5 баллов

Написать программу решения квадратного уравнения. Корни только вещественные.

Квадратное уравнение – это уравнение вида ax2 + bx + c = 0, где a не равно 0.

Для решения квадратного уравнения можно использовать формулы:   
x1и x2  
где D = b2 - 4ac — дискриминант многочлена ax2 + bx + c. Если D > 0, то уравнение имеет два различных вещественных корня. Если D = 0, то корень один. Если D < 0, то нет решений.

5 баллов

Сведения о надбавках к окладу у работников предприятия представлены в виде вектора К из 20 элементов. Найти среднее арифметическое число работников, имеющих надбавку.

К = {20; 0; 30; 20; 10; 0; 10; 0; 20; 40; 0; 20; 30; 10; 20; 20; 10; 30; 0; 20}

package task\_02;

public class Task\_02 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

int[] K = {20, 0, 30, 20, 10, 0, 10, 0, 20, 40, 0, 20, 30, 10, 20, 20, 10, 30, 0, 20};

double sum = 0; //сумма всех надбавок

int n = 0; //количество получающих надбавку

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

if (K[i] > 0) {

sum += K[i];

n++;

}

}

System.out.printf("Среднее арифметическая надбавка: %.2f \n", sum / n);

}

}

run:

Среднее арифметическая надбавка: 20,67

5 баллов

Выпуск продукции ткацкими цехами предприятия представлен в виде вектора V из 10 элементов. Найти, сколько цехов предприятия выпустили продукции в объеме от 300 до 500 погонных метров.

V = {600; 500; 300; 390; 640; 500; 500; 490; 270; 450)

package task\_03;

public class Task\_03 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

int[] V = {600, 500, 300, 390, 640, 500, 500, 490, 270, 450};

int n = 0; //количество элементов в диапазоне 300-500

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

if ((V[i] >= 300) && (V[i] <= 500)) {

n++;

}

}

System.out.println("Количество цехов предприятия, которые выпустили продукции в объеме от 300 до 500 погонных метров: " + n);

}

}

run:

Количество цехов предприятия, которые выпустили продукции в объеме от 300 до 500 погонных метров: 7

5 баллов

Индексы потребительских цен представлены в виде вектора из 15 элементов. Найти, сколько раз встречается минимальный и максимальный элемент.

P = {20; 45; 23; 34; 8; 27; 45; 24; 35; 17; 29; 45; 16; 34; 8}

package task\_04;

public class Task\_04 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

//P = {20,45,23,34,8,27,45,24,35,17,29,45,16,34,8}

int[] P = {20,45,23,34,8,27,45,24,35,17,29,45,16,34,8};

int min = 0, max = 0;

int min\_n = 0, max\_n = 0;

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

if (P[i] > max){

max = P[i];

}

if (P[i] < max){

min = P[i];

}

}

System.out.println("Индексы потребительских цен:");

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

if (P[i] == max){

max\_n++;

}

if (P[i] == max){

min\_n++;

}

System.out.print(" " + P[i]);

}

System.out.println("");

System.out.println("Количество минимальных элементов (" + min + "): " + min\_n);

System.out.println("Количество минимальных элементов (" + max + "): " + max\_n);

}

}

run:

Индексы потребительских цен:

20 45 23 34 8 27 45 24 35 17 29 45 16 34 8

Количество минимальных элементов (8): 3

Количество минимальных элементов (45): 3

5 баллов

Сведения о количестве проданных товаров каждым продавцом представлены в виде целочисленного вектора Р из 20 элементов. Найти количество продавцов, объем продаж у которых больше нуля.

P = {20; 0; 30; 23; 34; 0; 10; 9; 27; 45; 0; 24; 35; 17; 29; 24; 16; 34; 0; 8}

package task\_5;

public class Task\_5 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

int[] P = {20, 0, 30, 23, 34, 0, 10, 9, 27, 45, 0, 24, 35, 17, 29, 24, 16, 34, 0, 8};

int n = 0; //количество элементов > 0

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

if (P[i] > 0) {

n++;

}

}

System.out.println("Количество продавцов, объем продаж у которых больше нуля: " + n);

}

}

run:

Количество продавцов, объем продаж у которых больше нуля: 16

6 баллов

Написать программу проверки является ли число простым.

