# Day04 Java基础

## 数组的定义

### 数组的概念：

数组是用来存储多个同一类型的数据的长度不可变的容器， 数组会对其内部存放的数据从0开始进行编号。我们成这个编号为数组的下标。

### 数组的定义格式：

数据类型[] 数组名 = new 数据类型[数组的大小/长度]; --- 动态初始化

int[] arr = new int[5]; -- 表示定义了一个能存储5个整型元素的数组

也可以定义为int[] arr; arr=new int[5];

arr[2] = 5; --- 表示向arr数组中下标为2的位置放入元素5

数据类型[] 数组名 = new 数据类型[]{元素1, 元素2, ... , 元素n}; --- 静态初始化

int[] arr = new int[]{2,5,1,6,4,8};---同时规定了数组的长度就是6

也可以定义为int[] arr；arr = new int[]{2,5,1,6,4,8};

arr[3] = 7;--表示向arr数组中下标为三的位置放入元素7

数据类型[] 数组名 = {元素1, 元素2, ... , 元素n};

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

注意不能定义为int[] arr；arr== {2,1,6,4,3,1,8};的格式

注意当[] 定义在数据类型后边，作用范围是数据类型后边的所有变量，而定 义在变量后边，则范围只包括当前变量，即 int[] a,b;表示a和b都是数 组，而 int a[],b;则表示a是一个数组，b是一个变量。

## 数组的应用

### 获取数组中的元素：

数组名[下标]

例如有一个数组int[] arr = new int[]{2,5,1,6,4,8};我们要获取下标 为3的元素，则我们可以定义一个和数组类型一致的变量来接一下

int i = arr[3]

### 获取数组的长度

.length

例如有一个数组int[] arr = new int[]{2,5,1,6,4,8};我们要获取数组 的程度，则我们可以定义一个整型的变量来接一下

int i = arr.length;

### 循环遍历数组：

for循环和增强for循环

For循环：

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

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

}

增强for循环：增强for循环就是使用一个变量来接数组中的一个元素， 然后循环遍历数组中的每一个位置。

for(int i : arr){

System.out.println(i);

}

### 获取数组中的最大值：

我们先定义一个变量，来记录数组中的第一个位置 的元素，并假设它为最大，然后，我们循环遍历这个数组，并和这个变量进行 比较，如果比这个变量大，就把当前较大的值设为最大值。

int max = arr[0];

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

if(max < arr[i])

max = arr[i];

}

或者我们也可以利用下标来判断，我们先定义一个变量来记录数组第一个位置 的下标，并假设这个位置的元素最大，然后，我们循环遍历这个数组，并将其 他下标所存储的元素和这个变量所存储的元素进行比较，如果比这个变量所存 储的元素大，就把当前较大的值的下标设为最大值的下标。

int max = 0;

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

if(arr[max] < arr[i])

max = i;

}

### 数组的排序：

冒泡排序：我们将数组中相邻的两个元素，从左到右依次比较，并按照排序的 规则来确定要不要交换两个元素的位置，这样从未到位比较一轮，就能选出一 个最值，然后继续进行下一轮，通常我们需要比较的轮数等于数组的长度减一

代码为从小到大排序：

for(int i = 1; i < arr.length; i++){ // 控制轮数

for(int j = 1; j <= arr.length - i; j++){ // 控制次数和下标

if(arr[j - 1] > arr[j]){

int temp = arr[j - 1];

arr[j - 1] = arr[j];

arr[j] = temp;

}

}

}

快速排序：我们选中数组中的第一位元素，依次和后边的元素进行比较，并按 照排序规则确定要不要交换两个元素的位置，从头到尾比较一次，选出一个最 值，然后选中第二个比较第二轮，依次类推，比较的轮数为数组的长度减一

代码为从小到大排序：

for(int i = 1; i < arr.length; i++){ // 控制轮数和选择的下标

for(int j = 1 + i; j <= arr.length ; j++){ // 控制次数和比 较的下标

if(arr[i - 1] > arr[j - 1]){

int temp = arr[i - 1];

arr[i - 1] = arr[j - 1];

arr[j - 1] = temp;

}

}

}

### 数组的反转：

我们定义两个变量，分别代表数组的起始下标和末位下标， 然后我们将这两个位置的元素进行交换，然后起始下标后移一位，末位下标向 前移一位，再次进行交换，直到起始下标和末位下标指向同一个元素，或者起 始下标在末位下标后面，则完成反转

代码如下：

for(int start = 0,end = arr.length - 1;start<end;start++,end--){

int temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

}

### 数组元素的查找：

折半查找（只能用于有序数组）

因为数组中的元素是有序的假如是升序，我们先找到数组中间位置的元素，先 判断我们要找的数字和中间位置数字的关系，如果比中间位置元素小，则我们 去中间位置和起始位置这一段，如果比中间位置元素大，则取中间位置和末尾 这一段，然后再取这一段的中间位置进行比较。

代码实现：

int[] arr = {1,3,5,9,14,25,53,72};

int number = 58;

int min = 0;

int max = arr.length - 1;

int mid = (min + max) / 2;

while(arr[mid] != number){

if(arr[mid] > number){

max = mid - 1;

