 山西农业大学软件学院课堂笔记

**课程名称：**

**班级：1707原1704学号：20171611445 姓名：张钊源 日期：2020 年 7 月 23 日**

数组是一组数据的集合，数组中的每个数据被称为元素。在java中,数组也是对象。数组中的元素可以是任意类型(基本类型和引用类型),但同一个数组里只能存放类型相同的元素。

数组类型的变量、类类型的变量、接口类型的变量都属于引用类型的变量,这些变量都是可以指向其相应的对象的,那么数组类型的变量指向的就是数组对象。

1)数组类型

8种基本数据类型和3种引用类型,使用这些都可以声明出数组类型

即:任意类型都可以声明成数组类型,只需要加"[]"即可

例如:

基本类型的数组:

byte[] short[] int[] long[] float[] double[] char[] boolean[]

引用类型的数组:

//使用类声明出的数组

Student[]

//使用接口声明出的数组

Action[]

//使用数组声明出的数组(也就是二维数组)

int[][]

注意:使用X类型声明出的数组,那么将来该数组中就只能存放X类型的数据

创建数组对象

1)数组对象的长度

数组的长度是指一个数组对象中可以存放多少个数据(元素)。

每一个数组对象在被创建的时候,就需要指明其长度是多少,并且该长度一确定的就不能再做修改。

2)数组对象的创建

创建数组对象使用new关键字即可。

3)获得数组对象的长度

使用数组对象访问其length属性即可获得该数组的长度

由于数组对象在堆区中是一块连续的内存空间,那么我们可以通过连续的下标,找到每一个可以存放数据的元素位置

数组下标越界

对于一个长度为n的数组对象,它的的下标取值范围是[0,n-1],这里的0和n-1就是这个数组对象的下标边界,使用下标的过程中不能超出这个边界。如果超出那么就会运行报错

数组对象的拷贝

数组对象的长度确定之后便不能修改,但我们可以通过复制数组的内容变通实现改变数组长度。

在java.lang.System类中提供一个名为arraycopy的方法可以实现复制数组中元素的功能

//该方法的声明

public static void arraycopy(Object src,

int srcPos,

Object dest,

int destPos,

int length)

参数1,需要被复制的目标数组

参数2,从这个数组的那个一个位置开始复制

参数3,需要把数据复制到的另外的那一个新的数组对象

参数4,复制到新数组里面哪个位置(从这个位置开始算)

参数5,复制的目标数组的长度

由于数组对象本身并没有什么方法可以供我们调用,但API中提供了一个工具类Arrays供我们使用,从而可以对数据对象进行一些基本的操作.

Arrays类中的方法都是static修饰的静态方法,在使用的时候可以直接使用类名进行调用,而"不用"使用对象来调用(注意:是"不用" 而不是 "不能")

Arrays类中常用方法

toString方法,把数组转换位字符串形式并返回

例如:

int[] a = {1,3,5,7,9};

System.out.println(Arrays.toString(a));

//输出结果: [1, 3, 5, 7, 9]

binarySearch方法,在数组中查找指定元素并返回其下标

例如:

int[] a = {1,3,5,7,9};

int key = Arrays.binarySearch(a,7);

System.out.println(key);

//输出结果: 3

copyOf方法,复制或者截取指定数组并返回

例如: 复制

int[] a = {1,3,5,7,9};

a = Arrays.copyOf(a, 8);

System.out.println(Arrays.toString(a));

//输出结果: [1, 3, 5, 7, 9, 0, 0, 0]

例如: 截取

int[] a = {1,3,5,7,9};

a = Arrays.copyOf(a, 2);

System.out.println(Arrays.toString(a));

//输出结果: [1, 3]

copyOfRange方法,将数组中指定范围复制新数组并返回

例如:

int[] a = {1,3,5,7,9};

a = Arrays.copyOfRange(a,1,3);

System.out.println(Arrays.toString(a));

//输出结果: [3, 5]

equals方法,比较俩个数组是否相等

例如:

int[] a = {1,3,5,7,9};

int[] b = {1,3,5,7,9};

System.out.println(Arrays.equals(a, b));//true

System.out.println(a==b);//false

注意:==比较的是引用所指向对象的内存地址

Arrays.equals方法比较是俩个数组中的内容

fill方法,用指定值去填充数组对象

例如:

int[] a = {1,3,5,7,9};

Arrays.fill(a, 99);

System.out.println(Arrays.toString(a));

//输出结果:[99, 99, 99, 99, 99]

sort方法,把数据中的元素进行排序

例如:

int[] a = {3,5,1,9,7};

System.out.println("before:\t"+Arrays.toString(a));

Arrays.sort(a);

System.out.println("after:\t"+Arrays.toString(a));

//输出结果:

before: [3, 5, 1, 9, 7]

after: [1, 3, 5, 7, 9]

**评分： 任课教师： 第 页,共 页**