04-数组-作业

1.数组初始化考核

需求描述:

使用三种不同方式创建数组并初始化，数组包含a、b,c三个元素

答案:

```
public class Test01_init {
    public static void main(String[] args) {
        //第一种

        //第二种

        //第三种
    }
}
```

2.数组遍历考核

需求描述:

1. 一维数组遍历

   有数组`String[] arr = new String[]{"hello","world","!"};`请使用2种不同的方式进行遍历

答案:

```java
public class Test02_traversal {
    public static void main(String[] args) {
        String[] arr = new String[]{"hello","world","!"};
        //第一种方式：普通for循环

        //第二种方式：增强for循环

    }
}
```

2. 二维数组遍历
    - 二维数组的长度为3，二维数组中每个一维数组的长度也为3
    - 二维数组每个位置上的数组的数据都是通过键盘录入
    - 录入成功后，将二维数组按以下格式打印输出

```
[3, 6, 2]
[7, 1, 9]
[4, 3, 0]
```

答案：

```java
public class Test02_doubleArray {
    public static void main(String[] args) {
        //创建二维数组
        int arr[][] = new int[3][3];

        //创建扫描器对象

        // 键盘录入第一个元素数组的数据

        //打印二维数组

    }
}
```

3.获取次大值

需求描述:

补全方法，实现求一个int[]数组的次大值

思路：根据数组的第一个和第二个元素的大小关系确定最大值和次大值从第三个元素开始依次比较和最大值、次大值的大小关系，确定是否需要改变最大值和次大值的值:

```java
public static int getSecondValue(int[] arr){
    return 0;
}
```

答案:

```java
public class Test03_SecondValue {
    public static int getSecondValue(int[] arr) {
        //默认使用第一个值作为最大值
        int max = (arr[0] > arr[1]) ? arr[0] : arr[1];
        //默认使用第二个值作为第二大值
```

```
        int sec = (arr[0] > arr[1]) ? arr[1] : arr[0];


        return sec;
    }

    public static void main(String[] args) {
        int[] arr = {12,4,56,45,67,34,52,28,65};
        int num = getSecondValue(arr);
        System.out.println("次大值: " + num);
    }
}
```

4.增强for循环考核

需求描述:

分析以下代码运行结果

```
public class Test04_foreach {
    public static void main(String[] args) {
        int[] arr1 = {1,2,3};
        for (int num : arr1){
            num = 4;
        }
        System.out.println("输出: " + arr1[0]);
    }
}
```

答案:

5.数组内存考核

需求描述：

分析以下代码的运行结果，并画出代码对应的内存图

```
public class Test05_memory {
    public static void main(String[] args) {
        int[] arr1 = {1,2,3};
        int[] arr2 = arr1;
        arr2[1] = 0;
        System.out.println("arr1: " + Arrays.toString(arr1));
        System.out.println("arr2: " + Arrays.toString(arr2));
    }
}
```

答案：

6.生成随机数

需求描述：

产生10个随机数，范围在[1，100]，使用**java.lang.Math**类实现

答案：

```java
public class Test6_random {
    public static void main(String[] args) {
        // 定义存放生成的随机数的数组
        int[] arr = new int[10];

        // 循环遍历数组，为每个索引位置生成随机数，并输出生成的随机数


        //打印数组
        System.out.println("数组元素："+Arrays.toString(arr));
    }
}
```

7.生成验证码

需求描述：

随机产生一个长度为4位的验证码，包含大小写字母以及数字。

答案：

```java
public class Test7_validateCode {
    public static void main(String[] args) {
        //创建存放验证码的数组
        char[] validateCode = new char[4];
        //创建存放大小写字母以及数字数组
        char[] data = new char[62];

        //给data输出添加元素值，提示：通过循环实现


        //循环遍历，为存放验证码的数组每个位置赋值，并输出验证码

```

```
13
14          //输出验证码
15          System.out.println("验证码内容："+
    Arrays.toString(validateCode));
16      }
17  }
```

