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 №6**

по теме: «Модульное тестирование»

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 14

5 Тестирование программы 15

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
   1. Схема алгоритма основной программы



* 1. Схема алгоритма функции size\_v



* 1. Схема алгоритма функции write\_v



* 1. Схема алгоритма функции sort\_v\_Shell



1. Листинг программы

/\*Программа lab6.cs

Лабораторная работа №6 по профессиональному модулю МДК01.02 Поддержка и тестирование программных модулей

Тема "Модульное тестирование"

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. матрицы, состоящая из векторов;

j- переменная, используемая для счётчиков;

1. переменная, используемая для счётчиков;

s- переменная, используемая для вывода матрицы;

v- локальная переменная, накапливающая значение вектора;

d- локальная переменная, используемая при сортировке;

count- локальная переменная, запоминающая одно из значений вектора, для дальнейшего его использования.

//Листинг основной программы

using System;

using static ConsoleApp1.moduleAttribute;

namespace ConsoleApp1

{

class Program

{

//Функция ввода размера вектора

static void Main(string[] args)

//Локальная переменная:

x- переменная отвечающая за размер векторов.

{

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);

}

}

}

//Листинг процедуры

using System;

namespace ConsoleApp1

{

internal class moduleAttribute : Attribute

{

public static int size\_v()

{

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

public 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

public 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;

}

}

}

}

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.

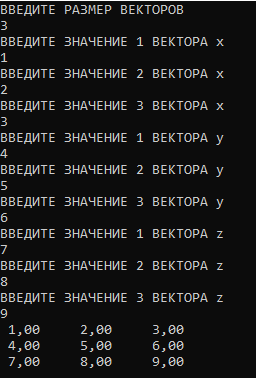


Рисунок 4.1 – Результаты выполнения программы

1. Тестирование программы

Тестирование – это набор процедур и действий, предназначенных для демонстрации правильности работы программы в заданных режимах и внешних условиях. Цель тестирования – выявить наличие ошибок или убедительно продемонстрировать их отсутствие.

Процесс тестирования проходит в 3 этапа:

— проверка поведения программы в нормальных условиях;

— проверка поведения программы в экстремальных условиях;

— проверка поведения программы в исключительных ситуациях.

Каждый из этапов предполагает задание определенного, характерного для данного этапа набора данных.

* 1. Тестирование программы в нормальных условиях

Размер векторов должен быть в пределах диапазона [2;7]. Результаты тестирования программы в нормальных ситуациях представлены на рисунке 5.1-5.3.



Рисунок 5.1 - Ввод размеров векторов

Значения векторов должны быть в пределах диапазона [-100;100].

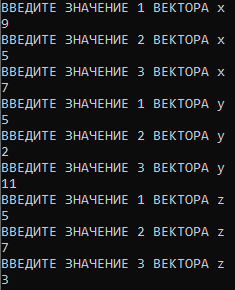


Рисунок 5.2 – Ввод элементов векторов x, y и z.



Рисунок 5.3 – Результаты работы программы.

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

Исходные значения векторов:

x=9,5,7;

y=5,2,11;

z= 5,7,3;

Отсортированные вектора будут выглядеть так:

x=5,7,9;

y=2,5,11;

z= 3,5,7;

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

Во время тестирования программы в нормальных условиях ошибок не выявлено.

* 1. Тестирование программы в экстремальных условиях

Размер векторов должен быть в пределах диапазона [2;7]. Результаты тестирования программы в нормальных ситуациях представлены на рисунке 5.4-5.9.

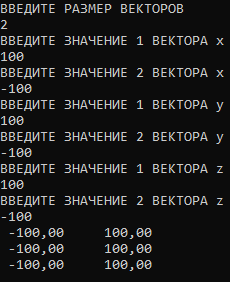


Рисунок 5.4 – Результат работы программы.



Рисунок 5.5 – Ввод размеров векторов.

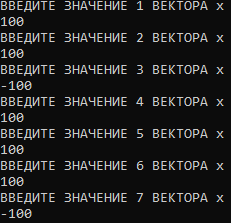


Рисунок 5.6 – Ввод элементов вектора х.

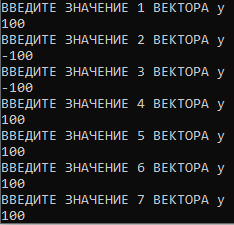


Рисунок 5.7 – Ввод элементов вектора у.

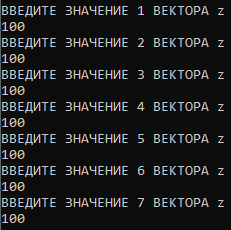


Рисунок 5.8 – Ввод элементов вектора z.

 Рисунок 5.9 – Результат работы программы.

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

Исходные значения векторов:

x= 100, 100, -100, 100, 100, 100, -100;

y= 100, -100, -100, 100, 100, 100, 100;

z= 100, 100, 100, 100, 100, 100, 100;

Отсортированные вектора будут выглядеть так:

x= -100, -100, 100, 100, 100, 100, 100;

y= -100, -100, 100, 100, 100, 100, 100;

z= 100, 100, 100, 100, 100, 100, 100;

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

Во время тестирования программы в экстремальных условиях ошибок не выявлено.

* 1. Тестирование программы в исключительных ситуациях

При попытке ввода значений, находящихся за пределами диапазонов, букв или знаков программа выдает сообщение с просьбой ввести значение, удовлетворяющее диапазону до тех пор, пока не получит подходящее значение. Результаты тестирования программы в исключительных ситуациях представлены на рисунке 5.10-5.13.

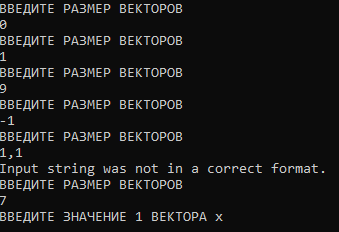


Рисунок 5.10 – Ввод размеров векторов.

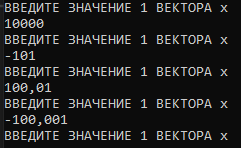


Рисунок 5.11 – Ввод значений вектора.



Рисунок 5.12 – Попытка ввода пустого значения.



Рисунок 5.13 – Попытка ввода буквы.

Во время тестирования программы в исключительных ситуациях ошибок не выявлено.