

3月24日作业 array2 王世杰

简答题

1. 数组操作中会碰到哪些异常？是什么原因导致的？

NullPointerException

这个异常的产生可能是因为源数组或是目的数组并没有引用到一个数组的实例，也就是说数组引用的值为**NULL**时，就会产生这个异常。

ArrayIndexOutOfBoundsException

当数组索引值指定错误时或是起始位置给错了，也可能是拷贝数据的个数超出数组的大小范围。

ArrayStroeException

产生的原因可能是来源或是目的根本就不是数组，或者是来、目的数组不是基本数据类型的数组。

2. 数组遍历方式中，普通for循环和增强for循环有什么不同？它们的使用场景是什么样的？

for增强可以更高效，但是不能改变数组的值
若只需要读取值用**for**增强，其他用**for**循环。

3. 什么是值传递？什么是引用传递？Java方法的传值方式是什么？这意味着什么？（方法能够对传入的实参数值做什么操作？）

值传递是复制一份不改变实参，引用传递会改变实参。

java是值传递，意味着**java**中函数不能直接改变基本数据类型的实参，但可以通过改变引用拷贝对应的堆上的对象来改变引用数据类型实参。

4. 可变参数的本质是什么？在方法体中怎么使用可变参数？

是数组，可变参当做数组用

4. 使用递归要注意什么？

不要死循环，递归调用栈不宜过大，会溢出

6. 递归的优缺点是什么？

递归的缺点：

1. 递归很危险，容易栈溢出

2. 递归是分解问题求解的过程，在这个过程中需要频繁调用方法，频繁求解，往往存在大量重复计算的部分，这导致递归求解不仅耗费内存，而且效率也很差，时空复杂度都不优越。

编程题

数组基本使用练习

1. 定义一个double数组用来存放学生成绩，然后键盘录入10位同学的成绩并存入数组，求这10位同学成绩的平均值。
2. 定义一个String数组，输出该数组的长度，并用多种方式遍历打印数组元素（常见的方式遍历即可）
注：
3. 请合理使用方法，不要胡子眉毛一把抓把代码全部写在main方法里。
4. 数组遍历的方式，比如for、增强for或者使用工具类等等。

```
package com.cskaoyan.homework.day03;

import java.util.Scanner;

/**
 * 定义一个double类型的数组，让数组中每个元素（包括首位元素）都除以首位元素，
 * 得到的结果作为该位置上的新元素。请在原先数组数组上操作，并打印新数组。
 * 例如数组[ 2.0 , 4.0 , 6.0 , 4.0 ] 经过方法运算得到新数组 [ 1.0 ,2.0 , 3.0 ,2.0 ]
 * 注：注意元素取值的变化。
 *
 * @author 王世杰
 * @created 2022/3/23 12:48
 */

public class Solution {

    public static double[] divisionFirst(double[] arr) {
        for (int i = arr.length - 1; i >= 0; i--) {
            arr[i] = arr[i] / arr[0];
        }
        return arr;
    }

    public static void printArr(double[] arr) {
        for (double v : arr) {
            System.out.print(v + " ");
        }
        System.out.println();
    }

    private static double[] getArr() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("请输入数组长度: ");
        int arrLength = scanner.nextInt();
        double[] arr = new double[arrLength];
        System.out.print("请输入"+arrLength+"个数: ");
        for (int i = 0; i < arrLength; i++) {
            arr[i] = scanner.nextDouble();
        }
        return arr;
    }

    public static void main(String[] args) {
        double[] arr = getArr();
    }
}
```

```

        System.out.print("操作后的新数组为: ");
        printArr(divisionFirst(arr));
    }
}

```

```

Solution x Solution (1) x
C:\eve\jdk\corretto-1.8.0_322\bin\java.exe ...
input ten scores:
1 2 3 4 5 6 7 8 9 10
the ten scores are: [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]
the average score is :5.5
the length of the array is 4
the String array are: [张三, 李四, 王二麻子, 张全蛋]
the length of the array is 4
the String array are: [张三, 李四, 王二麻子, 张全蛋]

Process finished with exit code 0

```

数组综合练习——删除元素

对于某个String类型数组，将其中的某个元素全部去掉，得到一个新数组，并统计去掉了几个元素。

举例，对于String数组["abc", "123", "123", "123", "666", "777"] 将其中的元素"123"全部去掉，就得到了新数组["abc", "666", "777"]，一共去掉了3个元素。

