数组作业

一、 填空题

1. 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。
2. 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。
3. __声明数组__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。
4. 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__。
5. 在 Java 中有二维数组 int [ ] [ ] array={{1,2,3},{4,5}} ，可以使用__array[0].length__得到二维数组中第二维中第一个数组的长度。
6. 数组元素下标(或索引)的范围是__[0, arr.length)__。

二、 选择题

1. **在 Java 中，以下程序段能正确为数组赋值的是（ AD）。（选择二项）**

A int a[]={1,2,3,4};
B. int b[4]={1,2,3,4};
C. int c[];c={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
B. 整型变量
C. 整型表达式
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（ CD）。（选择二项）**

A int t[10]=new int[ ];
B. char [ ]a=”hello”;
C. String [ ] s=new String [10];
D. double[ ] d [ ]=new double [4][ ];

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));
    }
}
```

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（ AC）。（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

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（ D ）。（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][];

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];

三、 判断题

1. 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 ）
2. 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 ）
3. 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）
4. 声明数组并分配空间组的每个元素将会赋予初始值。（ T ）
5. 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 ）
6. 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 ）
7. 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 ）
8. 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）
9. 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（ F ）
10. 数组是引用类型，数组也是对象。（ T ）

四、 简答题

1. 数组的特点。
2. 数组的优缺点
3. 冒泡排序的算法。
4. 数组的三种初始化方式是什么?

五、 编码题

1. 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”。
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。
   提示：使用 int num=(int)(n*Math.random());获取随机数
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。

1.　数组的特点。
其长度是确定的。数组一旦被被创建，它的大小就是不可以改变的。其元素必须是相同类型，不允许出现混合类型。数组中的元素可以是任何数据类型，包括基本类型和引用类型数组变量属引用类型，数组也可以看成是对象，数组中的每个元素相当于该对象的成员变量。
2. 数组的优缺点
使用方便，查询效率比链表高，内存唯一连续的区域缺点：大小固定，不适合动态存储，不方便动态添加。
3. 冒泡排序的算法。
冒泡排序是一种排序算法。它重复地走访过要排序的数列，一次比较两个元素，如果他们的顺序错误就把他们交换过来。
4. 数组的三种初始化方式是什么?
静态数组，动态数组

1.

```java
public class ArraySearching {
    1个用法  新 *
    public static void searching(String [] strings){
        System.out.println("请输入一个单词: ");
        Scanner scanner = new Scanner(System.in);
        String word = scanner.nextLine();
        boolean flag = false;
        for (String string : strings) {
            if (Objects.equals(word, string)) {
                System.out.println("Yes");
                flag = true;
            }
        }
        if (!flag){
            System.out.println("No");
        }
    }
    新 *
    public static void main(String[] args) {
        String [] strings = new String [] {"pig", "dog", "duck", "apple", "banana", "goose", "fabulous", "bird", "grape", "strawberry"};
        searching(strings);
    }
}
```

2.

```java
    public static void getMaxAndMinAndStatisticalFiguresFromRandomArray(int n){
        Random r = new Random();
        int count = 0;
        for (int i = 0; i < ar.length; i++) {
            int rr = r.nextInt(n);
            ar[i] = rr;
            if (rr >= 60){
                count++;
            }
        }
        System.out.println("大于等于60的数字有: " + count + "个");
        Arrays.sort(ar);
        System.out.println("最小值: " + ar[0]);
        System.out.println("最大值: " + ar[ar.length-1]);
    }

    新 *
    public static void main(String[] args) {
        getMaxAndMinAndStatisticalFiguresFromRandomArray( n: 100);

    }
}
```

```
运行:  GetMaxAndMinFromArray
/Library/Java/JavaVirtualMachines/jdk-12.0.2.jdk/Contents/Home/bin/java -javaagent:/Users/a
Support/JetBrains/Toolbox/apps/IDEA-U/ch-0/223.7571.182/IntelliJ IDEA.app/Contents/lib/ide
Support/JetBrains/Toolbox/apps/IDEA-U/ch-0/223.7571.182/IntelliJ IDEA.app/Contents/bin -Df
/Users/apple/JavaLearning/out/production/JavaLearning Homeworks.HomeworkDay04.GetMaxAndMin
大于等于60的数字有: 15个
最小值: 6
最大值: 95
```

3.

```java
    public static void reverseArray(int[] array){
        for (int i = 0; i < array.length / 2; i++) {
            int temp = array[i];
            array[i] = array[array.length - i - 1];
            array[array.length - i - 1] = temp;
        }
    }
    新 *
    public static void main(String[] args) {
        int [] arr = randomArray( size: 10);
        for (int i : arr) {
            System.out.print(i + "\t");
        }
        System.out.println();
        reverseArray(arr);
        for (int i : arr) {
            System.out.print(i + "\t");
        }
    }
}
```

```
运行:  ReverseArray
/Library/Java/JavaVirtualMachines/jdk-12.0.2.jdk/Contents/
Support/JetBrains/Toolbox/apps/IDEA-U/ch-0/223.7571.182/I
Support/JetBrains/Toolbox/apps/IDEA-U/ch-0/223.7571.182/I
/Users/apple/JavaLearning/out/production/JavaLearning Hom
576 686 488 32  469 842 845 109 848 750
750 848 109 845 842 469 32  488 686 576
进程已结束,退出代码0
```

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。

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： int newArr [] ={1,3,4,5,6,6,5,4,7,6,7,5}
思路： 确定出不为0的个数，这样可以开辟新数组；从旧的数组之中，取出内容，并将其赋给新开辟的数组。
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。
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}}

六、 可选题

1. 生成一百个随机数，放入数组，然后排序输出。
2. 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。
3. 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
4. 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。

4.

```java
public class CombineArray {
    新 *
    public static void main(String[] args) {
        int []oldArr = {
                1, 3, 4, 5, 0, 0, 6, 6, 0, 5, 4, 7, 6, 7, 0, 5
        };
        int adjustingLength = 0;


        for (int i : oldArr) {
            if (i != 0){
                adjustingLength++;
            }
        }
        int [] newArr = new int[adjustingLength];
        int j = 0;
        for (int i = 0; i < oldArr.length; i++) {
            if (oldArr[i] != 0){
                newArr[j] = oldArr[i];
                j++;
            }
        }
        for (int i : newArr) {
            System.out.println(i);
        }
    }
}
```

5.

```java
    public static int binarySearch(int[] arr, int n){
        int front = 0;
        int end = arr.length - 1;
        int mid = (front + end) / 2;
        for (int i = 0; i < arr.length / 2; i++) {
            if (arr[mid] > n){
                end = mid;
                mid = (front + end) / 2;
            }else if (arr[mid] < n){
                front = mid;
                mid = (front + end + 1) / 2;
            }
            if (arr[mid] == n){
                return mid;
            }
        }
        return -1;
    }
    新 *
    public static void main(String[] args) {
        int[] arr = {
            3, 12, 24, 36, 55, 68, 75, 88
        };
        System.out.println(binarySearch(arr, n: 88));

    }
```

6.

```java
public class DoubleLoop {
    新 *
    public static void main(String[] args) {
        int [][]b = {
                {11},
                {21, 22},
                {31, 32, 33}
        };
        int sum = 0;
        for (int[] ints : b) {
            for (int anInt : ints) {
                sum+=anInt;
            }
        }
        System.out.println(sum);
    }
}
```