# day02 数组、字符串练习

## 数组是否已排序

写一个方法，传入一个整型数组，判断数组里面所有的元素是否已排序。

方法定义如下：public static boolean isSorted(int[] numbers);

传入数组{2,8,10,12,33}时，返回true

传入数组{2,4,3,9,10}时，返回false

**package** day02;

**import** java.util.Arrays;

**public** **class** ArrayOne {

**public** **static** **void** main(String[] args) {

**int**[] numbers1= {2,8,10,12,33};

**int**[] numbers2= {2,4,3,9,10};

System.***out***.println(Arrays.*toString*(numbers1));

System.***out***.println(*isSorted*(numbers1));

System.***out***.println(Arrays.*toString*(numbers2));

System.***out***.println(*isSorted*(numbers2));

}

**public** **static** **boolean** isSorted(**int**[] numbers) {

**boolean** flag=**true**;

**for**(**int** i=0;i<numbers.length-1;i++)

{

**for**(**int** j=i+1;j<numbers.length;j++)

{

**if**(numbers[i]>numbers[j])

{

flag=**false**;

**break**;

}

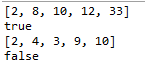
}

}

**return** flag;

}

}



## 数组倒置

写一个方法，传入一个整型数组，方法内部将数组元素倒置。

方法定义如下：public static void reverse(int[] numbers);

传入数组{2,8,10,12,33}时，经过倒置操作之后，变成 {33， 12， 10， 8， 2}

**import** java.util.Arrays;

**public** **class** ArrayTwo {

**public** **static** **void** main(String[] args) {

**int**[] numbers= {2,8,10,12,33};

System.***out***.println("原数组为：");

System.***out***.println(Arrays.*toString*(numbers));

*reverse*(numbers);

System.***out***.println("倒置后的数组为：");

System.***out***.println(Arrays.*toString*(numbers));

}

**public** **static** **void** reverse(**int**[] numbers) {

**if**(numbers.length%2==0)

{

**for** (**int** i=0;i<=numbers.length/2-1;i++) {

**int** temp=numbers[i];

numbers[i]=numbers[numbers.length-1-i];

numbers[numbers.length-1-i]=temp;

}

}**else** {

**for**(**int** i=0;i<(**int**)(numbers.length/2);i++)

{

**int** temp=numbers[i];

numbers[i]=numbers[numbers.length-1-i];

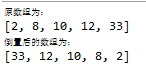
numbers[numbers.length-1-i]=temp;

}

}

}

}



## 数组打乱顺序

写一个方法，传入一个排好序整型数组，方法会将数组元素的顺序随机互换：

遍历数组，对每一个下标为i的元素，随机产生一个新的下标j(在数组长度范围之内)，然后将i和j位置的元素互换。

方法定义如下：public static void shuffle(int[] numbers);

传入数组{2,8,10,12,33}

**import** java.util.Arrays;

**public** **class** ArrayThree {

**public** **static** **void** main(String[] args) {

**int**[] numbers= {2,8,10,12,33};

System.***out***.println("原数组为：");

System.***out***.println(Arrays.*toString*(numbers));

*shuffle*(numbers);

System.***out***.println("打乱后的数组为：");

System.***out***.println(Arrays.*toString*(numbers));

}

**public** **static** **void** shuffle(**int**[] numbers) {

**int** temp=0;

**for**(**int** i=0;i<numbers.length;i++)

{

**int** j=(**int**)(Math.*random*()\*numbers.length);

temp=numbers[i];

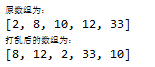
numbers[i]=numbers[j];

numbers[j]=temp;

}

}

}



## 连续相等的数判断

写一个方法，传入一个整型数组，判断数组里面是否有4个连续相等的整数。

方法定义如下：public static boolean hasContinuousFour(int[] numbers);

传入数组{8,9,9,9,9,8}时，返回true

传入数组{8,9,9,9,8}时，返回false

**import** java.util.Arrays;

**public** **class** ArrayFour {

**public** **static** **void** main(String[] args) {

**int**[] numbers1= {8,9,9,9,9,8};

**int**[] numbers2= {8,9,9,9,8};

System.***out***.println(Arrays.*toString*(numbers1));

System.***out***.println(*hasContinuousFour*(numbers1));

System.***out***.println(Arrays.*toString*(numbers2));

System.***out***.println(*hasContinuousFour*(numbers2));

}

**public** **static** **boolean** hasContinuousFour(**int**[] numbers) {

**boolean** ishasContinuousFour=**false**;

**for**(**int** i=0;i<numbers.length-4;i++) {

**if**(numbers[i]==numbers[i+1]&&numbers[i+1]==numbers[i+2]&&numbers[i+2]==numbers[i+3]) {

ishasContinuousFour=**true**;

**break**;

}

}

**return** ishasContinuousFour;

}

}

