**数组作业shu**

1. **填空题**
2. **数组会在内存中开辟一块\_\_\_连续\_\_\_\_\_的空间，每个空间相当于之前的一个变量，称为数组的元素。数组的长度一经确定，就无法再改变。**
3. **要获取一个数组的长度，可以通过\_\_\_.length\_\_\_属性来获取，但获取的只是为数组分配的空间的数量，而不是数组中实际已经存放的元素的个数。**
4. **\_int[] array\_\_仅仅是给出了数组名字和元素的数据类型，要想真正的使用数组还必须使用new关键字为它分配内存空间。**
5. **创建数组后，系统会给每一个数组元素一个默认的值，如String类型元素的默认值是\_\_\_\_\_null\_\_\_\_\_\_\_。**
6. **在Java中有二维数组int [ ] [ ] array={{1,2,3},{4,5}} ，可以使用\_array[1].length\_\_\_\_\_\_\_\_\_得到二维数组中第二维中第一个数组的长度。**
7. **数组元素下标(或索引)的范围是\_\_\_[0,数组长度-1]\_\_\_\_\_\_\_\_。**
8. **选择题**

|  |  |  |
| --- | --- | --- |
| **1.** | **在Java中，以下程序能正确为数组赋值的是（ACD**  **）。（选择二项）** | |
|  |  |  |
|  | **A** | **int a[]={1,2,3,4};** |
|  | **B.** | **int b[4]={1,2,3,4};** |
|  | **C.** | **int c[];**  **c=new int[] {1,2,3,4};** |
|  | **D.** | **int d[];**  **d=new int[]{1,2,3,4};** |

|  |  |  |
| --- | --- | --- |
| **2.** | **数组元素的索引可以是（ D ）。（选择一项）** | |
|  |  |  |
|  | **A** | **整型常量**  **scores[0]** |
|  | **B.** | **整型变量**  **scores[i]** |
|  | **C.** | **整型表达式**  **scores[i+1]** |
|  | **D.** | **以上都可以** |

|  |  |  |
| --- | --- | --- |
| **3.** | **已知表达式int [] m={0,1,2,3,4,5,6};下面（ B ）表达式的值与数组最大下标数相等。（选择一项）** | |
|  |  |  |
|  | **A** | **m.length()** |
|  | **B.** | **m.length-1** |
|  | **C.** | **m.length()+1** |
|  | **D.** | **m.length+1** |

|  |  |  |
| --- | --- | --- |
| **4.** | **在Java中，以下定义数组的语句正确的是（ C）。（选择二项）** | |
|  |  |  |
|  | **A** | **int t[10]=new int[ ];** |
|  | **B.** | **char [ ]a=new char[5];**  **char []a={‘a’,’b’};** |
|  | **C.** | **String [ ] s=new String [10];** |
|  | **D.** | **double[ ] d [ ]=new double [4][ ];**  **double[][] d;**  **double d[][];** |
| **5.** | **在Java中,下面代码的输出结果为（ A）。（选择一项）** | |
|  | **public static void main(String[] args) {**  **int[] arrA = { 12, 22, 8, 49, 3 };**  **int k = 0;**  **int len = arrA.length;**  **for (int i = 0; i < len; i++) {**  **for (int j = i + 1; j < len; j++) {**  **if (arrA[i] > arrA[j]) {**  **k = arrA[i];**  **arrA[i] = arrA[j];**  **arrA[j] = k;**  **}**  **}**  **}**  **for (int i = 0; i < arrA.length; i++) {**  **System.*out*.print(arrA[i]);**  **if (i < arrA.length - 1) {**  **System.*out*.print("，");**  **}**  **}**  **}** | |
|  |  |  |
|  | **A** | **3，8，12，22，49** |
|  | **B.** | **12，22，8，49，3** |
|  | **C.** | **49，22，12，8，3** |
|  | **D.** | **编译错误** |

|  |  |  |
| --- | --- | --- |
| **6.** | **分析下面的Java源程序，编译后的运行结果是（ B ）。（选择一项）** | |
|  | **import java.util.\*;**  **public class Test {**  **public static void main(String[ ] args) {**  **int [ ] numbers=new int[ ]{1,2,3};**  **System.*out*.println(Arrays.*binarySearch*(numbers, 2));**  **}**  **}** | |
|  |  | **输出** |
|  | **A** | **输出：0** |
|  | **B.** | **输出：1** |
|  | **C.** | **输出：2** |
|  | **D.** | **输出：3** |

