Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский

политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчет по лабораторной работе № 8

тема «Одномерные массивы и матрицы в Java»

по дисциплине «Информатика»

Выполнил: студент группы ИСТ-22-1б Язов М.М.

Проверил: Нетбай Георгий Владимирович

Пермь, 2023

**Задание 1**

* 1. ***Постановка задачи***

Ввести массив, состоящий из N элементов целого типа. Массив является элементами целого числа в троичной системе счисления. Проверить правильность введения элементов числа, т.е. в массиве должны быть только 0, 1 и 2. Если число введено верно, то перевести его в десятичную систему счисления и поэлементно записать число в массив. Пример:

|  |  |
| --- | --- |
| В 3-ой | В 10-ой |
| |  |  |  |  | | --- | --- | --- | --- | | 1 | 2 | 0 | 1 | | |  |  | | --- | --- | | 4 | 6 | |

***1.2. Решение задачи, код программы***

import *java.util.ArrayList*;  
import *java.util.Scanner*;  
  
public class *exc8\_1* {  
 public static void main(*String*[] *args*) {  
 *Scanner* input = new Scanner(*System*.in);  
 *ArrayList*<*Integer*> arr = new ArrayList<>();  
 int num, len, smt = 0, ans = 0;  
 *System*.out.println("Введите кол-во знаков числа в 3-ой СС: ");  
 len = input.nextInt();  
  
 while (smt != len) {  
 *System*.out.println("Введите число в 3-ой СС (т.е. числа: 0, 1, 2): ");  
 num = input.nextInt();  
 if (num >= 0 && num <= 2) {  
 arr.add(num);  
 smt++;  
 } else {  
 *System*.out.println("Числа которые могут быть (0, 1, 2)");  
 }  
 }  
  
 *System*.out.print("Массив: ");  
 for (int i = 0; i < arr.size(); i++) {  
 *System*.out.print(arr.get(i) + " ");  
 }  
 *System*.out.println();  
  
 for (int i = 0; i < arr.size(); i++) {  
 int pow = len - 1 - i;  
 int digit = arr.get(i);  
 if (digit < 0 || digit > 2) {  
 *System*.out.println("Введено некорректное значение.");  
 return;  
 }  
 ans += digit \* *Math*.*pow*(3, pow);  
 }  
  
 *System*.out.println("Результат перевода: " + ans);  
 }  
}

***1.3. Тестирование работы программы с проверкой***

В таблице 1 представлено тестирование работы программы с проверкой решения задачи на языке Java.

Таблица 1

Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| № п.п. | Решение Java | Проверка |
| 1 |  |  |
| 2 |  |  |

**Задание 2**

***2.1. Постановка задачи***

Ввести массив, состоящий из N элементов. Упорядочить массив так, чтобы все отрицательные числа были расположены вначале по возрастанию, а все положительные – в конце по убыванию, между ними нулевые числа.

***2.2. Решение задачи, код программы***

import *java.util.ArrayList*;  
import *java.util.Arrays*;  
  
public class *exc8\_2\_1* {  
  
 public static void main(*String*[] *args*) {  
 *ArrayList*<*Integer*> positive = new ArrayList<>();  
 *ArrayList*<*Integer*> negative = new ArrayList<>();  
 int[] arr = new int[200];  
 for (int i = 0; i < arr.length; i++) {  
 arr[i] = (int) (*Math*.*random*() \* 200) - 100;  
 }  
 *System*.out.println(*Arrays*.*toString*(arr));  
  
 for (int j : arr) {  
 if (j >= 0)  
 positive.add(j);  
 else  
 negative.add(j);  
 }  
  
 *Integer*[] smt = new Integer[positive.size()];  
 for (int i = 0; i < smt.length; i++) smt[i] = positive.get(i);  
 *Arrays*.*sort*(smt);  
 for (int i = 0; i < smt.length; i++) positive.set(i, smt[i]);  
  
 *System*.out.println(positive);  
  
 smt = new Integer[negative.size()];  
  
 for (int i = 0; i < smt.length; i++) smt[i] = negative.get(i);  
 *Arrays*.*sort*(smt);  
 for (int i = negative.size() - 1; i > -1; i--) negative.set(negative.size() - 1 - i, smt[i]);  
  
