数组操作

遍历数组

1. 用for循环遍历数组

```
    public static void main(String[] args) {
        int array[]={1,3,5,7,9,2,3,4};
        for(int i =0;i<array.length;i++)
        {
            System.out.println(array[i]);
        }
    }
```

2. 用for-each遍历数组

   for-each每次直接获取数组的值进行迭代，但无法获取数组的引索。

```
    public static void main(String[] args) {
    String s[]={"A","E","I","O","U"};
    for (String ar:s)
    {
        System.out.print(ar+" ");
    }
   }
```

打印数组

1. 用for-each进行时数组打印，可以输出想要的数组格式

```
    public static void main(String[] args) {
        int ns[]={1,2,3,4,2,4,5};
        for(int num:ns)
        {
            System.out.print(num+" ");
        }
    }
```

2. 用java自带的Arrays.toSring类打印，但是会输出"[]"

```
//Arrays.toString
    public static void main(String[] args) {
        String s[] = {"hello","world","!"};
```

```
        System.out.println(Arrays.toString(s));
    }
```

直接打印数*组会获*取到数*组*所在的地址内容。

数组排序

1. 冒泡排序（简单优化）

```
//优化后的冒泡排序
   public static void main(String[] args) {
    int ns[]={1,2,5,3,6,5,76,4,231,3,2};
    System.out.println("Original Arrays: " + Arrays.toString(ns));
    boolean flag = true;                        //冒泡排序优化，用flag判断该轮循环中
是否有交换。没有交换则排序已经完成。

    for (int i =0;i< ns.length-1;i++)
    {
        if(flag) {
            flag = false;                       //每开始一个新的排序，将flag标志初始
化为false.若进入循环才置为true.
            for (int j = 0; j < ns.length - i - 1; j++) {
                if (ns[j] > ns[j + 1]) {
                    int temp;
                    temp = ns[j + 1];
                    ns[j + 1] = ns[j];
                    ns[j] = temp;
                    flag = true;
                }
            }
        }
        else break;
    }
   }
```

2. Sort排序方法的调用

```
//Java自带的排序方法sort
    public static void main(String[] args) {
        int ns [] = {1,5,2,3,6,4,5};
        Arrays.sort(ns);
        System.out.println(Arrays.toString(ns));
    }
```

小结

常用的排序算法有冒泡排序、插入排序和快速排序等；

冒泡排序使用两层for循环实现排序；

交换两个变量的值需要借助一个临时变量。

可以直接使用Java标准库提供的Arrays.sort()进行排序；

对数组排序会直接修改数组本身。

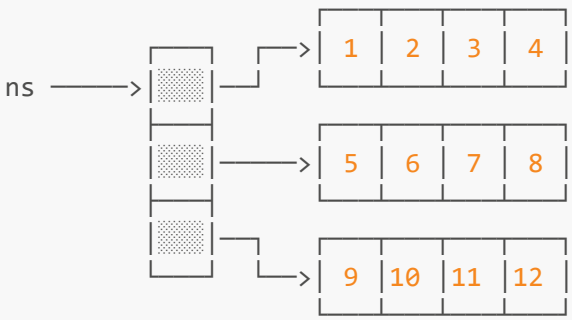
多维数组

二位数组

1. 二维数组就是数组的数组

```
//二维数组
public class Main {
    public static void main(String[] args) {
        int[][] ns = {
            { 1, 2, 3, 4 },
            { 5, 6, 7, 8 },
            { 9, 10, 11, 12 }
        };
        System.out.println(ns.length); // 3
    }
}

//二维数组在内存中的存放形式
```



2. 访问二维数组的某个元素需要使用`array[row][col]`
3. 二维数组的打印 s i. 用两个for循环

```
 public static void main(String[] args) {
    int ns[][]={
            {1,2,3,4},
            {1,3,5,7},
            {2,4,6,8}
    };
    for(int [] arr:ns)
    {
        for(int num:arr)
        {
```

```
            System.out.print(num);
            System.out.print(", ");
        }
    }
}

//Output
1, 2, 3, 4, 1, 3, 5, 7, 2, 4, 6, 8,
```

ii. 使用Java标准库的Arrays.deepToString()

```
    public static void main(String[] args) {
     int ns[][]={
             {1,2,3,4},
             {1,3,5,7},
             {2,4,6,8}
     };
     System.out.print(Arrays.deepToString(ns));
}

//Output
[[1, 2, 3, 4], [1, 3, 5, 7], [2, 4, 6, 8]]
```

小结

二维数组就是数组的数组，三维数组就是二维数组的数组；

多维数组的每个数组元素长度都不要求相同；

打印多维数组可以使用Arrays.deepToString()；

最常见的多维数组是二维数组，访问二维数组的一个元素使用array[row][col]。

命令行参数

Java程序的入口是`main`方法，而`main`方法可以接受一个命令行参数，它是一个`String[]`数组。

这个命令行参数由JVM接收用户输入并传给`main`方法

我们可以利用接收到的命令行参数，根据不同的参数执行不同的代码。例如，实现一个`-version`参数，打印程序版本号：

```
public class Main {
    public static void main(String[] args) {
        for (String arg : args) {
            if ("-version".equals(arg)) {
                System.out.println("v 1.0");
                break;
            }
```

```
            }
        }
    }
```

改程序必须在命令行编译运行。

小结

命令行参数类型是String[]数组；

命令行参数由JVM接收用户输入并传给main方法；

如何解析命令行参数需要由程序自己实现。