

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА  
(САМАРСКИЙ УНИВЕРСИТЕТ)»**

Институт информатики и кибернетики

Кафедра программных систем

**ОТЧЁТ**

**по лабораторной работе**

Лабораторная работа № 01

Основы языка С#: Классы. Повторениепо дисциплине «Языки программирования и структуры данных»

Выполнил Сидоров А.О., 6103

Проверил Котенева С.Э.

Самара  
2024

ЗАДАНИЕ

Задание 1.

Прочитать необходимый теоретический материал.

Задание 2.

Описать класс с именем «ArrayVector». Класс описывает вектор в n-мерном пространстве. Координаты конца вектора задаются массивом, количество элементов которого равно n-размерности пространства. Класс должен иметь следующую структуру:

* поле – массив элементов целого типа (координаты вектора в пространстве);
* конструктор с параметром – длиной массива;
* конструктор без параметров, создающий массив из 5 элементов;
* метод установки элемента массива по индексу SetElement() (параметры метода – индекс элемента и устанавливаемое значение);
* метод чтения элемента массива по индексу GetElement() (параметр метода – индекс элемента). Примечание: вместо методов SetElement() и GetElement() можно реализовать индексатор – он будет необходим в последующих лабораторных работах;
* метод GetNorm() получения модуля (длины, нормы) вектора;
* метод SumPositivesFromChetIndex() подсчета суммы всех положительных элементов массива с четными номерами;
* метод SumLessFromNechetIndex() подсчета суммы тех элементов массива, которые имеют нечетные номера и одновременно меньше среднего значения всех модулей элементов массива;
* метод MultChet() подсчета произведения всех четных положительных элементов (по значению);
* метод MultNechet() подсчета произведения всех нечетных элементов (по значению), не делящихся на три; Примечание: во всех четырех методах производящих действия с элементами массива нумерация элементов массива для конечного пользователя должна начинаться с единицы. То есть, в массиве [2, 3, 4, 5] элемент со значением «3» это второй элемент по индексу с точки зрения пользователя.
* метод SortUp() сортировки массива по возрастанию;
* метод SortDown() сортировки массива по убыванию.

Где необходимо выбрасывать разные типы исключений.

Задание 3.

Добавить класс с именем «Vectors», содержащий публичные статические методы:

* сложения двух векторов Sum(), который принимает в качестве параметра 2 объекта типа ArrayVector и возвращает новый объект ArrayVector;
* скалярного произведения двух векторов Scalar(), который принимает в качестве параметра 2 объекта типа ArrayVector и возвращает вещественное число;
* умножения вектора на число MultNumber(), который принимает в качестве параметра объект типа ArrayVector и вещественное число и возвращает новый объект ArrayVector;
* получения модуля (длины) вектора GetNormSt(), который принимает в качестве параметра объект типа ArrayVector и возвращает вещественное число.

Задание 4.

Подготовить отчет о работе.

КОД ПРОГРАММЫ

namespace newlab21

{

class Program

{

public static void Main()

{

bool alive = true;

Console.WriteLine(123);

while (alive == true)

{

Console.WriteLine("\nСеместр 2, Лабораторная работа 1, выполнил студент Артемий Сидоров\n");

Console.WriteLine("1 - использование класса ArrayVector");

Console.WriteLine("2 - использование класса Vectors");

Console.WriteLine("3 - завершение работы\n");

string menu = Console.ReadLine()!;

switch (menu)

{

case "1":

{

Console.WriteLine("Введите координаты вектора через пробел");

string[] temp = Console.ReadLine().Split(" ");

ArrayVector vector = new ArrayVector(temp.Length);

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

{

vector[i] = int.Parse(temp[i]);

}

try

{

int a = vector.SumPositivesFromChetIndex();

Console.WriteLine("Сумма всех положительных элементов с чётными номерами: " + a);

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

try

{

int a = vector.SumLessFromNechetIndex();

Console.WriteLine("Сумма всех элементов меньше среднего значения всех модулей: " + a);

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

try

{

int a = vector.MultChet();