 *System*.out.println(negative);  
  
 negative.addAll(positive);  
  
 *System*.out.println(negative);  
 }  
}

***2.3. Тестирование работы программы с проверкой***

Далее в таблице 2 представлено тестирование работы программы с проверкой решения задач на языке Java

Таблица 2

Тестирование работы программы

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |
| 2 |  |

**Задание 3**

***3.1. Постановка задачи***

В одномерном массиве с четным количеством элементов (2N) находятся координаты N точек плоскости. Они располагаются в следующем порядке: x1, y1, х2, y2, x3, y3, и т.д. Определить номера точек, которые могут являться вершинами прямоугольника.

***3.2. Решение задачи, код программы***

import *java.util.Scanner*;  
  
public class *exc8\_3* {  
 public static void main(*String*[] *args*) {  
 *Scanner* input = new Scanner(*System*.in);  
 *System*.out.println("введите количество точек");  
 int N = input.nextInt();  
 int[] coordinates = new int[2 \* N];  
 for (int i = 0; i < 2 \* N; i++) {  
 if (i % 2 == 0) {  
 *System*.out.println("Введите x точки " + ((i / 2) + 1));  
 } else {  
 *System*.out.println("Введите y точки " + ((i / 2) + 1));  
 }  
 coordinates[i] = input.nextInt();  
 }  
  
  
 for (int i = 0; i < (2 \* N - 4); i += 2) {  
 for (int j = i + 2; j < (2 \* N - 3); j += 2) {  
 for (int k = j + 2; k < (2 \* N - 2); k += 2) {  
 for (int l = k + 2; l < (2 \* N - 1); l += 2) {  
 if (((coordinates[i] == coordinates[j]) & (coordinates[k] == coordinates[l]) & (coordinates[i + 1] == coordinates[l + 1]) & (coordinates[j + 1] == coordinates[k + 1])) ||  
 ((coordinates[i] == coordinates[l]) & (coordinates[k] == coordinates[l]) & (coordinates[i + 1] == coordinates[j + 1]) & (coordinates[l + 1] == coordinates[k + 1])) ||  
 ((coordinates[i] == coordinates[l]) & (coordinates[k] == coordinates[j]) & (coordinates[i + 1] == coordinates[k + 1]) & (coordinates[j + 1] == coordinates[l + 1]))) {  
 *System*.out.println("Точки с координатами " + "(" + coordinates[i] + ";" + coordinates[i + 1] + "), (" + coordinates[j] + ";" + coordinates[j + 1] + "), ("  
 + coordinates[k] + ";" + coordinates[k + 1] + "), (" + coordinates[l] + ";" + coordinates[l + 1] + ") являются вершинами прямоугольника");  
 }  
 }  
 }  
 }  
 }  
 }  
}

***3.3. Тестирование работы программы с проверкой***

Далее в таблице 3 представлено тестирование работы программы с проверкой решения задач на языке Java.

Таблица 3

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |
| 2 |  |

**Задание 4**

***4.1. Постановка задачи***

Алгоритм быстрой сортировки. Отличительной особенностью быстрой сортировки является операция разбиения массива на две части относительно опорного элемента. Пусть крайний левый элемент опорный pivot. Установим указатель left на следующий за ним элемент; right – на последний. Алгоритм должен определить правильное положение элемента pivot. Указатель left движется вправо, пока элементы, на которые он показывает, остаются меньше опорного. Указатель right движется влево, пока элементы, на которые он показывает, остаются больше опорного. Указатель left перемещается до тех пор, пока не покажет элемент больше pivot; right движется, пока не покажет элемент меньше pivot. Элементы left и right меняются местами. И движением возобновляется снова. Процесс продолжается до тех пор, пока right не окажется слева от left. Тем самым будет определено правильное место опорного элемента. Осуществляется перестановка опорного элемента с элементом, на который указывает right. Опорный элемент находится в нужном месте: элементы слева от него имеют меньшие значения; справа – большие. Алгоритм рекурсивно вызывается для сортировки подмассивов слева от разрешающего и справа от него. Реализовать алгоритм быстрой сортировки. Выполнить сравнение быстрой сортировки с сортировкой «пузырек», рассмотрев количество итераций, а так же промежуточные результаты сортировки.

