复习：

1. 流程控制

顺序结构

分支结构：

判断结构：if, if-else, if-else if

选择结构：switch-case --- 选项的类型可以是byte/ short/ char/ int, 从JDK1.7开始，允许使用String。case之后有无break，那么他的顺序将会影响结果

循环结构：while, do-while, for

do-while的循环体无论如何都会至少执行一次。

在次数固定或者是变化规律的前提下，建议使用for循环；在次数不固定或者变化不规律的情况下，建议使用while循环

循环嵌套

break和continue

break：用于选择和循环结构，表示终止当前的一层结构。

continue：用于循环结构，表示跳出当前的一次循环继续下次循环（也是只能跳一层）

## 数组

用于存储多个同一类型的数据的容器 --- 大小是固定的

下标是从0开始，所以最大下标是长度-1

### 定义格式

数据类型[] 数组名 = new 数据类型[长度];

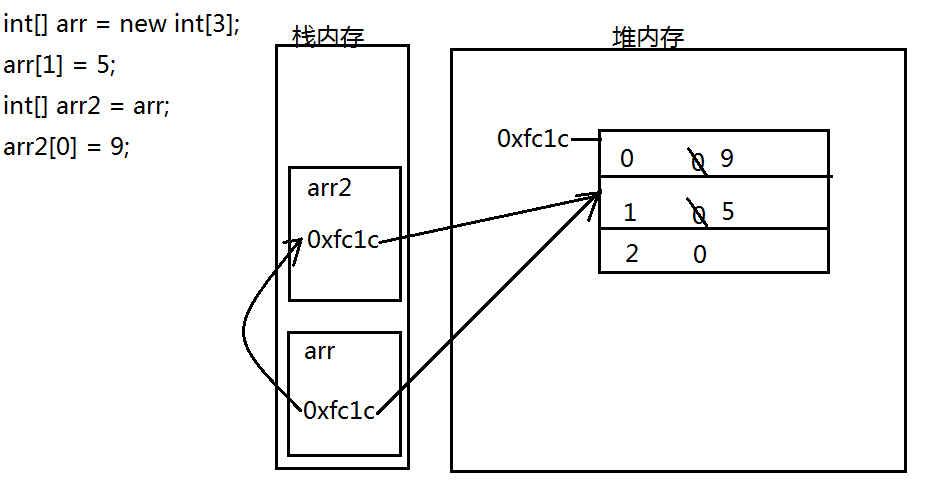
数据类型[] 数组名 = new 数据类型[]{元素1, 元素2, ...};--- 这种方式在定义的时候给定了具体的元素同时也规定了数组的长度

int[] arr = new int[]{5,3,7,1,8,9,2}; 表示定义了一个元素类型为整型的数组，数组中的元素是5,3,7,1,8,9,2，意味着数组的长度是7

数组类型[] 数组名 = {元素1, 元素2, ...}; --- 不允许分开定义

### 数组的内存

数组存储在堆内存中，并且在堆内存中会自动赋予默认值：byte/short/int - 0, long - 0L, float - 0.0f, double - 0.0, char - ‘\u0000’, boolean - false，其他所有类型的默认值都是null



注意：数组在赋值的时候赋的实际上是地址

### 数组的应用

1. 操作指定位置上的元素：数组名[下标]

2. 获取数组的长度：数组名.length

3. 遍历数组

// 方式一：下标是从0->length - 1

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

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

}

// 方式二：直接获取每一个位置上的元素

// 增强for循环

// 只能遍历数组但是不能改变数组中的元素

for(int i : arr){

System.out.println(i);

}

// 方式三：将数组中的元素一个个的拿出来拼接成字符串

String str = Arrays.toString(arr);

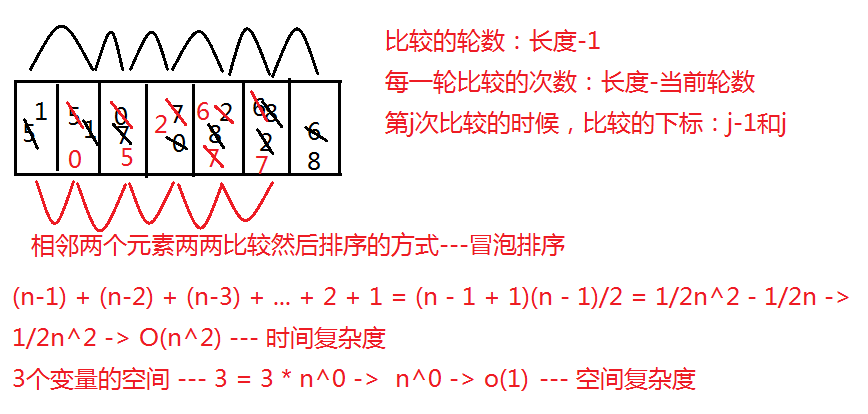
System.out.println(str);