package task\_06;

import java.util.Scanner; // импорт сканера

public class Task\_06 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

System.out.print("Введите любое целое число: ");

Scanner scan = new Scanner(System.in);

int number = scan.nextInt();

int n = 0;

for (int i = 1; i <= number; i++) {

if (number % i == 0) {

n++;

if ((n == 2)&(i == number)) {

System.out.println("Введённое число - простое");

System.exit(0);

}

}

}

System.out.println("Введённое число - не простое");

}

}

run:

Введите любое целое число: 13

Введённое число - простое

6 баллов

В последовательности вещественных чисел найти три элемента, ближайших больших по величине минимального. Вывести их на экран, отсортировав в натуральном порядке. Минимальный элемент также вывести на экран.

package task\_07;

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

public class Task\_07 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

List<Double> testList = new ArrayList();

boolean b = false;

String[] str;

do {

System.out.print("Введите через пробел не менее 4-х вещественных чисел: ");

Scanner scan = new Scanner(System.in);

String s = scan.nextLine();

str = s.split(" ");

if (str.length < 4) {

System.out.println("Необходимо ввести не менее 4-х вещественных чисел!");

b = true;

} else {

b = false;

}

} while (b);

for (String st : str) {

testList.add(Double.parseDouble(st));

}

Collections.sort(testList);

System.out.println("Минимальный элемент: " + testList.get(0));

System.out.print("Три элемента, ближайших больших по величине минимального: ");

for (int i = 1; i < 4; i++) {

System.out.print(testList.get(i) + " ");

}

System.out.println();

}

}

run:

Введите через пробел не менее 4-х вещественных чисел: 5.5 3.4 4.3 2.1 1.5 5.1 2.1 3.5 6.8 7.2 2.2

Минимальный элемент: 1.5

Три элемента, ближайших больших по величине минимального: 2.1 2.1 2.2

6 баллов

Дан набор из десяти целочисленных элементов. Найти количество элементов, содержащихся между первым и последним минимальным. Если минимальных элементов всего 2, и они идут подряд, то вывести 0. Если в наборе имеется единственный минимальный элемент, вывести на экран соответствующее уведомление.

package task\_08;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class Task\_08 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

List<Integer> testList = new ArrayList();

boolean b;

String[] str;

do {

System.out.print("Введите через пробел 10 целых чисел: ");

Scanner scan = new Scanner(System.in);

String s = scan.nextLine();

str = s.split(" ");

if (str.length < 10) {

System.out.println("Необходимо ввести 10 целых чисел!");

b = true;

} else {

b = false;

}

} while (b);

for (String st : str) {

testList.add(Integer.parseInt(st));

}

int min = testList.get(0);

int iLastMin = 0;

for (int i = 0; i < testList.size(); i++) {

if (min >= testList.get(i)) {

min = testList.get(i);

iLastMin = i;

}

}

System.out.println("Минимальный элемент: " + min);

int count = iLastMin - testList.indexOf(min) - 1;

if (count == -1) {

System.out.println("В наборе имеется единственный минимальный элемент");

} else {

System.out.println("Элементов между первым и последним минимальными: " + count);

}

}

}

6 баллов

Написать программу, которая сортирует последовательность чисел по убыванию. При разработке программы можно использовать классы-коллекции.

package task\_09;

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

package task\_09;

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

public class Task\_09 {

public static void main(String[] args) {

List<Double> testList = new ArrayList();

boolean b;

String[] str;

do {

System.out.print("Введите через пробел не менее 2-х чисел: ");

Scanner scan = new Scanner(System.in);

String s = scan.nextLine();

str = s.split(" ");

if (str.length < 2) {

System.out.println("Необходимо ввести не менее 2-х вещественных чисел!");

b = true;

} else {

b = false;

}

} while (b);

for (String st : str) {

testList.add(Double.parseDouble(st));

}

Collections.sort(testList);

Collections.reverse(testList);

for (Double itm : testList) {

System.out.print(itm + " ");

}

System.out.println();

}

}

6 баллов

Проверить, является ли квадратная матрица симметричной относительно побочной диагонали.

package tasck\_10;

import java.util.Scanner;

public class Tasck\_10 {

public static void main(String[] args) {

System.out.print("Введите размер матрицы: ");

Scanner scan = new Scanner(System.in);

//размер матрицы

int w = scan.nextInt();

