二维数组、面向对象

回顾

```
1.Eclipse的使用

2.数组的声明初始化方式
        声明
        int[] arr;
        int arr[];//不推荐使用
        初始化
        2.1静态初始化
          arr=new int[]{10,20,30};
          int[] arr2=new int[]{3,4,5};
          //简写 必须一条语句完成
          int[] arr3={1,3,5};
        2.2动态初始化
           int[] arr4=new int[4];//创建数组，长度4，数组元素都是默认值 0
          String[] arr5=new String[5];  //创建数组，长度5，数组元素都是默认值null

3.数组的使用（元素访问，元素修改，遍历）
          3.1 访问元素   使用下标 ，下标范围 0 - 长度-1
          3.2 获取长度   arr.length;
          3.3 修改元素
              arr[0]=10;
              arr[1]=20;
              arr[2]=30;
          3.4 数组遍历
            //for
             for(int i=0;i<arr.length;i++){
                    System.out.println(arr[i]);
             }
             //增强for  ,缺点不能访问下标
             for(int num:arr){
                     System.out.println(num);
             }
             注意：1 数组不要越界    ArrayIndexOutofBoundsException
                  2 数组一定要初始化   NullPointerException
          3.5数组内存空间的分配
            栈：存储基本类型数据和引用类型的地址 ,特点:先进后出   空间比较小，访问速度较快
            堆：存储引用类型的实际数据，特点:空间比较大，访问速度较慢

            int[] nums=new int[]{20,30,40};

            nums 数组 栈 地址
            {20，30，40}存放在堆中

4.数组的应用（排序和查找）
      排序：
          4.1 冒泡排序
              N个数字来排列
              两两比较小靠前
              外层循环n-1
              内层循环n-1-i
              for(int i=0;i<nums.length-1;i++){
```

```
            for(int j=0;j<nums.length-1-i;j++){
                if(nums[j]>nums[j+1]){
                   int temp=nums[j];
                   nums[j]=nums[j+1];
                   nums[j+1]=temp;
                }
            }
        }
```

4.2 选择排序

N个数字来排列

选择一个来比较

外层循环n-1

内层循环小于n

```
        for(int i=0;i<nums.length-1;i++){
            for(int j=i+1;j<nums.length;j++){
               if(nums[i]>nums[j]){
                 int temp=nums[i];
                 nums[i]=nums[j];
                 nums[j]=temp;
               }
            }
        }
```

优化

```
        for(int i=0;i<nums.length-1;i++){
           int k=i;
            for(int j=i+1;j<nums.length;j++){
               if(nums[k]>nums[j]){
                  k=j; //最小的数的下标
               }
            }
            if(k!=i){
               int temp=nums[k];
               nums[k]=nums[i];
               nums[i]=temp;
            }
        }
```

查找：

4.3 顺序查找

4.4 二分法查找

1 先排序

2 每次从中间开始，进行比较

```
        int low=0;
        int upper=nums.length-1;

        while(low<=upper){
           int middle=(low+upper)>>1;
           if(nums[middle]>key){
              upper=middle-1;
           }else if(nums[middle]<key){
              low=middle+1;
```

```
            }else{
                return middle;
            }
        }
        return -1;
```

今天内容

```
1.方法传参和返回值
2.二维数组
 1.1 二维数组的概念
 1.2 二维数组的定义
 1.3 数组的初始化
 1.4 二维数组的访问
3.调试和文档注释
4.面向对象
```

