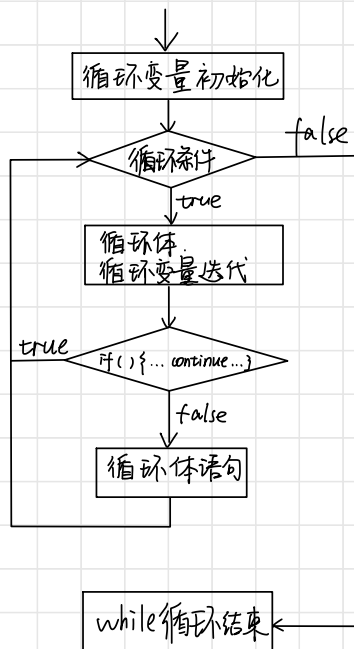


跳转控制语句 - continue

- 1) continue 用于结束本次循环, 继续执行下一次循环
- 2) continue 语句也可使用标签达到跳转指定循环的效果

```
{ ...  
    continue;  
    ...  
}
```

以 while 使用 continue 为例, 画出示意图



跳转控制语句 - return

return 语句多用在定义方法中, 表示退出该方法, 若 return 写在 main 方法中, 则退出程序。

2021.4.27.

Pr4 - P Java - 数组. (第六章)

数组介绍

数组可以存放多个同一类型的数据, 数组是引用数据类型, 即数组就是一组数据。

数组的定义

数据类型 数组名[] = new 数据类型[大小] (数据类型[] 数组名 = new 数据类型[大小])

int a[] = new int[5] // 创建一个数组, 名字 a, 存放 5 个 int

数组的动态初始化

方法 1:

```
2 import java.util.Scanner;  
3  
4 public class Array01{  
5     // 编写一个 main 方法  
6     public static void main(String[] args){  
7         /* 演示: 数据类型 数组名[] = new 数据类型[大小]  
8         循环输入五个成绩, 保存到 double 数组, 并输出  
9         */  
10  
11         // 步骤 1, 创建一个 double 数组, 大小 5  
12         double scores[] = new double[5];  
13         // 步骤 2, 循环输入  
14         Scanner myScanner = new Scanner(System.in);  
15         for(int i = 0; i < scores.length; i++){  
16             System.out.println("请输入第" + (i+1) + "个元素的值: ");  
17             scores[i] = myScanner.nextDouble();  
18         }  
19  
20         // 步骤 3, 循环打印  
21         System.out.println("====数组的元素值的情况如下====");  
22         for(int i = 0; i < scores.length; i++){  
23             System.out.println("第" + (i+1) + "个元素的值=" + scores[i]);  
24         }  
25     }  
26 }
```

```
[^C(base) MacBook-Pro:0427 songfei$ java Array01  
请输入第 1 个元素的值:  
86.4  
请输入第 2 个元素的值:  
98.3  
请输入第 3 个元素的值:  
76.9  
请输入第 4 个元素的值:  
83.1  
请输入第 5 个元素的值:  
78.2  
====数组的元素值的情况如下====  
第 1 个元素的值=86.4  
第 2 个元素的值=98.3  
第 3 个元素的值=76.9  
第 4 个元素的值=83.1  
第 5 个元素的值=78.2
```

方法2:

先声明, 再分配(new)

```
// 步骤1. 创建一个double数组, 大小5
double scores[]; // 声明数组, 这时scores是null
scores = new double[5]; // 分配内存空间, 可以存放数据
// 如果不定义new, 会报空指针异常!
// 步骤2. 循环输入
Scanner myScanner = new Scanner(System.in);
for(int i = 0; i < scores.length; i++){
```

静态初始化

语法: 数据类型 数组名[] = {元素值1, 元素值2, ..., 元素值n};

例: `int a[] = {1, 2, 5, 6, 8, 91, 90, 34}`

数组使用注意事项和细节

- 1> 数组是多个相同类型数据的组合, 实现对数据的统一管理
- 2> 数组中的元素可以是任何数据类型, 包括基本数据类型和引用数据类型, 但是不能混用
- 3> 数组创建后, 如果未赋值, 有默认值; 其中 `int` 为 0, `short` 为 0, `byte` 为 0, `long` 为 0, `float` 为 0.0, `double` 为 0.0, `char` 为 `\u0000`, `boolean` 为 `false`, `String` 为 `null`.
- 4> 使用数组的步骤: 声明数组并开辟空间; 给数组各个元素赋值; 使用数组
- 5> 数组下标从 0 开始
- 6> 数组下标必须在范围内使用, 否则异常 数组下标越界异常 [编译√, 但运行×]
- 7> 数组属于引用类型, 数组型数据是对象 (Object).