} else {

min = mid + 1;

}

if(min > max){

mid = -1;

break;

}

mid = (min + max) / 2;

}

### 数组的复制：

将一个数组中已有的数据复制到另一个数组中

复制格式：

System.arraycopy(要复制的数组,要复制的起始下标,要存放的数组,要存 放的起始下标,要复制的元素的个数);

### 数组的扩容：

我们说过数组一旦定义，长度不可变，所以数组的扩容和数 组的复制类型，都是指向了一个新的数组：

扩容格式：

数组 = Arrays.copyOf(要扩容的数组, 扩容之后的长度);

## 数组的内存

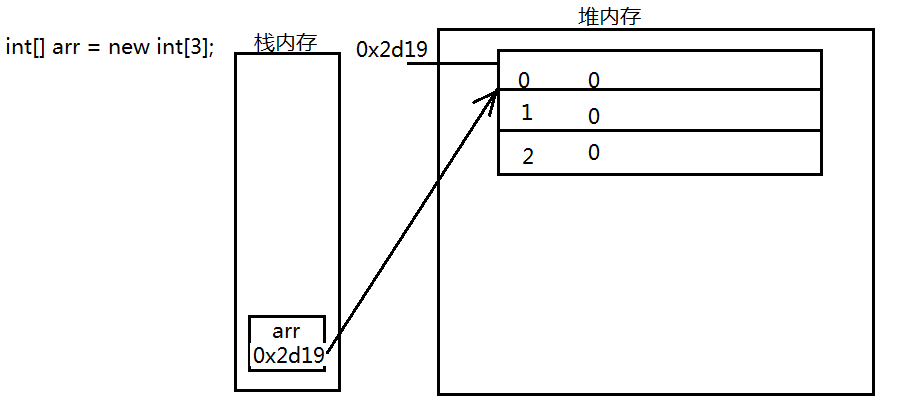
### 内存的介绍：

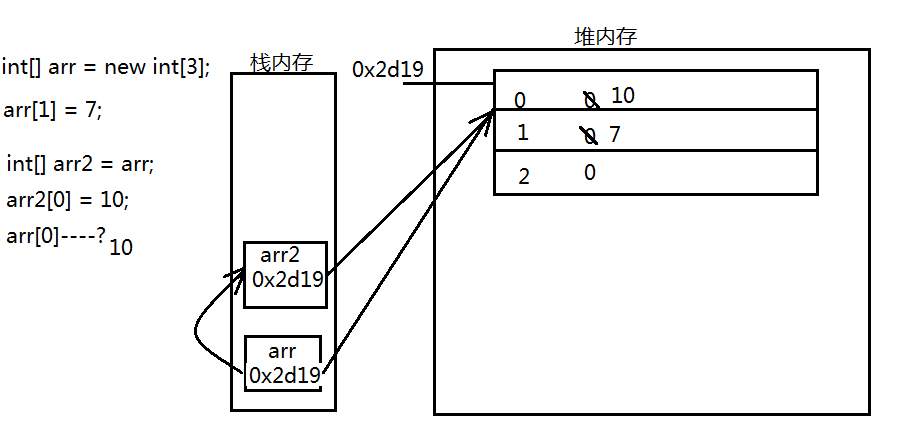
Java将内存分为了5块：栈内存、堆内存、方法区、本地方法栈、寄存器

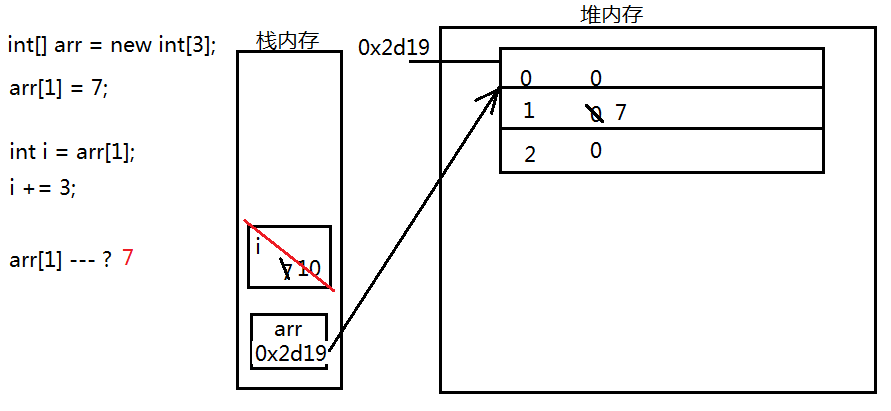
栈内存：存储变量的。栈内存不会自动给变量赋值。而且变量用完之后会从栈 内存中立即移除以释放内存空间

堆内存：存储的是数组或者对象。碰到new关键字的时候会到堆内存中开辟 空间。堆内存会对存入其中的元素自动的赋予一个默认值 ---byte/short/int-0, long-0L, float-0.0f, double-0.0, char- ‘\u0000’, boolean-false, 除了这八种基本类型以外的所有的类型的默认 值都是null。 堆内存中的数据用完之后会在不定的某个时刻被回收。

### 关于数组的内存：







注意：如果数组超过了最大的下标，则会出现数组下标越界异常。