**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В. И. Ульянова (Ленина)**

**Кафедра информационной безопасности**

отчет

**по лабораторной работе №5**

**по дисциплине «Объектно-ориентированное программирование»**

Тема: **Создание и использование массивов**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 3363 |  | Овсейчик Н.И. Минко Д.А. |
| Преподаватель |  | Новакова Н.Е. |

**Цель работы**

Цель лабораторной работы заключается в освоении базовых операций с массивами, включая их создание, заполнение, обработку данных, а также использование методов для анализа содержимого массивов. В рамках работы выполняется чтение данных из файла, их обработка с подсчетом гласных, согласных и строк, а также выполнение операции умножения матриц с использованием методов и массивов. Дополнительно рассматривается обработка исключений для повышения надежности программы.

**Ход работы**

1. Работа с массивом.

Открыта Visual Studio 2008. В меню File выбрано New и затем Project. Из списка шаблонов выбрана Console Application, указанное название проекта — FileDetails. Установлен путь к проекту С:\Labs\Lab5. В метод Main класса FileDetails добавлен параметр массив строк args и выражение Console.WriteLine для вывода длины args. Добавлен цикл foreach для записи каждой строки из args на консоль. Выполнена компиляция, исправлены ошибки и сохранен результат (рис. 1).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 1 – Добавление массива args и вывод его длины

Создан текстовый файл text.txt, содержащий гласные и согласные буквы. Запущена программа из командной строки с передачей имени файла в метод Main. В метод Main добавлено выражение для объявления переменной filename типа string, инициализированной как args[0]. Из исходного кода, представленного в FileDetails.cs, удалены комментарии, относящиеся к объявлению FileStream и StreamReader. Определена длина текстового файла. В метод Main добавлено выражение для переменной contents, как массива символов, инициализированного длиной файла. Введено выражение for, которое читает символы из reader по одному и добавляет их в contents, используя метод Read и преобразование в char. Добавлено выражение foreach, которое записывает символьный массив на консоль и проверяет успешность считывания файла в массив contents. Выполнена компиляция программы, исправлены ошибки и сохранен результат. Программа запущена из командной строки (рис. 2).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 2 – Чтение файла в массив символов и вывод в консоль

Закомментировано выражение foreach и закрыт объект Reader, вызвав метод StreamReader.Close(). Объявлен новый статический метод Summarize в классе FileDetails, который принимает массив символов в качестве параметра и не возвращает значений. В метод Main добавлено выражение для вызова метода Summarize с аргументом contents. В методе Summarize добавлено выражение foreach, которое проверяет каждый символ из contents, определяет, является ли символ гласной, согласной или переходом на новую строку, и подсчитывает количество символов в каждой группе. Для проверки символов на гласные используется метод IndexOf. Реализованы операторы вывода на консоль общего количества символов, количества гласных букв, согласных букв и числа строк в файле. Выполнена компиляция программы, исправлены ошибки и сохранен результат (рис. 3).

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

Рисунок 3 – Реализован метод Summarize: подсчет гласных, согласных и строк файла

Сделана модификация предыдущей версии кода. Добавлена обработка исключений (рис. 4).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 4 – Добавлена модификация в виде обработки исключений

1. Использование методов со ссылочными параметрами.

Создан новый проект типа Console Application с названием, например, Matrix. Создан класс MatrixMultiply. В метод Main добавлены два массива 2х2, названные a и b, и определен массив result 2х2. Реализованы операторы ввода четырех чисел для каждого массива и вывода результата перемножения матриц на консоль. Выполнена компиляция программы, исправлены ошибки и сохранен результат (рис. 5).