//создание и инициализация матрицы

int[][] a = new int[w][w];

// System.out.println("Создана матрица:");

for (int i = 0; i < w; i++) {

for (int j = 0; j < w; j++) {

System.out.print("Задайте значение a["+i+"]["+j+"]:");

a[i][j] = scan.nextInt();

}

}

for (int i = 0; i < w; i++) {

for (int j = 0; j < w; j++) {

System.out.print(a[i][j] + " ");

}

System.out.println();

}

boolean symmetric = true; //селектор для вывода результата

//цикл для сравнения элементов

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

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

int m = a.length-1-j;

int n = a.length-1-i;

System.out.println("a[" + i + "][" + j + "]=" + a[i][j] + " <--> a[" + m + "][" + n + "]=" + a[m][n]);

if (a[i][j] != a[m][n]) {

symmetric = false;

break;

}

}

if (!symmetric) {

System.out.println("Матрица не симметрична.");

break;

}

}

if (symmetric) {

System.out.println("Матрица симметрична.");

}

}

}

7 баллов

Ввести с консоли строку, содержащую набор слов, разделённых пробелами, точками, дефисами и запятыми. Найти и вывести на экран слова, для которых последняя буква одного слова совпадает с первой буквой другого слова.

package task\_11;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class Task\_11 {

public static void main(String[] args) {

System.out.println("Введите набор слов, разделённых пробелами, точками, дефисами и запятыми:");

String s = new Scanner(System.in).nextLine();

String[] str = s.split("\\s+|\\.+|-+|,+");

List<String> sList = new ArrayList();

for (String st : str) {

if (st.length() > 0) {

sList.add(st);

}

}

if (sList.size() > 0) {

for (int i = 0; i < sList.size() - 1; i++) {

if (sList.get(i).length() > 0) {

char first = sList.get(i).charAt(0);

for (int j = i + 1; j < sList.size(); j++) {

char last = sList.get(j).charAt(sList.get(j).length() - 1);

if (first == last) {

System.out.println(sList.get(i) + " " + sList.get(j));

}

}

}

}

}

}

}

7 баллов №12

В последовательности целых чисел определить **сумму** четных *положительных* чисел и **произведение** *отрицательных*, меньших -10. Действия по определению суммы, произведения, проверки четности и отрицательности реализовать с использованием методов с параметрами.

package task\_12;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class Task\_12 {

public static void main(String[] args) {

List<Integer> testList = new ArrayList();

boolean b;

String[] str;

do {

System.out.print("Введите через пробел не менее 3-х целых чисел: ");

Scanner scan = new Scanner(System.in);

String s = scan.nextLine();

str = s.split(" ");

if (str.length < 3) {

System.out.println("Необходимо ввести не менее 3-х целых чисел!");

b = true;

} else {

b = false;

}

} while (b);

for (String st : str) {

testList.add(Integer.parseInt(st));

}

System.out.println("Сумма четных положительных чисел: " + Summa(testList));

System.out.println("Произведение отрицательных, меньших -10: " + Mux(testList));

}

public static boolean isPositiveAndEven(int n) {

return n > 0 && n % 2 == 0;

}

public static boolean isNegMin10(int n) {

return n < -10;

}

private static int Summa(List<Integer> tList) {

int sum = 0;

for (int n : tList) {

if (isPositiveAndEven(n)) {

sum += n;

}

}

return sum;

}

private static int Mux(List<Integer> tList) {

int m = 1;

boolean b = false;

for (int n : tList) {

if (isNegMin10(n)) {

b = true;

m \*= n;

}

}

if (b) {

return m;

} else {

return 0;

}

}

}

**Вариант №2**

package method;

import java.util.List;

public class Method {

public static boolean isPositiveAndEven(int n) {

return n > 0 && n % 2 == 0;

}

public static boolean isNegAndMin10(int n) {

return n < -10;

}

public static int sum(List<Integer> l) {

return l.stream().filter(n -> isPositiveAndEven(n)).mapToInt(Integer::intValue).sum();

}

public static int mult(List<Integer> l) {

return l.stream().filter(n -> isNegAndMin10(n)).reduce(1, (x, y) -> x \* y);

}

public static void main(String[] args) {

List<Integer> list = java.util.Arrays.asList(1, 2, -11, 0, -8, -15, 12, -11);

System.out.println(mult(list));