教学目标

```
1.掌握方法传参和返回值
2.掌握二维数组的初始化和遍历
3.掌握调试和文档注释
4.面向对象
```

第一节 方法的传参和返回值

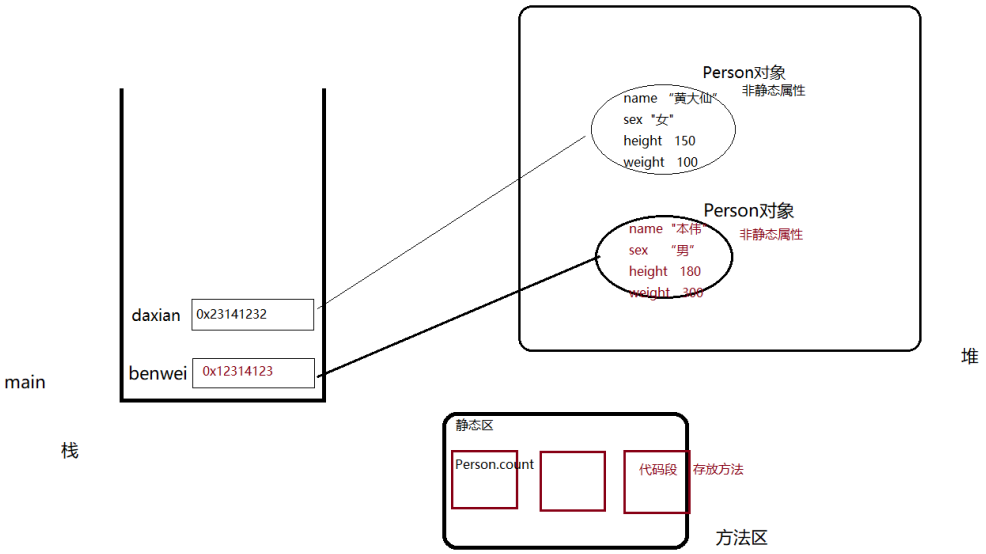
根据有没有参数：（1）无参方法 （2）有参方法

根据有没有返回值：（1）无返回值 （2）有返回值

2.1 基本类型作为方法参数

案例1:交换两个数字

```
public class Demo2 {
    public static void main(String[] args) {
        int num1=10;
        int num2=20;
        swap(num1, num2);
        System.out.println(num1);
        System.out.println(num2);
    }
    public static void swap(int a,int b) {
        System.out.println("交换之前: a:"+a+" b:"+b);
        int temp=a;
        a=b;
        b=temp;
        System.out.println("交换之后: a:"+a+" b:"+b);
    }

}
```

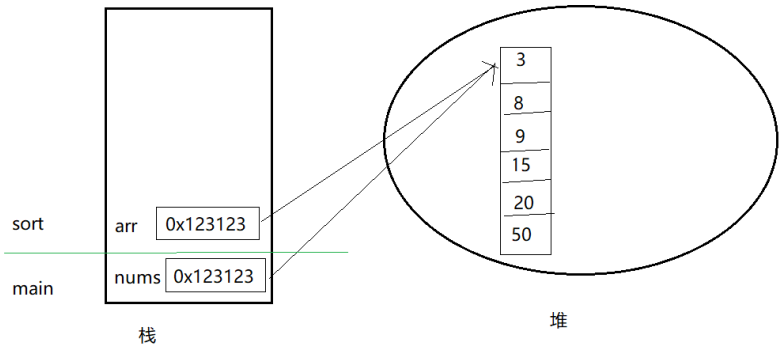


2.2 引用类型作为方法参数

案例2：实现数组排序

```
import java.util.Arrays;

/*
 * 方法的参数传递和返回值
 * 1 交换两个数字
 * 2 实现数组排序
 *
 */
public class Demo2 {
    public static void main(String[] args) {
        int[] nums=new int[] {15,8,3,9,20,50};
        sort(nums);
        //遍历nums
        System.out.println("遍历nums");
        for(int i=0;i<nums.length;i++) {
            System.out.print(nums[i]+" ");
        }
    }

    //2实现数组排序
    public static void sort(int[] arr) {
        //排序之前
        System.out.println("排序之前");
        System.out.println(Arrays.toString(arr));
        //冒泡
        for(int i=0;i<arr.length-1;i++) {
            for(int j=0;j<arr.length-i-1;j++) {
                if(arr[j]>arr[j+1]) {
                    int temp=arr[j];
```

```
                    arr[j]=arr[j+1];
                    arr[j+1]=temp;
                }
            }
        }
        //排序之后
        System.out.println("排序之后");
        System.out.println(Arrays.toString(arr));

    }

}
```