8.程序输入

需求描述：

程序运行，利用随机数生成生成一个长度为20的正整型数组。然后等待用户输入要查找的数字。

1. 如果数组中包含用户输入数字，提示用户数字所在数组下标位置
2. 如果数组中不包含用户输入数字，提示用户数据不存在
3. 用户可以一直进行输入数字，如果用户输入-1，程序停止运行

答案：

```
1   public class Test8_scanner {
2       public static void main(String[] args) {
3           //定义长度为20的数组，存放随机数
4           int[] arr = new int[20];
5
6           //生成随机数，对数组元素赋值
7
8
9           //程序一直运行
10          //创建Scanner对象，用来在控制台输入数据
11          Scanner scanner = new Scanner(System.in);
12          while (true){
13              //自行分析实现过程，并补全代码
```

```
14
15
16              }
17          }
18      }
```

9.统计次数

需求描述：

1. 产生100个[1,6]之间的随机数，并统计每个数字出现的概率
2. 可以创建2个数组分别保存随机数信息和统计产生次数信息
3. 例如：随机数生成了10个1，20个2，30个3，5个4，10个5，25个6，对应的统计数组内容[10,20,30,5,10,25]，即统计数组的每一个位置上存放的是对应下标出现的次数

答案：

```
1   public class Test09_count {
2       public static void main(String[] args) {
3           //定义一个长度为100的数组存放随机生成的数据
4           int[] randomArr = new int[100];
5           //定义统计数组
6           int[] countArr = new int[6];
7
8           //初始化数组randomArr
9
10
11          //统计每个数字出现的概率，最终填入countArr中
12
13
14          //输出每个数字出现的概率
15
16      }
```

```
17    }
```

10.排序算法考核

需求描述：

分别使用冒泡排序、选择排序、插入排序三种不同算法列出数组arr每一轮排序后的结果，并编写代码实现对数组arr排序

```
int[] arr = {56,34,45,12,67,52,4}
```

答案：

冒泡排序

每一轮冒泡排序的结果：

1. **[34, 45, 12, 56, 52, 4, 67]**
2. **[34, 12, 45, 52, 4, 56, 67]**
3. **[12, 34, 45, 4, 52, 56, 67]**
4. **[12, 34, 4, 45, 52, 56, 67]**
5. **[12, 4, 34, 45, 52, 56, 67]**
6. **[4, 12, 34, 45, 52, 56, 67]**

```java
public class Test10_sort {
    public static void bubbleSort(int[] arr) {

    }

    public static void main(String[] args) {
        int[] arr = {56,34,45,12,67,52,4};
        Test10_sort.bubbleSort(arr);
    }
}
```

选择排序

每一轮选择排序的结果：

1. **[4, 34, 45, 12, 67, 52, 56]**
2. **[4, 12, 45, 34, 67, 52, 56]**
3. **[4, 12, 34, 45, 67, 52, 56]**
4. **[4, 12, 34, 45, 67, 52, 56]**
5. **[4, 12, 34, 45, 52, 67, 56]**
6. **[4, 12, 34, 45, 52, 56, 67]**

```java
public class Test10_sort {
    public static void selectionSort(int[] arr) {
        for (int i = 0; i < arr.length - 1; i++) {

        }
        System.out.println("选择排序："+Arrays.toString(arr));
    }

    public static void main(String[] args) {
        int[] arr = {56,34,45,12,67,52,4};
        Test10_sort.bubbleSort(arr);
        Test10_sort.selectionSort(arr);
```

```
    }
}
```

插入排序

每一轮插入排序的结果：

1. **[34, 56, 45, 12, 67, 52, 4]**
2. **[34, 45, 56, 12, 67, 52, 4]**
3. **[12, 34, 45, 56, 67, 52, 4]**
4. **[12, 34, 45, 56, 67, 52, 4]**
5. **[12, 34, 45, 52, 56, 67, 4]**
6. **[4, 12, 34, 45, 52, 56, 67]**

```java
public class Test10_sort {
    public static void insertSort(int[] arr) {

    }

    public static void main(String[] args) {
        int[] arr = {56,34,45,12,67,52,4};
        Test10_sort.insertSort(arr);
    }
}
```