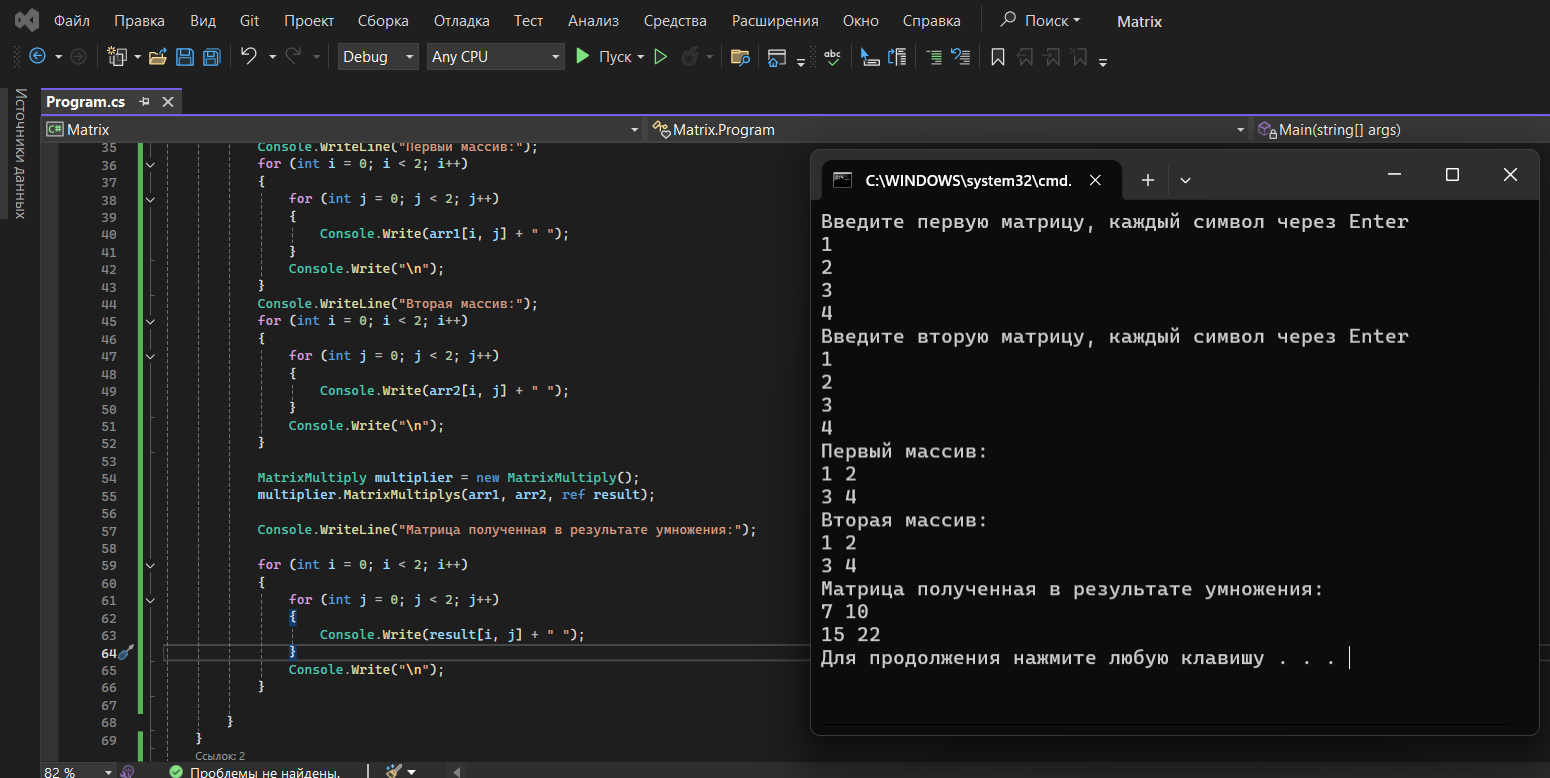


Рисунок 5 – Перемножение матриц 2х2 и вывод результата

**Вывод**

В ходе выполнения лабораторной работы были освоены основные операции работы с массивами, такие как чтение данных из текстового файла, их хранение и обработка. В первом упражнении удалось успешно реализовать программу для подсчета общего количества символов, гласных, согласных букв и строк в файле. Были использованы циклы, методы обработки символов и операторы вывода данных на консоль. Во втором упражнении было реализовано перемножение двух матриц с помощью массивов, что позволило углубить знания в работе с многомерными массивами и операциями над ними. Все задачи были выполнены, программа корректно обрабатывала вводимые данные и выводила результаты на экран.

ИСХОДНЫЙ КОД

Упражнение 1 (9):

using System;

using System.IO;

class FileDetails

{

static void Main(string[] args)

{

Console.WriteLine(args.Length);

foreach (string arg in args)

{

Console.WriteLine(arg);

}

}

}

Упражнение 1 (17):

using System;

using System.IO;

class FileDetails

{

static void Main(string[] args)

{

int length;

string fileName;

// Вывод длины массива args

Console.WriteLine("Lengths of args: " + args.Length);

// Вывод всех аргументов

foreach (string arg in args)

{

Console.WriteLine(arg);

}

// Получение имени файла

fileName = args[0];

Console.WriteLine("Filename: " + fileName);

// Открываем файл и читаем его содержимое

FileStream stream = new FileStream(fileName, FileMode.Open);

StreamReader reader = new StreamReader(stream);

// Получаем длину файла

length = (int)stream.Length;

char[] contents = new char[length];

// Читаем символы по одному и добавляем их в массив

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

{

contents[i] = (char)reader.Read();

}

//Выводим символы в консоль

foreach (char c in contents)

{

Console.WriteLine(c);

}

// Закрываем потоки

reader.Close();

stream.Close();

}

}

//cd C:\Users\Nef0o0r\OneDrive\Study\Второй курс\ООП C#\Лабы\Laba5\Код\FileDetails\FileDetails\bin\Debug>

//FileDetails.exe text.txt

//Код работает только через консоль!!!