数组赋值机制: ★★

```
// 基本数据类型类型赋值, 赋值方式为值拷贝
// n2值的变化不会影响n1
int n1 = 10;
int n2 = n1;

n2 = 80;
System.out.println("n1 = " + n1);
System.out.println("n2 = " + n2);

// 数组在默认情况下为引用传递, 赋的值是地址, 赋值方式为引用传达
// arr2中存储的arr1值的地址, 因此arr2的改变会影响arr1的值
int arr1[] = {1, 2, 3};
int arr2[] = arr1;

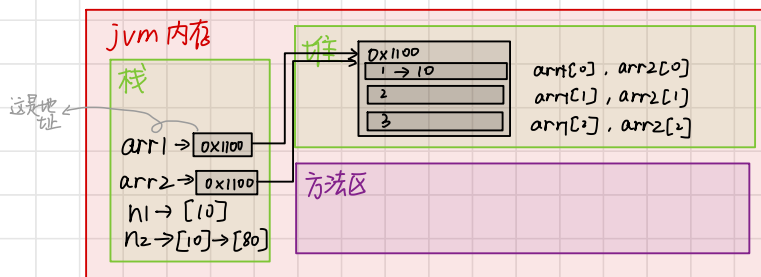
arr2[0] = 4;
System.out.print("arr1的值为: ");
for(int i = 0; i < arr1.length; i++){
    System.out.print(arr1[i] + " ");
}
System.out.print("\narr2的值为: ");
for(int i = 0; i < arr2.length; i++){
    System.out.print(arr2[i] + " ");
}
System.out.println();
```

```
(base) MacBook-Pro:0427 songfei$ javac ArrayAssign.java
(base) MacBook-Pro:0427 songfei$ java ArrayAssign
```

```
n1 = 10
n2 = 80
arr1的值为: 4 2 3
arr2的值为: 4 2 3
```

重点理解: jvm的内存分配方式.

值传递和引用传递的区别



! jvm中还还有本地方法栈和程序计数器

! 在内存空间中只要分配了数据空间, 一定会对应一个地址.

∴ 引用传递也称为地址拷贝.

排序的介绍

将多个数据, 依指定的顺序进行排列

分类:

- 1) 内部排序: 指将需要处理的所有数据都加载到内部存储器中进行排序. 包括交换式排序法、选择式排序法和插入式排序法)
- 2) 外部排序法: 数据量过大, 无法全部加载到内存中, 需要借助外部存储进行排序. 包括合并排序法和直接合并排序法.

冒泡排序法

分析冒泡排序

数组 [24, 69, 80, 57, 13]

第1轮排序: 目标把最大数放在最后

第1次比较 [24, 69, 80, 57, 13]

第2次比较 [24, 69, 80, 57, 13]

第3次比较 [24, 69, 57, 80, 13]

第4次比较 [24, 69, 57, 13, 80]

第2轮排序: 目标把第二大数放在倒数第二位置

第1次比较 [24, 69, 57, 13, 80]

第2次比较 [24, 57, 69, 13, 80]

第3次比较 [24, 57, 13, 69, 80]

第3轮排序: 目标把第三大数放在倒数第三位置

第1次比较 [24, 57, 13, 69, 80]

第2次比较 [24, 13, 57, 69, 80]

第4轮排序: 目标把第四大数放在倒数第四位置

第1次比较 [13, 24, 57, 69, 80]

总结冒泡排序特点

1. 我们一共有5个元素
 2. 一共进行了4轮排序, 可以看成是外层循环
 3. 每1轮排序可以确定一个数的位置, 比如第1轮排序确定最大数, 第2轮排序, 确定第2大的数位置, 依次类推
 4. 当进行比较时, 如果前面的数大于后面的数, 就交换
 5. 每轮比较在减少 4 -> 3 -> 2 -> 1
- 分析思路 -> 代码..