}

}

7 баллов №13

В последовательности вещественных чисел поменять местами соседние четные и нечетные по номеру элементы. Действия по определению четности индекса и перестановки двух элементов реализовать с использованием методов с параметрами.

package task\_13;

import java.util.Scanner;

public class Task\_13 {

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

System.out.print("Введите размер массива: ");

int n = scan.nextInt();

Double[] a = new Double[n];

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

a[i] = (double) Math.round((Math.random() \* 30) - 15);

}

for (double itm : a) {

System.out.print(itm + " ");

}

System.out.println();

for (int i = 1; i < a.length - 2; i++) {

double tmp = a[i];

a[i] = a[i + 2];

a[i + 2] = tmp;

}

for (double itm : a) {

System.out.print(itm + " ");

}

System.out.println();

}

}

7 баллов №14

Найти седловые точки в матрице (седловая точка – это число, которое одновременно является максимальным элементом в соответствующем столбце матрицы и минимальным элементом в соответствующей строке матрицы.)

package task\_14;

import java.util.Scanner;

public class Task\_14 {

public static void main(String[] args) {

System.out.print("Введите количество строк матрицы: ");

Scanner scan = new Scanner(System.in);

int row = scan.nextInt();

System.out.print("Введите количество столбцов матрицы: ");

int col = scan.nextInt();

//создание и инициализация матрицы

int[][] a = new int[row][col];

// System.out.println("Создана матрица:");

for (int i = 0; i < row; i++) {

for (int j = 0; j < col; j++) {

System.out.print("Задайте значение a[" + i + "][" + j + "]:");

a[i][j] = scan.nextInt();

}

}

for (int i = 0; i < row; i++) {

for (int j = 0; j < col; j++) {

System.out.print(a[i][j] + " ");

}

System.out.println();

}

int[] aMax = new int[col];

for (int i = 0; i < col; i++) {

aMax[i] = a[0][i];

for (int j = 1; j < row; j++) {

if (aMax[i] < a[j][i]) {

aMax[i] = a[j][i];

}

}

}

int[] aMin = new int[row];

for (int i = 0; i < row; i++) {

aMin[i] = a[i][0];

for (int j = 1; j < col; j++) {

if (aMin[i] > a[i][j]) {

aMin[i] = a[i][j];

}

}

for (int i = 0; i < row; i++) {

for (int j = 0; j < col; j++) {

if ((a[i][j] == aMax[j]) & (a[i][j] == aMin[i])) {

System.out.println("Cедловая точка: a["+i+"]["+j+"] = " + a[i][j]);

}

}

}

}

}

7 баллов №15

Описать метод remove(A, N, k), осуществляющий удаление последних k столбцов вещественной матрицы A размера NхN (0 < k < N – 1). Матрица A, N и k — входные параметры. Новая матрица – выходной параметр. С помощью этого метода осуществить удаление k1, k2, k3 столбцов из матриц x, y, z. размеров N1, N2, N3. Ссылкам на матрицы x, y, z присвоить значения результатов работы метода.

package task\_15;

import java.util.Scanner;

public class Task\_15 {

public static int[][] remove(int[][] A, int N, int k) {

int[][] newA = new int[N][N - k];

for (int i = 0; i < N; i++) {

for (int j = 0; j < N - k; j++) {

newA[i][j] = A[i][j];

}

}

return newA;

}

public static int rnd(int max) {

return (int) (Math.random() \* ++max);

}

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

for (int I = 0; I < 3; I++) {

System.out.print("Введите размер квадратной матрицы: ");

int n = in.nextInt();

int[][] array = new int[n][n];

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

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

array[i][j] = rnd(100);

}

}

System.out.println("Исходная матрица:");

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

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

System.out.print(array[i][j] + "\t");

}

System.out.println();

}

boolean b = false;

int k;

do {

System.out.print("Сколько столбцов удалить: ");

k = in.nextInt();

if (k > n) {

System.out.println("Количество удаляемых столбцов не может быть больше " + n);

} else {

b = true;

}

} while (!b);

int[][] x = remove(array, n, k);

if (x[0].length > 0) {

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

for (int j = 0; j < n - k; j++) {

System.out.print(x[i][j] + "\t");

}

System.out.println();

}

} else {

System.out.println("В результирующей матрице нет элементов");

}

}

}

}