Упражнение 1 (23):

using System;

using System.IO;

class FileDetails

{

static void Main(string[] args)

{

int length;

string fileName;

// Вывод длины массива args

Console.WriteLine("Lengths of args: " + args.Length);

// Вывод всех аргументов

foreach (string arg in args)

{

Console.WriteLine(arg);

}

// Получение имени файла

fileName = args[0];

Console.WriteLine("Filename: " + fileName);

// Открываем файл и читаем его содержимое

FileStream stream = new FileStream(fileName, FileMode.Open);

StreamReader reader = new StreamReader(stream);

// Получаем длину файла

length = (int)stream.Length;

char[] contents = new char[length];

// Читаем символы по одному и добавляем их в массив

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

{

contents[i] = (char)reader.Read();

}

//Выводим символы в консоль

/\*foreach (char c in contents)

{

Console.WriteLine(c);

}\*/

// Закрываем потоки

reader.Close();

stream.Close();

Summarize(contents);

}

static void Summarize(char[] chars)

{

int lineCount = 1; // Начинаем с 1, так как первая строка уже существует

int vowelCount = 0; // Счетчик гласных

int consonantCount = 0; // Счетчик согласных

foreach (char c in chars)

{

if ("AEIOUaeiouАУОИЭЫЁЕЮЯауоиеэыёеяю".IndexOf(c) != -1)

{

// Если символ гласный

vowelCount++;

}

else if (char.IsLetter(c)) // Если это буква, но не гласная, значит согласная

{

consonantCount++;

}

if (c == '\n')

{

lineCount++; // Увеличиваем количество строк

}

}

Console.WriteLine($"Гласные = {vowelCount}, согласные = {consonantCount}, строк = {lineCount}");

}

}

//cd C:\Users\Nef0o0r\OneDrive\Study\Второй курс\ООП C#\Лабы\Laba5\Код\FileDetails\FileDetails\bin\Debug>

//FileDetails.exe text.txt

//Код работает только через консоль!!!

Упражнение 1 (24):

using System;

using System.IO;

class FileDetails

{

static void Main(string[] args)

{

try

{

int length;

string fileName;

// Вывод длины массива args

Console.WriteLine("Lengths of args: " + args.Length);

// Вывод всех аргументов

foreach (string arg in args)

{

Console.WriteLine(arg);

}

// Получение имени файла

fileName = args[0];

Console.WriteLine("Filename: " + fileName);

// Открываем файл и читаем его содержимое

FileStream stream = new FileStream(fileName, FileMode.Open);

StreamReader reader = new StreamReader(stream);

// Получаем длину файла

length = (int)stream.Length;

char[] contents = new char[length];

// Читаем символы по одному и добавляем их в массив

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

{

contents[i] = (char)reader.Read();

}

//Выводим символы в консоль

/\*foreach (char c in contents)

{

Console.WriteLine(c);

}\*/

// Закрываем потоки

reader.Close();

stream.Close();

Summarize(contents);

}

catch (Exception ex)

{

Console.WriteLine(ex.ToString());

}

}

static void Summarize(char[] chars)

{

int lineCount = 1; // Начинаем с 1, так как первая строка уже существует

int vowelCount = 0; // Счетчик гласных

int consonantCount = 0; // Счетчик согласных

foreach (char c in chars)

{

if ("AEIOUaeiouАУОИЭЫЁЕЮЯауоиеэыёеяю".IndexOf(c) != -1)

{

// Если символ гласный

vowelCount++;

}

else if (char.IsLetter(c)) // Если это буква, но не гласная, значит согласная

{

consonantCount++;

}

if (c == '\n')

{

lineCount++; // Увеличиваем количество строк

}

}

Console.WriteLine($"Гласные = {vowelCount}, согласные = {consonantCount}, строк = {lineCount}");

}

}

//cd C:\Users\Nef0o0r\OneDrive\Study\Второй курс\ООП C#\Лабы\Laba5\Код\FileDetails\FileDetails\bin\Debug>

//FileDetails.exe text.txt

//Код работает только через консоль!!!

Упражнение 2:

using System;

namespace Matrix

{

internal class Program

{

static void Main(string[] args)

{

int[,] arr1 = new int[2, 2];

int[,] arr2 = new int[2, 2];

int[,] result = new int[2, 2];

string line;

Console.WriteLine("Введите первую матрицу, каждый символ через Enter");

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 2; j++)

{

line = Console.ReadLine();

arr1[i, j] = int.Parse(line);

}

}

Console.WriteLine("Введите вторую матрицу, каждый символ через Enter");

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 2; j++)

{

line = Console.ReadLine();

arr2[i, j] = int.Parse(line);

}

}

Console.WriteLine("Первый массив:");

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 2; j++)

{

Console.Write(arr1[i, j] + " ");

}

Console.Write("\n");

}

Console.WriteLine("Вторая массив:");

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 2; j++)

{

Console.Write(arr2[i, j] + " ");

}

Console.Write("\n");

}

MatrixMultiply multiplier = new MatrixMultiply();

multiplier.MatrixMultiplys(arr1, arr2, ref result);

Console.WriteLine("Матрица полученная в результате умножения:");

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 2; j++)

{

Console.Write(result[i, j] + " ");

}

Console.Write("\n");

}

}

}

internal class MatrixMultiply

{

public void MatrixMultiplys(int[,] arr1, int[,] arr2, ref int[,] result)

{

for (int i = 0; i < arr1.GetLength(0); i++)

{

for (int j = 0; j < arr1.GetLength(1); j++)

{

result[i, j] = 0;

for (int k = 0; k < arr1.GetLength(0); k++)

result[i, j] += arr1[i, k] \* arr2[k, j];

}

}

}

}

}