注：

1. 这里说的去掉，不是指用0/null等默认值替代原先的元素，而是指真正的删除掉。
2. 思考：同一个数组能不能实现这个功能？如果不能，应该怎么完成呢？

```

package com.cskaoyan.homework.day04;

import java.util.Arrays;
import java.util.Objects;

/**
 * since 2022/3/24 19:18
 * authored by BIGSHAOSHI
 */

public class DeleteElement {
    public static String[][] deleteSameElement(String[] strings, String str) {
        int count = 0;
        for (int i = 0; i < strings.length - count; i++) {
            if (strings[i] == str) {
                for (int j = i; j < strings.length - 1; j++) {
                    strings[j] = strings[j + 1];
                }
                count++;
                i--;
            }
        }
    }
}

```

```

String[] resStrings = new String[strings.length - count];
System.arraycopy(strings, 0, resStrings, 0, resStrings.length);
String[] resCount = new String[]{String.valueOf(count)};

//返回二维数组：其中 [0] 为 删除元素后的最终数组 [1] 为删除元素的个数以字符串形式存
储在[0]号位
return new String[][]{resStrings, resCount};
}

public static void main(String[] args) {
String[] strings = {"123", "adc", "a", "123", "sda", "asa",
"weqw", "123"};
String str = "123";
System.out.println("删除前: " + Arrays.toString(strings));
String[][] returnDoubleArrays = deleteSameElement(strings, str);
strings = returnDoubleArrays[0];
int count = Integer.parseInt(returnDoubleArrays[1][0]);
System.out.print("删" + count + "个\"" + str + "\"");
System.out.println("后: " + Arrays.toString(strings));
}
}

```

```

DeleteElement x ToBinary x
C:\eve\jdk\corretto-1.8.0_322\bin\java.exe ...
删除前: [123, adc, a, 123, sda, asa, weqw, 123]
删3个"123"后: [adc, a, sda, asa, weqw]

Process finished with exit code 0

```

值传递练习

Java有且仅有值传递，Java中的方法不能直接改变实参本身。

对于下列代码：

```

public static void main(String[] args) {
int a = 10;
int b = 20;
method(a, b);
System.out.println("a = " + a);
System.out.println("b = " + b);
}

public static void method(int a, int b) {
a *= 2;
b *= 2;
}

```

思考以下问题：

1. 对于以上Java代码，method()方法可以把a，b的取值变为原先的2倍吗？如果不能，为什么？
2. 上述main方法不变，修改method方法实现，让程序输出：

1. a = 20
2. b = 40

如何实现呢？

注：这是个脑筋急转弯，不会的可以互相问问或者查一查。

```
1. 不能，因为是值传递
2.
public static int[] method(int a, int b) {
    System.out.println("a = 20" );
    System.out.println("b = 40" );
    System.exit(1);
}
```

可变参数练习

可变参数的本质是数组，完成下列需求：

求不限定个数参数的最大值（要求使用可变参数，使用int类型即可）

```
package com.cskaoyan.homework.day03;

import java.lang.reflect.Array;
import java.util.Arrays;

/**
 * @author 王世杰
 * @created 2022/3/24 12:20
 */

public class GetMax {
    // 求不限定个数参数的最大值（要求使用可变参数，使用int类型即可）

    public static int getMax(int... number){
        int maxNum = number[0];
        for (int num : number) {
            if (num > maxNum) {
                maxNum = num;
            }
        }
        return maxNum;
    }

    public static void main(String[] args) {
        int[] nums = {1,2,3,4,5,6,7,8,9};
        System.out.println(Arrays.toString(nums)+"中最大值为: "+getMax(nums));
    }
}
```

```
Solution x GetMax x
C:\eve\jdk\corretto-1.8.0_322\bin\java.exe ...
[1, 2, 3, 4, 5, 6, 7, 8, 9]中最大值为: 9

Process finished with exit code 0
```

数组综合练习——合并数组（扩展）

首先准备两个数组（简单起见，用两个int数组即可）

然后写方法，将这两个数组合并。

注：

1. 数组合并也是很常见的操作，实际开发中有很多现成的手段去完成它。由于有些手段是超纲的，所以这里不再细表。
2. 如果后续工作中碰到了，可以再百度学习一下。目前就先考虑自己手写实现就可以了。

```
package com.cskaoyan.homework.day04;

import java.util.Arrays;

/**
 * @author 王世杰
 * @created 2022/3/24 12:26
 */

public class MergeArrays {
    public static int[] mergeArrays(int[] nums1, int[] nums2) {
        int[] nums = new int[nums1.length + nums2.length];
        int index1 = 0;
        int index2 = 0;
        int index3 = 0;

        while (index3 < nums1.length + nums2.length) {
            if (index1 == nums1.length) {
                for (int i = index2; i < nums2.length; i++) {
                    nums[index3++] = nums2[i];
                }
                break;
            }
            if (index2 == nums2.length) {
                for (int i = index1; i < nums1.length; i++) {
                    nums[index3++] = nums1[i];
                }
            }
            if (nums1[index1] > nums2[index2]) {
                nums[index3++] = nums2[index2++];
            } else {
                nums[index3++] = nums1[index1++];
            }
        }
    }
}
```

```

        nums[index3++] = nums1[index1++];
    }
}
return nums;
}

public static void main(String[] args) {
    int[] nums1 = {1, 3, 5, 7, 9};
    int[] nums2 = {2, 4, 6, 8, 10};
    int[] mergerNums = mergeArrays(nums1, nums2);
    System.out.println("合并前的两个数组: ");
    System.out.println(Arrays.toString(nums1));
    System.out.println(Arrays.toString(nums2));
    System.out.println("合并后的数组: ");
    System.out.println(Arrays.toString(mergerNums));
}
}