Console.WriteLine("Произведение всех чётных положительных элементов: " + a);

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

try

{

int a = vector.MultNechet();

Console.WriteLine("Произведение всех нечетных положительных элементов: " + a);

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

int[] b = vector.SortUp();

Console.Write("Сортировка по возрастанию");

PrintArray(b);

int[] c = vector.SortDown();

Console.Write("Сортировка по убыванию");

PrintArray(c);

double d = vector.GetNorm();

Console.WriteLine("Модуль вектора: " + d);

break;

}

case "2":

{

Console.WriteLine("Введите координаты вектора 1 через пробел");

string[] temp1 = Console.ReadLine().Split(" ");

ArrayVector vector1 = new ArrayVector(temp1.Length);

for (int i = 0; i < temp1.Length; i++)

{

vector1[i] = int.Parse(temp1[i]);

}

Console.WriteLine("Введите координаты вектора 2 через пробел");

string[] temp2 = Console.ReadLine().Split(" ");

ArrayVector vector2 = new ArrayVector(temp2.Length);

for (int i = 0; i < temp2.Length; i++)

{

vector2[i] = int.Parse(temp2[i]);

}

try

{

ArrayVector sum = Vectors.Sum(vector1, vector2);

double scalar = Vectors.Scalar(vector1, vector2);

sum.PrintArray();

Console.WriteLine(scalar);

}

catch (Exception e)

{

Console.WriteLine(e.Message);

}

Console.Write("Введите число, на которое умножить вектор: ");

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

ArrayVector mult = Vectors.MultNumber(vector1, n);

Console.Write("Вектор после умножения: ");

mult.PrintArray();

break;

}

case "3":

{

Console.WriteLine("\nВы завершили работу программы\n");

alive = false;

break;

}

default:

{

Console.WriteLine("\nПовторите ввод\n");

break;

}

}

Console.WriteLine("\n\nНажмите на любую клавишу, чтобы продолжить\n");

Console.ReadKey();

}

}

private static void PrintArray(int[] array)

{

Console.Write("{ ");

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

{

Console.Write(array[i]);

if (i < array.Length - 1)

{

Console.Write(", ");

}

}

Console.WriteLine(" }");

}

}

}

namespace newlab21

{

internal class ArrayVector

{

public int[] vector;

public ArrayVector(int size)

{

if (size < 1)

{

throw new Exception("Размерность вектора должна быть больше нуля!");

}

vector = new int[size];

}

public int Length

{

get

{

return vector.Length;

}

}

public ArrayVector() // конструктор без параметра

{

vector = new int[5];

}

public int this[int i] // индексатор

{

get

{

return vector[i];

}

set

{

vector[i] = value;

}

}

public double GetNorm() // метод получения модуля вектора

{

int temp = 0;

foreach (int i in vector)

{

temp += i\*i;

}

return Math.Sqrt(temp);

}

public int SumPositivesFromChetIndex()

{

int sumPositives = 0;

for (int i = 1; i < vector.Length; i += 2) // чётные по номерам, а не по индексам, начиная со второго элемента

{

if (vector[i] > 0)

{

sumPositives += vector[i];

}

}

if (sumPositives == 0)

{

throw new Exception("Нет положительных элементов с чётным номером!");

}

return sumPositives;

}

public int SumLessFromNechetIndex()

{

int sumLess = 0;

foreach (int i in vector)

{

sumLess += Math.Abs(i);

}

double sumAverage = sumLess / vector.Length;

int sum = 0;

for (int i = 0; i < vector.Length; i += 2)

{

if (vector[i] < sumAverage)

{

sum += vector[i];

}

}

if (sum == 0)

{

throw new Exception("Нет элементов с нечётным номером, которые были бы меньше среднего значения всех модулей элементов массива!");

}

return sum;

}

public int MultChet()

{

int multChet = 1;

bool isError = true;

foreach (int i in vector)

{

if ((i % 2 == 0) && i > 0)

{

multChet \*= i;

isError = false;

}

}

if (isError)

{

throw new Exception("Нет положительных чётных элементов!");

}

return multChet;