|  |  |  |
| --- | --- | --- |
| **7.** | **以下选项中关于Java中方法的可变参数的说法正确的是（ C D）。（选择二项）** | |
|  |  |  |
|  | **A** | **可变参数是JDK1.5增加的内容，用来表示方法的形参 int … a** |
|  | **B.** | **一个方法可以没有可变参数，可以有1个或者多个可变参数** |
|  | **C.** | **可变参数可以被当作数组参数来处理** |
|  | **D.** | **可变参数对应的实参可以1个，2个，多个，但不能0个，还可以是一个数组** |

|  |  |  |
| --- | --- | --- |
| **8.** | **以下选项中能够正确创建一个数组的是（ AD ）。（选择二项）** | |
|  |  |  |
|  | **A.** | **float []f[] = new float[6][6];** |
|  | **B.** | **float f[][] = new float[][];** |
|  | **C.** | **float [6][]f = new float[6][6];** |
|  | **D.** | **float [][]f = new float[6][];**  **f[0]=new float[2];**  **f[1]=new float[3];**  **….** |

|  |  |  |
| --- | --- | --- |
| **9.** | **下面的数组定义哪些是正确的？（ CD）。（选择二项）** | |
|  |  |  |
|  | **A** | **int a[][] = new int[3,3];** |
|  | **B.** | **int a[3][3] = new int[][];** |
|  | **C.** | **int a[][] = new int[3][3];** |
|  | **D.** | **int []a[] = new int[3][3];** |

1. **判断题**
2. **数组可以声明为任何数据类型，包括任何基本数据类型和引用数据类型。（ T ）**
3. **数组的长度是确定的，数组一旦被创建，它的大小就是不可以改变的。但是其元素类型可以是不同类型，允许出现混合类型。（F）**
4. **数组中的元素的类型必须是相同的，并且数组中的元素是有序的。（T ）**
5. **声明数组并分配空间组的每个元素将会赋予初始值。（F）**
6. **创建数组后，系统会给每个数组元素一个默认值，如double型元素的默认值是0.0。（T ）**
7. **数组的主要优点是按照索引查找某个元素效率高，同时按照元素值查询某个元素效率也很高，但是添加和删除元素需要大量移动元素，效率低下。（T ）**
8. **数组的某个元素被传递给一个方法并被该方法修改，当被调用方法执行完毕时，这个元素中含有修改过的数值。（ F ）**

|  |
| --- |
| ***public static void main(String[] args) {***  ***int[] a = {1,2,3};***  ***test(a[0]);***  ***}***  ***public static void test(int num){***  ***num = 10;***  ***}*** |

1. **Java允许创建不规则数组，即Java多维数组中各行的列数可以不同。（ T ）**

**int a[][]={{1,2,3},**

**{4,5},**

**{7,8, 9}**

**};**

1. **对于数组int[][] t={{1,2,3},{4,5,6}}来说，t.length等于3，t[0].length等于2（T ）**
2. **数组是引用类型，数组也是对象。（ T ）**
3. **简答题**
4. **数组的特点。**

1）数组本身是引用数据类型，而数组中的元素可以是任何数据类型，包括基本数据类型和引用数据类型。

2）创建数组对象会在内存中开辟一整块连续的空间，而数组名中引用的是这块连续空间的首地址。

3）数组的长度一旦确定，就不能修改。

4）可以直接通过下标(或索引)的方式调用指定位置的元素，速度很快。

1. **数组的优缺点**

**数组的主要优点是按照索引查找某个元素效率高，同时按照元素值查询某个元素效率也很高，但是添加和删除元素需要大量移动元素，效率低下。**

1. **冒泡排序的算法。**

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

**for** (**int** j = 0; j <a.length-i-1 ; ++j) {

**if** (a[j + 1] < a[j]) {

int temp = a[j];a[j] = a[j + 1];a[j + 1] = temp;

}

}

}

1. **数组的三种初始化方式是什么?**

**Int [] a=new int[1];**

**Int[] a={1,2,3,4};**

**Int[] a=new int[]{1,2,2}；**

1. **编码题**
2. **数组查找操作：定义一个长度为10 的一维字符串数组，在每一个元素存放一个单词；然后运行时从命令行输入一个单词，程序判断数组是否包含有这个单词，包含这个单词就打印出“Yes”，不包含就打印出“No”。**