4. 获取数组中最值（最大值/最小值）

方式一：定义变量来记录数组中的最大值，然后遍历数组，让数组中的元素依次与最大值进行比较；如果大于最大值，则将用这个元素覆盖原来的最大值

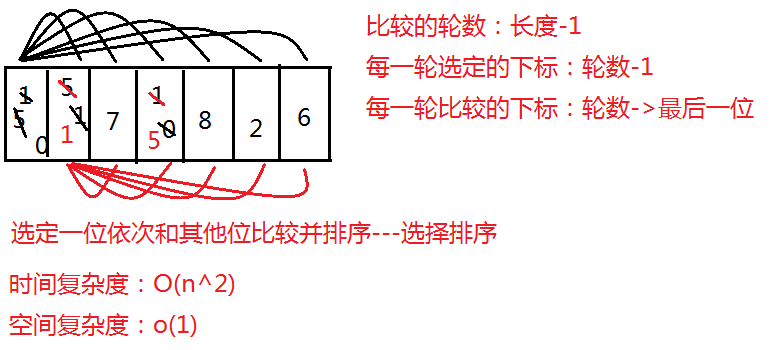
方式二：定义变量来记录最大值的下标

5. 数组的排序



时间复杂度：在程序中找一段必然会执行的代码，将这段代码的执行时间认为是单位1，执行这个单位1需要的次数就是时间复杂度 - 时间复杂度不考虑系数，一般来说是找最高阶 -> O(nx), O((logn)x),O(nx(logn)y)

空间复杂度：这段程序执行所需要额外耗费的空间就是空间复杂度



扩展：冒泡排序和选择排序都是稳定的排序算法 --- 排序算法的稳定与否的依据是相等的元素在排序的时候是否需要交换

6. 反转数组：首尾互换 --- 时间复杂度O(n)，空间复杂度o(1)

7. 数组元素的查找：

数组元素无序的前提下，获取一个元素的位置只能通过遍历的方式一一比较。

如果数组元素有序，使用二分查找 --- 空间复杂度o(1)，时间复杂度O(logn)

2x = n -> x = log2n -> logn --- 默认以2为底数

8. 数组的复制

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

## 二维数组

存储的元素是一维数组 --- 存储数组的数组

### 定义格式

数据类型[][] 数组名 = new 数据类型[包含的一维数组的个数][每一个一维数组的长度];

int[][] arr = new int[3][5]; 表示定义了一个能存储3个整型一维数组的二维数组，每一个一维数组能5个整型元素

数据类型[][] 数组名 = new 数据类型[包含的一维数组的个数][];

int[][] arr = new int[5][]; 定义了一个能存储5个整型一维数组的二维数组 --- 必须先保证这一位上的一维数组先给定大小，然后再给值

arr[0] = new int[3];

arr[1] = new int[7];

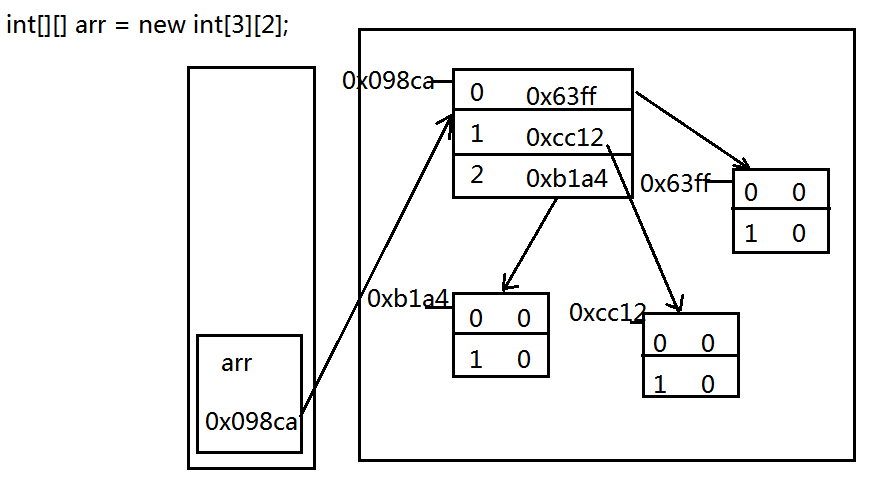
arr[2] = new int[2];

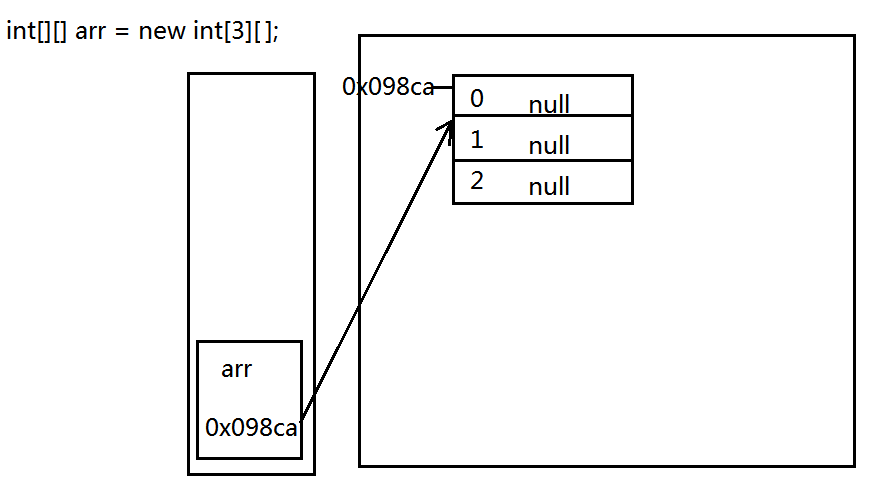
数据类型[][] 数组名 = {{数组1}, {数组2}, {数组3},...};

int[][] arr = {{2,4,1}, {4,7,2,9}, {3}, {5,0,6,7,4,3}}; --- 二维数组的大小为4

注意：[]如果在变量名之前那么紧跟数据类型，也就意味着后面定义的变量实际上都是数组；但如果[]在变量名之后，那么[]只管跟着的这个变量不会去影响其他的变量。

### 内存存储





练习：杨辉三角

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

特点：每一行的开始和结束都是1；其余位置的元素是计算：

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

输入一个数字n表示行数，输出对应的前n行