Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана»

(национальный исследовательский университет)

Московский техникум космического приборостроения

**ЛАБОРАТОРНАЯ РАБОТА №5**

по теме: «Тестирование программного обеспечения методом “чёрного ящика”»

Специальность 09.02.07 Информационные системы и программирование

Группа ТИП-71

|  |  |
| --- | --- |
| Проверил | Н.А. Сидорова |
| Разработал | С.С. Бобылёв |

Москва 2021

**СОДЕРЖАНИЕ**

1 Постановка задачи 3

2 Схемы алгоритмов программы 4

2.1 Схема алгоритма основной программы 4

2.2 Схема алгоритма функции size\_v 5

2.3 Схема алгоритма функции write\_v 6

2.4 Схема алгоритма функции sort\_v\_Shell 7

3 Листинг программы 8

4 Результаты выполнения программы 12

1. Постановка задачи

Даны три вектора x(n), y(n), z(n). Построить матрицу А, строками которой являются соответственно векторы x, y, z, упорядоченные в порядке возрастания. Упорядочение вектора по возрастанию оформить в виде подпрограммы.

Исходные данные:

x={5.2; 1.4; 3.5}, y ={3.4; 2.7; 6.1}, z ={7.8; 4.2; 2.1}

n≤7 – размерность векторов и матрицы.

1. Схемы алгоритмов программы

2.1 Схема алгоритма основной программы



2.2 Схема алгоритма функции size\_v



* 1. Схема алгоритма функции write\_v



* 1. Схема алгоритма функции sort\_v\_Shell



1. Листинг программы

/\*Программа lab5.cs

Лабораторная работа №5 по профессиональному модулю МДК01.02 Поддержка и тестирование программных модулей

Тема "Тестирование программного обеспечения методом ‘чёрного ящика’"

Язык: С#

Разработал Бобылёв Сергей ТИП-71

Дата: 05.10.2021г.

Задание и ожидаемые входные данные:

Даны три вектора x(n), y(n), z(n). Построить матрицу А, строками которой являются соответственно векторы x, y, z, упорядоченные в порядке возрастания.

Упорядочение вектора по возрастанию оформить в виде подпрограммы.

Исходные данные:

x={5.2; 1.4; 3.5}, y ={3.4; 2.7; 6.1}, z ={7.8; 4.2; 2.1}

n≤7 – размерность векторов и матрицы.

Переменные, используемые в программе:

n- размер векторов;

x- первый вектор;

y- второй вектор;

z-третий вектор;

1. матрица, состоящая из векторов;

j- переменная, используемая для счётчиков столбцов;

1. переменная, используемая для счётчиков строк;

s- переменная, используемая для вывода матрицы;

v- локальная переменная, накапливающая значение вектора;

d- локальная переменная, используемая при сортировке;

count - локальная переменная, которая накапливает в себе значение, для последующей замены.

Функции, используемые в программе:

size\_v - функция ввода размеров векторов;

write\_v - функция ввода значения вектора.

sort\_v\_Shell - функция сортировки вектора методом Шелла;

using System;

namespace ConsoleApp1

{

class Program

{

//Функция ввода размера вектора

static int size\_v()

//Локальная переменная:

x- переменная отвечающая за размер векторов.

{

int x = -1;

while (x > 7 || x < 1)

{

try

{

Console.WriteLine("ВВЕДИТЕ РАЗМЕР ВЕКТОРОВ");

x = int.Parse(Console.ReadLine());

}

catch(Exception ex)

{

Console.WriteLine(ex.Message);

}

}

return x;

}

//Функция ввода значения вектора

static void write\_v(float[] v,string s)

//Формальные параметры:

v- параметр накапливающая значение вектора;

s- параметр используемая для вывода матрицы.

//Локальная переменная:

i- переменная, используемая для счётчиков строк.

{

int i = 0;

while (i < v.Length)

{

try

{

Console.WriteLine(string.Format("ВВЕДИТЕ ЗНАЧЕНИЕ {0:d} ВЕКТОРА {1:s}",i+1,s));

v[i] = float.Parse(Console.ReadLine());

i++;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

}

//Функция сортировки вектора методом Шелла

static void sort\_v\_Shell(float[] v)

//Формальный параметр:

v- параметр накапливающая значение вектора.

//Локальные переменные:

d- локальная переменная, используемая при сортировке;

i- переменная, используемая для счётчиков строк;

j- переменная, используемая для счётчиков столбцов;

count - локальная переменная, которая накапливает в себе значение, для последующей замены.

{

int d = v.Length;

d = d / 2;

while (d > 0)

{

for (int i = 0; i < v.Length - d; i++)

{

int j = i;

while (j >= 0 && v[j] > v[j + d])

{

float count = v[j];

v[j] = v[j + d];

v[j + d] = count;

j--;

}

}

d = d / 2;

}

}

static void Main(string[] args)

{

int n = size\_v();

float[] x = new float[n];

write\_v(x,"x");

sort\_v\_Shell(x);

float[] y = new float[n];

write\_v(y,"y");

sort\_v\_Shell(y);

float[] z = new float[n];

write\_v(z,"z");

sort\_v\_Shell(z);

//Задания матрицы

float[,] A = new float[3, x.Length];

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

{

A[0,i] = x[i];

}

for (int i = 0; i < y.Length; i++)

{

A[1,i] = y[i];

}

for (int i = 0; i < z.Length; i++)

{

A[2,i] = z[i];

}

//Вывод матрицы

string s = "";

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < n; j++)

{

s += string.Format(" {0:f} ", A[i, j]);

}

s += "\n";

}

Console.WriteLine(s);

}

}

}

1. Результаты выполнения программы

Программа выводит матрицу, строками которой являются соответственно векторы x, y, z, упорядоченные в порядке возрастания. Результаты выполнения программы представлены на рисунке 4.1.

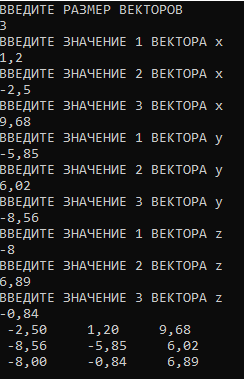


Рисунок 4.1 – Результаты выполнения программы

Для проверки правильности работы программы был проведён визуальный просмотр полученных результатов:

Исходные значения векторов:

x= 1.2, -2.5, 9.68;

y= -5.85, 6.02, -8.56;

z= -8, 6.89, -0.84;

Отсортированные вектора будут выглядеть так:

x=-2.5, 1.2, 9.68;

y=-8.56, -5.85, 6.02;

z=-8, -0.84, 6.89;

Так как результаты визуального просмотра совпали с результатами программы, следовательно программа работает корректно.