中北大学软件学院

**实验报告**

|  |  |
| --- | --- |
| 专 业： | 软件工程 |
| 课程名称： | 面向对象程序设计 |
| 班 级： | 22130403 |
| 学 号： | 2213040340 |
| 姓 名： | 张恒瑜 |
| 辅导教师： | 李华玲 |

2021年9月制

成绩：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 实验时间 | 2023年 9 月28 日 17 时至 21时 | 学时数 | 2学时 | |
| 1.实验名称  数组和循环嵌套 | | | | |
| 2.实验目的  （1）掌握数组的定义、初始化、访问和使用；  （2）学会循环嵌套语句在数组中的使用；  （3）学会使用数组解决实际问题；  （4）学会数组在面向对象程序设计中的运用； | | | | |
| 3.训练知识点集群  定义数组，初始化数组与数组赋值，使用数组，数据类型转换，while循环块，for循环块 | | | | |
| 4.实验内容  （1）根据用户输入的10个数值，自己写排序算法进行升序排列并输出排序后的结果。  （2）向控制台输出10 \* 10的数值表格。输出效果如下：  （3）课程融合之“N元组”：  设计实现抽象数据类型“N元组 （Tuple）” 。每个N元组由任意N个实数的序列构成，基本操作包括：创建一个N元组，取N元组的任意一个分量，置N元组的任意一个分量，求N元组的最大分量、最小分量，显示N元组等。  要求：用类封装需要定义的数据类型Tuple，完成所有基本操作的Java语言实现与调用，并写测试程序。  （4）小组自主创作实验：围绕实验目的，题目自拟。 | | | | |
| 1. 写出实验内容（1）所采用的排序算法的思路   实验一所使用的是冒泡排序  思路：冒泡排序是有两层循环，第一层是交换的次数，第二层是比较的次数，次数是比较难想象的，但是可以通过列举数子13245等等少量数字来达到目的。  冒泡排序是从第一个数开始，一个个比较，12相比，若1>2则交换，第一次会在末尾得到数组中最大的数，第二次会将数组中第二大的数排在倒数第二个的位置，以此类推，得到升序或者降序的排列是十分容易的 | | | | |
| 1. 画出实验内容（3）对应的类图  |  | | --- | | **NTuple** | | Elem[];float | | Initntuple();void  SearElem();float  ChangeElem();void  Maxntuple();float  Minntuple();float  Showntuple();void | | | | | |
| 7.实验源代码      //根据用户输入的10个数值，自己写排序算法进行升序排列并输出排序后的结果  //升序排列，这里使用冒泡排序进行  import java.util.Scanner;  class methodSort {  int[] arr;  public void bubbleSort(int[] arr) {  boolean flag = true;// 做一个标志flag并把它定义为true  for(int i = 0; i < arr.length - 1; i++){  for(int j = 0 ; j < arr.length - i - 1; j++){  if(arr[j] > arr[j + 1]) {  flag = false; //若经过上述判断进入循环则将标志物定义为flase 继续进行if条件  int temp = arr[j];  arr[j] = arr[j + 1];  arr[j + 1] = temp;  }  }  if(flag) {//若没有进行if条件的交换参与，则直接按照原来的true进行break  break;  }  }  }  }  public class TestBubbleSort {  public static void main(String[] args) {  methodSort method = new methodSort();  Scanner sc = new Scanner(System.in);  int[] ints = new int[10];  System.out.println("请用户输入10个数：");  for(int i = 0; i < ints.length; i++) {  ints[i] = sc.nextInt();  }  method.bubbleSort(ints);  for(int anInt:ints) {  System.out.println(anInt + " ");  }  }  }      class Form0 {  public void crOptimize(int row,int colume) {  for(int i = 0; i < row; i++ ) {  for(int j = 0; j < colume; j++) {  System.out.print("("+i+","+j+")");  }  System.out.println();  }  }  }  class FormOptimize {  public static void main(String[] arge) {  Form0 form = new Form0();  form.crOptimize(10,10);  }  }      //设计实现抽象数据类型“N元组 （Tuple）” 。每个N元组由任意N个实数的序列构成，基本操作包括：创建一个N元组，取N元组的任意一个分量，置N元组的任意一个分量，求N元组的最大分量、最小分量，显示N元组等。  