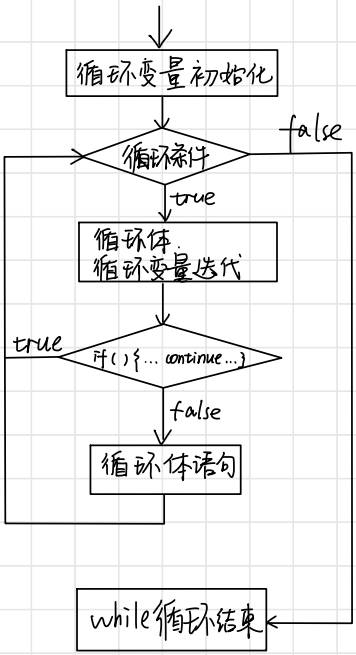
跳转控制语句 - continue

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.

P154 - 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：

```
import java.util.Scanner;

public class Array01{
    // 编写一个main方法
    public static void main(String[] args){
        /* 演示 数据类型 数组名[] = new 数据类型[大小]
            循环输入五个成绩，保存到double数组，并输出
        */

        // 步骤1. 创建一个double数组，大小5
        double scores[] = new double[5];
        // 步骤2. 循环输入
        Scanner myScanner = new Scanner(System.in);
        for(int i = 0; i < scores.length; i++){
            System.out.println("请输入第" + (i+1) + "个元素的值: ");
            scores[i] = myScanner.nextDouble();
        }

        // 步骤3. 循环打印
        System.out.println("====数组的元素值的情况如下====");
        for(int i = 0; i < scores.length; i++){
            System.out.println("第" + (i+1) + "个元素的值=" + scores[i]);
        }
    }
}
```

```
^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，会报错空指针异常
// 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).

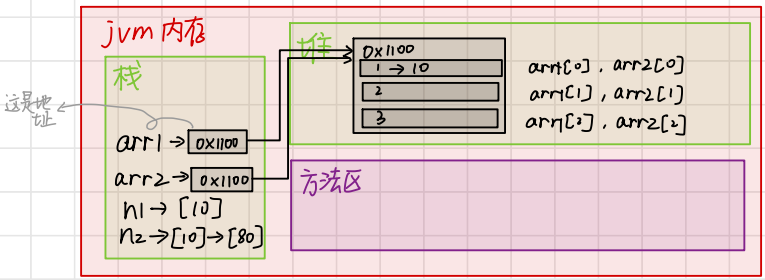
数组赋值机制 ☆☆

```
// 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细化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.

1、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