总结： Java中的方法传参采用的是传值的方式。

1 基本类型传递的实际数据

2 引用类型传递的是地址

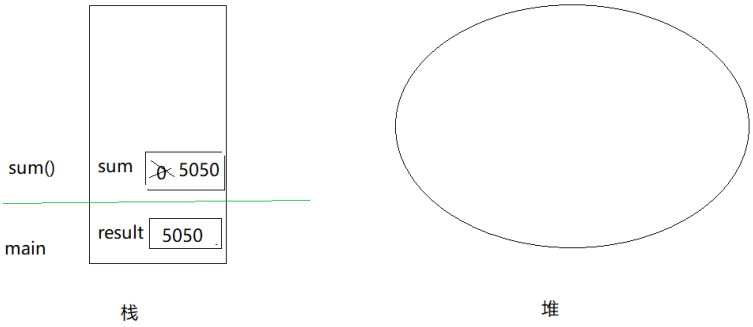
基本类型传递修改后对调用方没有任何影响。引用类型传递修改后对调用方有影响。

2.3 基本类型作为方法返回值

案例3 编写一个方法，实现1-100和，并返回结果

```
public class Demo3 {

    public static void main(String[] args) {
        int result=sum();
        System.out.println(result);
    }

    public static int sum() {
        //计算1-100的和
        int sum=0;
```

```
        for(int i=0;i<=100;i++) {
            sum+=i;
        }
        return sum;
    }
}
```

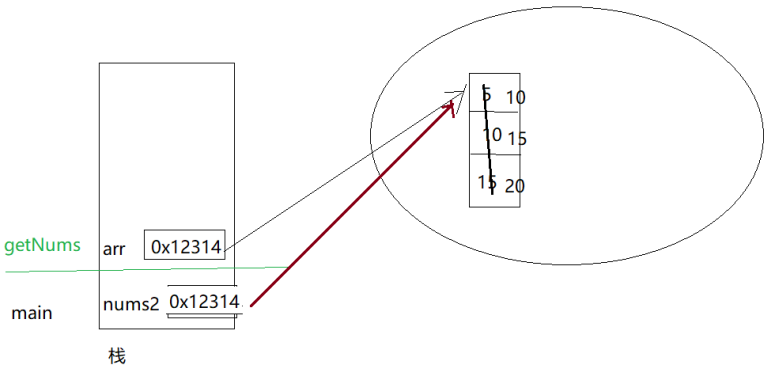


2.4 引用类型作为方法返回值

案例4：编写一个方法，返回一个数组

```
package com.qf.day07;

import java.util.Arrays;

/*
 * 方法返回值
 * 1 基本类型作为方法返回值
 * 2 引用类型作为方法返回值
 */
public class Demo3 {

    public static void main(String[] args) {
        int[] nums2=getNums();
        for(int i=0;i<nums2.length;i++) {
            System.out.println(nums2[i]);
        }
    }

    //把数组中的每个数字加5，再返回一个数组
    public static int[] getNums() {
        int[] arr=new int[] {5,10,15};
```

```
        for(int i=0;i<arr.length;i++) {
            arr[i]=arr[i]+5;
        }
        return arr;
    }
}
```



总结：方法的返回值如果是基本类型则返回实际数据，如果是引用类型则返回地址。

特殊：String类型传值方法和返回值遵循基本类型。

第二节 二维数组

3.1 二维数组的概念

```
本质上还是一个一维数组，只是其数组元素又是一个一维数组

    举例说明:变量，一维数组，二维数组之间的关系
    吸烟:
    没钱        1根          一个变量
    稍微有钱    一包         一维数组【20根】
    有钱        一条         二维数组【10包】
```

