; 把arr1记录的地址, 再赋值给arr2
```

此时: arr1和arr2就是同一个地址值

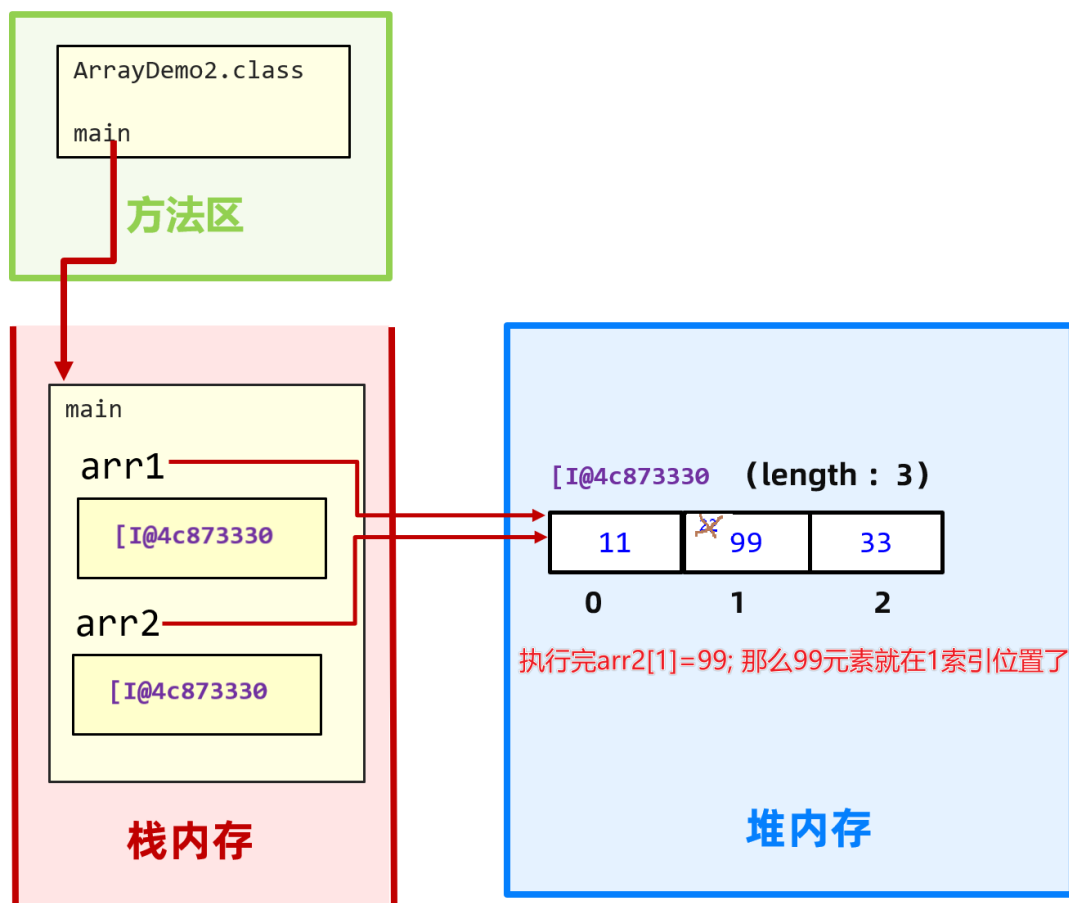
刚执行完 `int[] arr1 = {11,22,33};` 时, 内存原理如下



当执行完 `int[] arr2 = arr1;` 后，内存原理如下



当执行到 `arr2[1]=99;` 时，内存原理如下



总结一下：

- 两个变量指向同一个数组时，两个变量记录的是同一个地址值。
- 当一个变量修改数组中的元素时，另一个变量去访问数组中的元素，元素已经被修改过了。

到这里有关数组的基本操作，和内存原理我们就全部学习完了。



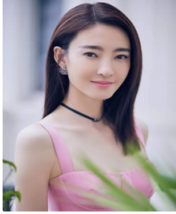


✧ 数组专项练习

接下来我们做一些专项练习题，把数组的常见操作练习一下。在学习这个案例时，重点掌握数组求最值的思路，代码只是用来表达你的思路的。

数组求最值

1 需求：定义一个int类型数组，求数组中元素的最大值，并打印最大值

我们先看一下选美比赛，是怎么选出颜值最高的人的。然后再以此思路，来写代码找出数组中元素的最大值。

					
颜值：15	颜值：9000	颜值：10000	颜值：20000	颜值：9500	颜值：-5

- ① 首先，准备一个擂台，凤姐带着15的颜值先上台
- ② 接着，后面的每一个人，依次上台和擂台上的主角进行比较
- ③ 然后，每次比较，将颜值胜出的人留在擂台上。
- ④ 等每一个元素都比较完了，留在擂台上的就是颜值最高的



- 1 数组求最大值思路：
- 2 1) 先找出数组中0索引的元素，假设为最大值，用max表示【擂主】
- 3 2) 遍历后面的每一个元素和max比较，把较大的元素值重新赋值给max(擂主换人)
- 4 3) 最后max就是所有元素的最大值(最后站在台上的擂主)

总结一下：

通过这个案例，我们主要掌握求最值的思路，以后不管遇到求最大值还是最小值，编程思路都是一样的，不同的可能是数据不同。

课堂练习：

```
1 // 求最小值
```

使用增强for循环遍历数组

JDK1.5及其之后的版本中提供了增强for循环语句，实现了Iterable接口的类都可以使用增强for循环进行元素的迭代。增强for循环的语法规则如下：

```
1 for (元素类型 变量名 : 要迭代的对象) {
2     System.out.println(变量名);
3 }
4
5 int[] arr = {15, 9000, 10000, 20000, 9500, -5};
```



```
6   for (int e : arr) {  
7       System.out.println(e);  
8   }  
9  
10  for (int i = 0; i < arr.length; i++) {  
11      System.out.println(arr[i]);  
12  }  
13  
14  for (;;) {  
15  
16  }
```

语法解析：

- 元素类型是指数组或集合中的元素的类型。
- 变量名在循环时用来保存每个元素的值。
- 冒号后面是要遍历的数组或集合的名称。