***4.2. Решение задачи, код программы***

import *java.util.Arrays*;  
  
public class *exc8\_4* {  
  
 public static void quickSort(int[] *array*, int *low*, int *high*) {  
 if (*array*.length == 0)  
 return;*//завершить выполнение, если длина массива равна 0* if (*low* >= *high*)  
 return;*//завершить выполнение если уже нечего делить  
  
 // выбрать опорный элемент* int middle = *low* + (*high* - *low*) / 2;  
 int opora = *array*[middle];  
  
 *// разделить на подмассивы, который больше и меньше опорного элемента* int i = *low*, j = *high*;  
 while (i <= j) {  
 while (*array*[i] < opora) {  
 i++;  
 }  
  
 while (*array*[j] > opora) {  
 j--;  
 }  
  
 if (i <= j) {*//меняем местами* int temp = *array*[i];  
 *array*[i] = *array*[j];  
 *array*[j] = temp;  
 i++;  
 j--;  
 }  
 }  
  
 *// вызов рекурсии для сортировки левой и правой части* if (*low* < j)  
 *quickSort*(*array*, *low*, j);  
  
 if (*high* > i)  
 *quickSort*(*array*, i, *high*);  
 }  
  
 public static void main(*String*[] *args*) {  
 int[] x = new int[10];  
 for (int i = 0; i < x.length; i++) {  
 x[i] = (int) (*Math*.*random*() \* (20 - (-5))) - 5;  
 }  
 *System*.out.println("Было");  
 *System*.out.println(*Arrays*.*toString*(x));  
  
 int low = 0;  
 int high = x.length - 1;  
  
 *quickSort*(x, low, high);  
 *System*.out.println("Стало");  
 *System*.out.println(*Arrays*.*toString*(x));  
 }  
}

***4.3. Тестирование работы программы с проверкой***

Далее в таблице 4 представлено тестирование работы программы с проверкой решения задач на языке Java.

Таблица 4

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |
| 2 |  |

**Задание 7**

***5.1. Постановка задачи***

Заполнить двумерный массив n×n по образцу. Образец заполнения и пример:

|  |  |
| --- | --- |
| Образец заполнения массива | Пример |
|  |  |

***5.2. Решение задачи, код программы***

import *java.util.Scanner*;  
  
public class *exc8\_7* {  
 public static void main(*String*[] *args*) {  
 *Scanner* input = new Scanner(*System*.in);  
 *System*.out.print("Введите размерность матрицы NxN: ");  
 int n = input.nextInt();  
  
 int[][] matrix = new int[n][n];  
  
 int value = 1;  
  
 for (int j = matrix.length; j >= 0; j--) {  
  
 for (int i = matrix.length - j; i < j; i++) matrix[matrix.length - j][i] = value++;  
  
 for (int i = matrix.length - j + 1; i < j; i++) matrix[i][j - 1] = value++;  
  
 for (int i = matrix.length - j + 1; i < j; i++) matrix[i][matrix.length - j] = value++;  
  
 for (int i = matrix.length - j + 2; i < j; i++) matrix[j - 1][i - 1] = value++;  
 }  
  
 for (int i = 0; i < matrix.length; i++) {  
 for (int j = 0; j < matrix[i].length; j++) {  
 *System*.out.print(*String*.*format*("%3.0f", (double) matrix[i][j]) + " ");  
 }  
 *System*.out.println();  
 }  
 }  
}

***5.3. Тестирование работы программы с проверкой***

Далее в таблице 5 представлено тестирование работы программы с проверкой решения задач на языке Java.

Таблица 5

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |
| 2 |  |

**Задание 5**

***6.1. Постановка задачи***

В заданной матрице поменять местами две строки: строку, содержащую максимальный элемент матрицы, и строку, содержащую минимальный элемент матрицы. Если номера таких строк совпали, то выполнить генерацию двумерного массива еще раз и т.д. пока строчки с минимальным и максимальным элементами не будут совпадать.