3.2 二维数组的定义

```
    方式一：元素类型[][] 数组名称;
    方式二：元素类型 数组名称[][];
    推荐使用方式一
```

3.3 数组的初始化

```
静态初始化：
语法：元素类型[][] 数组名称 = new 元素类型[][]{{一维数组1,一维数组2,一维数组3....};
简化：元素类型[][] 数组名称 =m{{一维数组1,一维数组2,一维数组3....};

举例：int[][] arr = new int[][]{{2,3},{5,2,1},{10,45,22,54}};
      int[][] arr = {{2,3},{5,2,1},{10,45,22,54}};

动态初始化：

语法：元素类型[][] 数组名称 = new 元素类型[二维数组的长度][一维数组的长度]
举例：int[][] arr = new int[3][4];
说明：定义一个数组arr，二维数组中一维数组的个数为3个，每个一维数组中元素的个数为4个
```

3.4 二维数组的访问

通过下标访问二维的指定元素

```
class TwiceArrayDemo01
{
    public static void main(String[] args)
    {
        int[][] arr = new int[3][4];

        System.out.println(arr);//[[I@15db9742
        System.out.println(arr.length);//3
        System.out.println(arr[0]);//[I@6d06d69c
        System.out.println(arr[0].length);//4
        System.out.println(Arrays.toString(arr));//[[I@6d06d69c, [I@7852e922,
[I@4e25154f]
        System.out.println(Arrays.toString(arr[0]));//[0, 0, 0, 0]

        /*
        [[I@15db9742
        3
        [I@6d06d69c
        4
        [[I@6d06d69c, [I@7852e922, [I@4e25154f]
        [0, 0, 0, 0]
        */
    }
}
```

遍历二维数组

```
//常见的操作：遍历二维数组
class TwiceArrayDemo02
{
    public static void main(String[] args)
    {
        //如果二维数组中一维数组的元素个数不确定
        //int[][] arr = new int[3][];

        int[][] arr = new int[][]{{2,3},{5,2,1},{10,45,22,54}};

        //遍历arr
        for(int i = 0;i < arr.length;i++) {
            System.out.println(arr[i]);
        }
```

```
        //赋值：给arr中的第1个元素修改值
        arr[1] = new int[2];

        //给arr[0]中的第0个元素修改值
        arr[0][0] = 10;

        //遍历arr[0]
        for(int i = 0;i < arr[0].length;i++) {
            System.out.println(arr[0][i]);
        }

        //二维数组的遍历：嵌套for循环
        //简单for循环
        for(int i = 0;i < arr.length;i++) {
            int[] subArr = arr[i];
            for(int j = 0;j < subArr.length;j++) {
                System.out.println(subArr[j]);
            }
        }

        //增强for循环
        for(int[] subArr1:arr) {
            for(int n:subArr1) {
                System.out.println(n);
            }
        }
    }
}
```

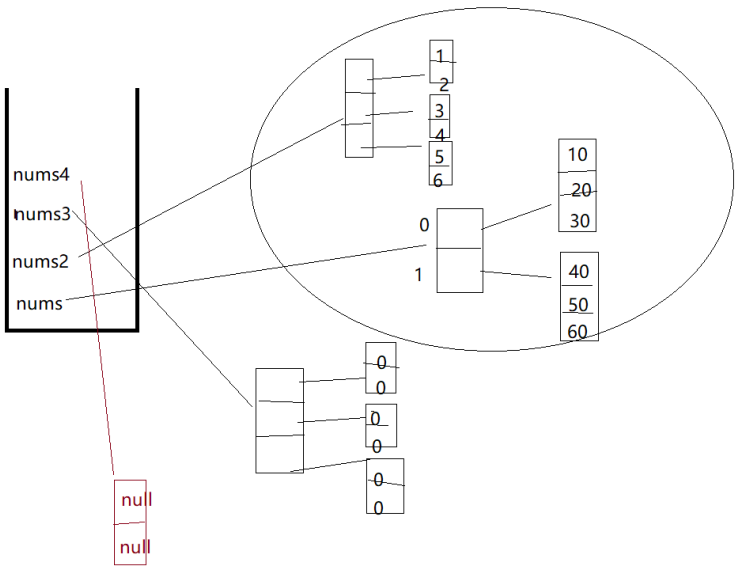
上机练习：一个班级有两门课程，5个学生，使用二维数组保存每门课的成绩，并计算每门课的总分和平均分

```
public static void main(String[] args) {
        double[][] scores=new double[2][5];
        Scanner input=new Scanner(System.in);
        for(int i=0;i<scores.length;i++) {
            for(int j=0;j<scores[i].length;j++) {
                System.out.println("请输入第"+(i+1)+"门课，第"+(j+1)+"个学生的成绩");
                scores[i][j]=input.nextDouble();
            }
        }
        System.out.println("----------------------------");
        for(int i=0;i<scores.length;i++) {
            double sum=0;
            for(int j=0;j<scores[i].length;j++) {
                sum+=scores[i][j];
            }
            System.out.println("第"+(i+1)+"门课的总分是:"+sum);
            System.out.println("第"+(i+1)+"门课的平均分是:"+(sum/5));

        }
    }
```