}

public int MultNechet()

{

int multNechet = 1;

bool isError = true;

foreach (int i in vector)

{

if (i % 2 != 0 && i % 3 != 0)

{

multNechet \*= i;

isError = false;

}

}

if (isError)

{

throw new Exception("Нет нечётных элементов, не делящихся на три!");

}

return multNechet;

}

public int[] SortUp()

{

int[] temp = (int[])vector.Clone();

Array.Sort(temp);

return temp;

}

public int[] SortDown()

{

int[] temp = (int[])vector.Clone();

Array.Sort(temp);

Array.Reverse(temp);

return temp;

}

public void PrintArray()

{

Console.Write("{ ");

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

{

Console.Write(vector[i]);

if (i < vector.Length - 1)

{

Console.Write(", ");

}

}

Console.WriteLine(" }");

}

}

}

namespace newlab21

{

internal class Vectors

{

public static ArrayVector Sum(ArrayVector vector1, ArrayVector vector2)

{

if (vector1.Length != vector2.Length)

{

throw new Exception("Размерность векторов не совпадает!");

}

ArrayVector sum = new ArrayVector(vector1.Length);

for (int i = 0; i < vector1.Length; i++)

{

sum[i] = vector1[i] + vector2[i];

}

return sum;

}

public static double Scalar(ArrayVector vector1, ArrayVector vector2)

{

if (vector1.Length != vector2.Length)

{

throw new FormatException("Размерность векторов не совпадает!");

}

double scalar = 0;

for (int i = 0; i < vector1.Length; i++)

{

scalar += vector1[i] \* vector2[i];

}

return scalar;

}

public static ArrayVector MultNumber(ArrayVector vector, int number)

{

ArrayVector multNumber = new ArrayVector(vector.Length);

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

{

multNumber[i] = vector[i] \* number;

}

return multNumber;

}

public static double GetNormSt(ArrayVector vector)

{

return vector.GetNorm();

}

}

}

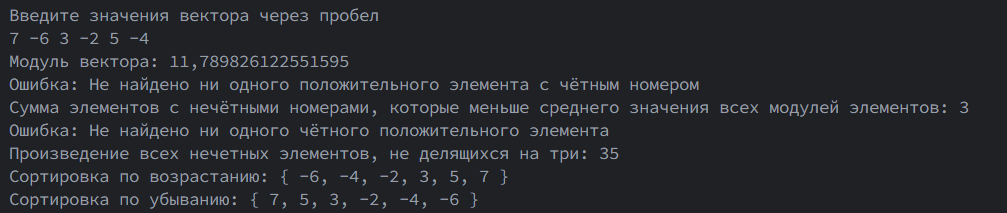


Рисунок 1 – Работа класса ArrayVector

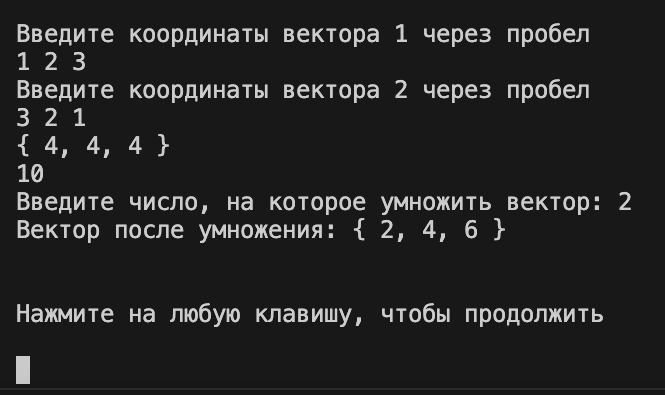


Рисунок 2 – Работа класса Vectors

ВЫВОДЫ

В лабораторной работе были использованы возможности языка C#:

* ветвления if;
* цикл с параметром for;
* цикл foreach;
* методы;
* классы;
* индексатор.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Полное руководство по языку программирования С# 11 и платформе .NET 7. – Режим доступа: URL: <https://metanit.com/sharp/tutorial/> (дата обращения: 24.12.23).