**package** homework;

**import** java.util.Scanner;

**public** **class** **Test01** {

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

//数组查找操作：定义一个长度为10 的一维字符串数组，在每一个元素存放一个单词；

//然后运行时从命令行输入一个单词，程序判断数组是否包含有这个单词，

//包含这个单词就打印出“Yes”，不包含就打印出“No”。

**Scanner** sc=**new** Scanner(**System**.***in***);

**boolean** flag=**false**;

**System**.***out***.println("输入一个单词：");

**String** word=sc.next();

**String**[] str=**new** **String**[] {"hello","word","is","people","stduent","are","for","while","dowhile","if"};

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

**if**(word.equals(str[i])) {

flag=**true**;

**break**;

}

}

**if**(flag)

**System**.***out***.println("Yes");

**else**

**System**.***out***.println("No");

}

}

**2.获取数组最大值和最小值操作：利用Java的Math类的random()方法，编写函数得到 0到n之间的随机数，n是参数。并找出产生50个这样的随机数中最大的、最小的数， 并统计其中>=60的有多少个。  
package** homework;

**import** java.util.Scanner;

**public** **class** **Test02** {

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

/\*

\* 获取数组最大值和最小值操作：利用Java的Math类的random()方法，

\* 编写函数得到 0到n之间的随机数，n是参数。

\* 并找出产生50个这样的随机数中最大的、最小的数， 并统计其中>=60的有多少个。

\*/

**Scanner** sc=**new** Scanner(**System**.***in***);

**System**.***out***.println("请输入参数：");

**int** n=sc.nextInt();

**int** count=0;

**double**[] number=**new** **double**[50];

number[0]=**Math**.*random*()\*n;

**double** max=number[0],min=number[0];

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

number[i]=**Math**.*random*()\*n;

}

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

**if**(number[i]>=60)

count++;

**if**(number[i]>max)

max=number[i];

**if**(number[i]<min)

min=number[i];

}

**System**.***out***.println("大于60的有："+count+"max:"+max+"min:"+min);

}

}

**3.数组逆序操作：定义长度为10的数组，将数组元素对调，并输出对调前后的结果。**

**思路：把0索引和arr.length-1的元素交换，把1索引和arr.length-2的元素交换…..**

**只要交换到arr.length/2的时候即可。**

**package** homework;

**public** **class** **Test03** {

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

//数组逆序操作：定义长度为10的数组，将数组元素对调，并输出对调前后的结果。

//思路：把0索引和arr.length-1的元素交换，把1索引和arr.length-2的元素交换…..

// 只要交换到arr.length/2的时候即可。

**int**[] arr=**new** **int**[] {1,2,3,4,5,6,7,8,9,10};

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

**System**.***out***.print(arr[i]);

}

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

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

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

arr[i]=arr[arr.length-(i+1)];

arr[arr.length-(i+1)]=temp;

}

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

**System**.***out***.print(arr[i]);

}

}

}

**4.合并数组操作：现有如下一个数组：   int oldArr[]={1,3,4,5,0,0,6,6,0,5,4,7,6,7,0,5}   要求将以上数组中值为0的项去掉，将不为0的值存入一个新的数组，生成的新数组为：**

**package** homework;

**import** javax.swing.plaf.synth.SynthSpinnerUI;

**public** **class** **Test04** {

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

//合并数组操作：现有如下一个数组：

//int oldArr[]={1,3,4,5,0,0,6,6,0,5,4,7,6,7,0,5}

//要求将以上数组中值为0的项去掉，将不为0的值存入一个新的数组，生成的新数组为：

**int** oldArr[]={1,3,4,5,0,0,6,6,0,5,4,7,6,7,0,5} ;

**int** count=0;

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

**if**(oldArr[i]==0) {

count++;

}

}

**int**[] newArr=**new** **int**[oldArr.length-count] ;

**int** j=0;

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

**if**(oldArr[i]!=0)

newArr[j++]=oldArr[i];