***6.2. Решение задачи, код программы***

import *java.util.Arrays*;  
import *java.util.Scanner*;  
  
public class *exc8\_5* {  
  
 public static void main(*String*[] *args*) {  
 *Scanner* input = new Scanner(*System*.in);  
  
 *System*.out.println("Сколько строк в матрице? ");  
 int line = input.nextInt();  
 *System*.out.println("Сколько столбцов в матрице? ");  
 int column = input.nextInt();  
  
 int[][] arr = new int[line][column];  
 int[] gawno = {*Integer*.MAX\_VALUE, *Integer*.MIN\_VALUE, -1, -1};  
  
 while (true) {  
 for (int i = 0; i < line; i++) {  
 for (int j = 0; j < column; j++) {  
 arr[i][j] = (int) (*Math*.*random*() \* (100 - 10)) + 10;  
 }  
 }  
 for (int i = 0; i < line; i++) *System*.out.println(*Arrays*.*toString*(arr[i]));  
  
 for (int i = 0; i < line; i++) {  
 for (int j = 0; j < column; j++) {  
 if (gawno[0] > arr[i][j]) {  
 gawno[0] = arr[i][j];  
 gawno[2] = i;  
 }  
 if (gawno[1] < arr[i][j]) {  
 gawno[1] = arr[i][j];  
 gawno[3] = i;  
 }  
 }  
 }  
 if (gawno[2] == gawno[3]) *System*.out.println("Gawno");  
 else {  
 int[] smt = arr[gawno[2]];  
 arr[gawno[2]] = arr[gawno[3]];  
 arr[gawno[3]] = smt;  
 *System*.out.println();  
 for (int i = 0; i < line; i++) *System*.out.println(*Arrays*.*toString*(arr[i]));  
 break;  
 }  
 }  
 }  
}

***6.3. Тестирование работы программы с проверкой***

Далее в таблице 6 представлено тестирование работы программы с проверкой решения задач на языке Java.

Таблица 6

Тестирование работы программы и проверка результатов решения

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |
| 2 |  |

**Задание 12**

***7.1. Постановка задачи***

Дан двумерный квадратный массив коэффициентов системы линейных алгебраических уравнений A, одномерный массив столбца правой части СЛАУ F и одномерный массив столбца полученного решения X. При этом пользователь вводит размерность массива и данные сам и может допустить ошибку при вычислении решения СЛАУ или при вводе. Поэтому выполнить проверку соответствует ли полученное решение данной СЛАУ A\*X=F.

***7.2. Решение задачи, код программы***

import *java.util.Arrays*;  
import *java.util.Scanner*;  
  
  
public class *exc8\_12* {  
 public static void main(*String*[] *args*) {  
 *Scanner* input = new Scanner(*System*.in);  
 *System*.out.println("Введите n:");  
 int n = input.nextInt();  
 int[][] arr = new int[n][n]; *// A* int[] arr\_1 = new int[n]; *// X* int[] arr\_2 = new int[n]; *// F* int[] arr\_sr = new int[n];  
 *System*.out.println("Заполните коэффициенты: ");  
 for (int[] ints : arr) {  
 for (int anInt : ints) {  
 anInt = input.nextInt();  
 }  
 }  
 *System*.out.println("Заполните F: ");  
 for (int a : arr\_1) {  
 a = input.nextInt();  
 }  
 *System*.out.println("Заполните решения X: ");  
 for (int a : arr\_2) {  
 a = input.nextInt();  
 }  
 for (int i = 0; i < n; i++) {  
 for (int j = 0; j < n; j++) {  
 arr\_sr[i] += arr[i][j] \* arr\_1[j];  
 }  
 }  
 if (*Arrays*.*equals*(arr\_2, arr\_sr)) {  
 *System*.out.println("Все ок");  
 } else *System*.out.println("Не верно");  
 }  
}

***7.3. Тестирование работы программы***