3.5 内存中的二维数组

```
public static void main(String[] args) {
        int[][] nums={{10,20,30},{40,50,60}};
        int[][] nums2={{1,2},{3,4},{5,6}};
        int[][] nums3=new int[3][2];
        int[][] nums4=new int[2][];
 }
```

画图分析：



第三节：调试技巧和文档注释

3.1调试技巧

使用eclipse的调试技巧，帮助开发人员理解程序执行过程。使用eclipse调试程序需要两个步骤：

- 1添加断点：
- 2单步执行：

  快捷键： F5单步进入 Step Into

  F6单步跳过 Step Over

  F7单步返回 Step Return

  F8继续执行 Resume

  查看变量窗口、断点窗口

3.2文档注释

java中注释

- 单行注释：//这里是单行注释
- 多行注释：/*这里是多行注释，

  可以多行 */

- JavaDoc注释：用来注释类、属性和方法等

  使用语法 /** .....*/

- JavaDoc常用标签

标签	含义	标签	含义
@author	标识一个类的作者，例如 @author wgy	@version	指定类的版本
@see	指定参考的链接或内容	@param	说明一个方法的参数
@since	标记一个特定的变化时，@since jdk1.2	@return	说明返回值类型

```
/**
 * 学校类
 * @author wgy
 * @version 2.0 2018/03/20
 */
public class School {
    /**
    *学校名称
    */
    String schoolName;
    /**
     * 招生方法
     * @return total
     */
    public void drawStudent() {
        System.out.println(schoolName+"开始招生...");
    }
    //...
}
```

文档注释作用：

1.编写代码时可通过提示显示文档注释

2.JavaDoc能够从源代码中抽取类、属性、方法等的注释，形成一个配套的API帮助文档 。

演示1：eclipse生成帮助文档。注意编码 -encoding utf-8 -charset utf-8

演示2：javadoc -d doc -encoding utf-8 -charset utf-8 src\com\qf\day23_3\Person.java

总结

1 方法的参数传递和返回值

1 方法的参数传递采用传值方式： 基本类型传递的实际数据 引用类型传递的地址

2 方法的返回值，返回数据，基本类型返回实际数据，引用类型返回地址。

2 二维数组，实际上是由一维数组组成，一维数组的每个元素还是一个数组。

3 二维数组的声明、初始化、访问(通过下标访问，遍历)

```
int[][] arr=new int[][]{{10,20},{30,40}}
```

4 调试技巧

1 添加断点

2 单步执行　　F6单步跳过　　F5单步进入　F8继续执行　　ctrl+f12 终止调试

5 文档注释

帮助开发人员生成API文档

默写

根据下面要求完成题目：
1.分别使用静态初始化和动态初始化的方式定义一个数组
2.对静态初始化的数组分别使用冒泡和选择进行排序，其中，冒泡实现升序，选择实现降序
3.使用for循环和foreach遍历排好序的数组

作业

1.班上有3个学生，每个学生都参加了三门功课的考试，其中第二个学生是特长生，上级要求给他每门功课都+5.【要求：使用二维数组做，并且分别使用for循环和增强for循环遍历二维数组】
2.求一个3*3矩阵对角线元素之和

```
10  20  30
8   6   7
20  25  50
```

面试题

1.二维数数组在内存中的存储方式是怎样的？