```

```

Solution x MergeArrays x
C:\eve\.jdk\corretto-1.8.0_322\bin\java.exe ...
合并前的两个数组:
[1, 3, 5, 7, 9]
[2, 4, 6, 8, 10]
合并后的数组:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Process finished with exit code 0

```

练习使用递归

递归在实际开发中用途并不广泛，但我们仍有必要学习它的基本使用。

这道题目如果想不明白，那就百度一下吧。

使用递归，把十进制正整数（ $N \geq 0$ ）转换成二进制数

```

package com.cskaoyan.homework.day04;

import java.util.Arrays;
import java.util.Scanner;

/**
 * @author 王世杰
 * @created 2022/3/24 12:52
 */

public class ToBinary {

    public static void toBinary(int n){
        if(n/2==0)

```

```

        System.out.print(n%2);
    }
    else
        toBinary(n/2);
    System.out.print(n%2);
}

public static void main(String args[]){
    System.out.print("请输入一个10进制整数: ");
    int num =new Scanner(System.in).nextInt();
    System.out.print("它的2进制表示为: ");
    toBinary(num);
}
}

```

```

Solution x ToBinary x
C:\eve\jdk\corretto-1.8.0_322\bin\java.exe ...
请输入一个10进制整数: 123
它的2进制表示为: 11111011
Process finished with exit code 0

```

数组排序 (扩展)

随意给出一个长度为10的int数组，然后升序排列它。

注:

1. 如果想自己写排序实现，可以使用最简单的冒泡排序或者选择排序。（不会就百度一下）
2. 排序对于数组而言是非常常见的操作，Java源码工具类中，早有对应的实现。请百度查询一下，不需要了解它具体怎么实现的，会用即可。（百度并学习是非常重要的能力，我们不可能在课堂上学会所有的知识）

```

package com.cskaoyan.homework.day04;

import java.util.Arrays;

/**
 * @author 王世杰
 * @created 2022/3/24 13:05
 */

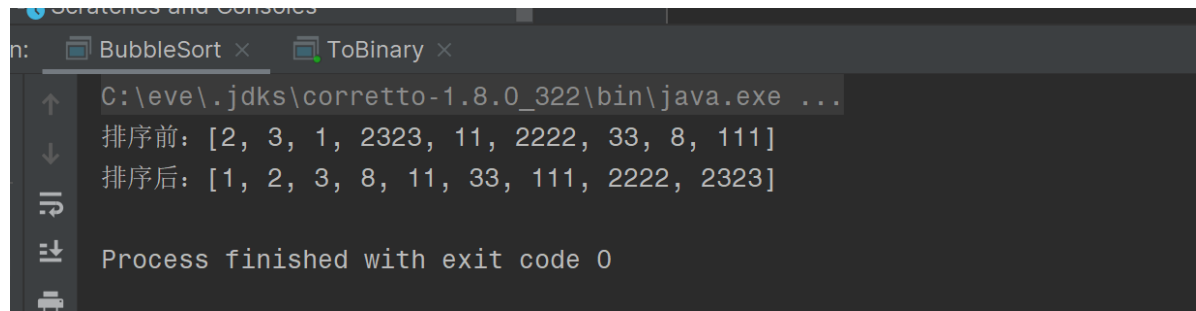
public class BubbleSort {

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

```



```
    }  
    }  
    return nums;  
}  
  
public static void main(String[] args) {  
    int[] nums = {2,3,1,2323,11,2222,33,8,111};  
    System.out.println("排序前: "+Arrays.toString(nums));  
    nums = bubbleSort(nums);  
    System.out.println("排序后: "+Arrays.toString(nums));  
}  
}
```



```
Scratches and Consoles  
n: BubbleSort x ToBinary x  
C:\eve\jdk\corretto-1.8.0_322\bin\java.exe ...  
排序前: [2, 3, 1, 2323, 11, 2222, 33, 8, 111]  
排序后: [1, 2, 3, 8, 11, 33, 111, 2222, 2323]  
Process finished with exit code 0
```