import java.util.Scanner;  class NTuple {  float[] elem;  public void Initntuple() {  Scanner sc = new Scanner(System.in);  System.out.println("请输入你想创建n元组的元素个数：");  int num = sc.nextInt();  elem = new float[num];  for(int i = 0; i < elem.length; i++) {  int j = i + 1;  System.out.println("请输入第" + j + "个数：");  elem[i] = sc.nextFloat();  System.out.println(" ");  }  }  public void SearchElem() {  Scanner sc1 = new Scanner(System.in);  System.out.println("请输入你要查数组内第几个数： ");  int num = sc1.nextInt();  System.out.println(elem[num - 1]);  }  public void ChangeElem() {  Scanner sc2 = new Scanner(System.in);  System.out.println("请输入你要改变数组内第几个数：");  int num = sc2.nextInt();  System.out.println("这个数你要改变为：");  elem[num - 1] = sc2.nextFloat();  System.out.println("数组内第" + num + "个数已经改为" + elem[num - 1]);  }  public float Maxntuple() {  float max = elem[0];  int maxindex = 0;//用排序后再找也不是不行，但是聪明绝顶的张某立马想到了选择排序，而这里只用到了选择排序的一部分，也是最重要的一部分，就是利用数组下标进行查找  for(int i = 0; i < elem.length - 1; i++) {  if(elem[i] < elem[i + 1]) {  maxindex = i + 1;  }  }  max = elem[maxindex];  return max;  }  public float Minntuple() {  float min = elem[0];  int minindex = 0;  for(int i = 0; i < elem.length - 1; i++) {  if(elem[i] > elem[i + 1]) {  minindex = i + 1;//可以想象成传送锚点，在最小值设置一个，然后直接传送！  }  }  min = elem[minindex];  return min;  }  public void Showntuple() {  for(float anFloat:elem) {  System.out.println("[" + anFloat + "]");  }  }  }  public class TestNtuple {  public static void main(String[] args) {  Scanner sc3 = new Scanner(System.in);  NTuple tuple = new NTuple();  while(true) {  System.out.println("\*\*\*\*\*\*请输入你想进行的操作\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*1.创建一个N元组\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*2.查找N元组的数\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*3.修改N元组的数\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*4.找N元组最大值\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*5.找N元组最小值\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*6.列目前的N元组\*\*\*\*\*\*\*\*");  System.out.println("\*\*\*\*\*\*\*\*\*\*\*7.退出程序\*\*\*\*\*\*\*\*\*\*\*");  int choose = sc3.nextInt();  switch(choose) {  case 1 -> {  tuple.Initntuple();  break;  }  case 2 -> {  tuple.SearchElem();  break;  }  case 3 -> {  tuple.ChangeElem();  break;  }  case 4 -> {  System.out.println("N元组中最大的数是：" + tuple.Maxntuple());  break;  }  case 5 -> {  System.out.println("N元组中最小的数是：" + tuple.Minntuple());  break;  }  case 6 -> {  tuple.Showntuple();  break;  }  case 7 -> {  return;  }  default -> {  System.out.println("输入错误请重新输入！");  break;  }  }  }  }  }  8.实验运行结果截图及心得体会  （1）  屏幕截图 2023-09-28 192022  （2）  屏幕截图 2023-09-29 230456  （3）  屏幕截图 2023-09-28 234143  屏幕截图 2023-09-28 234158  **心得体会：和三元组多的只是一个数组和排序方法，大同小异，虽然思路很到位，但还是码了很长时间，希望能够更快！再接再厉！** | | | |
|  | | | |