}

**for**(**int** temp:newArr)

**System**.***out***.print(temp+"\t");

}

}

**5.二分法查找操作：使用二分法查找有序数组中元素。找到返回索引，不存在输出-1。**

**分析：二分法查找的前提是数组有序。**

**假如有一组数为3，12，24，36，55，68，75，88要查给定的值24.可设三个变量front，mid，end分别指向数据的上界，中间和下界，mid=（front+end）/2.**

1. **开始令front=0（指向3），end=7（指向88），则mid=3（指向36）。因为mid>x，故应在前半段中查找。**
2. **令新的end=mid-1=2，而front=0不变，则新的mid=1。此时x>mid，故确定应在后半段中查找。**
3. **令新的front=mid+1=2，而end=2不变，则新mid=2，此时a[mid]=x，查找成功。**
4. **如要查找的数不是数列中的数，例如x=25，当第三次判断时，x>a[mid]，按以上规律，令front=mid+1，即front=3，出现front>end的情况，表示查找不成功。**

**package** homework;

**import** java.util.Scanner;

**public** **class** **Test05** {

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

//二分法查找操作：使用二分法查找有序数组中元素。找到返回索引，不存在输出-1。

//分析：二分法查找的前提是数组有序。

//假如有一组数为3，12，24，36，55，68，75，88要查给定的值24.

//可设三个变量front，mid，end分别指向数据的上界，中间和下界，mid=（front+end）/2.

//1)开始令front=0（指向3），end=7（指向88），则mid=3（指向36）。因为mid>x，故应在前半段中查找。

//2)令新的end=mid-1=2，而front=0不变，则新的mid=1。此时x>mid，故确定应在后半段中查找。

//3)令新的front=mid+1=2，而end=2不变，则新mid=2，此时a[mid]=x，查找成功。

//4)如要查找的数不是数列中的数，例如x=25，当第三次判断时，x>a[mid]，按以上规律，

//令front=mid+1，即front=3，出现front>end的情况，表示查找不成功。

**Scanner** sc=**new** Scanner(**System**.***in***);

**int**[] arr=**new** **int**[] {3,12,24,36,55,68,75,88};

**System**.***out***.println("请输入查找的数：");

**int** number=sc.nextInt();

**int** front=0,end=arr.length-1,mid=(front+end)/2;

**boolean** flag=**false**;

**while**(arr[front]<=arr[end]) {

**if**(number==arr[mid]) {

flag = **true**;

**break**;

}

**else** **if**(number>arr[mid]) {

front = mid+1;

mid = (front+end)/2;

}

**else** {

end = mid-1;

mid = (front+end)/2;

}

}

**if**(flag) {

**System**.***out***.println("查找成功！");

**System**.***out***.println("你要查找的数的下标是：" + mid);

}

**else**

**System**.***out***.println("查找失败！");

}

}

**6.二维数组遍历求和操作：用二重循环求出二维数组b所有元素的和：  
 int[][] b={{11},{21,22},{31,32,33}}**

**package** homework;

**public** **class** **Test06** {

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

//二维数组遍历求和操作：用二重循环求出二维数组b所有元素的和：

//int[][] b={{11},{21,22},{31,32,33}}

**int**[][] b={{11},{21,22},{31,32,33}};

**int** sum=0;

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

**for** (**int** j = 0; j < b[i].length; j++) {

sum+=b[i][j];

}

}

**System**.***out***.println("sum:"+sum);

}

}

1. **可选题**
2. **生成一百个随机数，放入数组，然后排序输出。**

**package** homework;

**import** java.util.Random;

**public** **class** **Test07** {

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

//生成一百个随机数，放入数组，然后排序输出。

**int**[] arr=**new** **int**[102];

**for**(**int** i=0;i<=100;i++) {

**Random** rd = **new** Random();

**int** number=rd.nextInt(100);

arr[i]=number;

}

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

**for** (**int** j = 0; j < arr.length-1; j++) {

**if**(arr[j]>arr[j+1])

{

**int** temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

}

}

}

**for** (**int** j = 0; j < arr.length; j++) {

**System**.***out***.println(arr[j]);

}

}

}

1. **题目：输入某年某月某日，判断这一天是这一年的第几天？**

**分析：以3月5日为例，先把前两个月的加起来，然后再加上5天即本年第几天，特殊情况，闰年且输入月份大于3需考虑多加一天。可定义数组存储1-12月各月天数。**

1. **使用二分法查找有序数组中元素。找到返回索引，不存在输出-1。使用递归实现**
2. **数组A：1，7，9，11，13，15，17，19；数组b：2，4，6，8，10**

**两个数组合并为数组c，按升序排列。 要求：使用Arrays类的方法快速实现。**