

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

Институт (факультет) информатики и кибернетики

1. Кафедра программных систем

**ОТЧЁТ**

**по лабораторной работе**

Лабораторная работа № 03

Основы языка С#: Программы с использованием циклов -2по дисциплине «ОСНОВЫ ПРОГРАММИРОВАНИЯ»

Выполнил Фокин Е.А., 6103

Проверил Котенева С.Э.

Самара  
2023

ЗАДАНИЕ

Задание 0.

Прочитать теоретический материал.

Задание 1. «Ввод и обработка матриц»

Написать программу, осуществляющую ввод и обработку целочисленных квадратных матриц.

Размер матрицы не должен превышать 10. Размер матрицы пользователь вводил с экрана.

Ввод исходных матриц осуществляется с клавиатуры, вывод результатов обработки производится на консоль.

В программе должны выполняться действия над матрицами:

* сложение;
* вычитание;
* умножение;
* умножение на число;
* сравнить на равенство.

При возникновении ошибок должны выводиться соответствующие сообщения.

Текст и формат сообщений придумайте самостоятельно.

Задание 2. «Перевод из двоичной системы счисления в десятичную»

Привести число в десятичной системе счисления в двоичную систему счисления.

Вывести оба значения на экран.

Поменять местами первую и третью триады.

Перевести новое число в десятичную систему счисления.

На экран вывести новое число в двоичной и десятичной системах счисления.

Задание 3.

Подготовить отчет о работе.

КОД ПРОГРАММЫ

Console.WriteLine("Лабораторная работа 3. Выполнил студент 6103 Фокин Евгений\n");

var mainConsoleMenu = new ConsoleMenu();

mainConsoleMenu.Run(new Dictionary<string, Action>{

{"Ввод и обработка матриц", () => {

var firstMatrix = InputHelpers.ReadIntMatrixWithMaxSize(10);

var menu = new ConsoleMenu(1);

menu.Run(new Dictionary<string, Action>{

{"Сложение матриц", () => {

var secondMatrix = InputHelpers.ReadIntMatrixWithMaxSize(10);

Console.WriteLine("\tСумма равна:");

Matrix.Print(Matrix.Add(firstMatrix, secondMatrix), 1);

}},

{"Вычитание матриц", () => {

var secondMatrix = InputHelpers.ReadIntMatrixWithMaxSize(10);

Console.WriteLine("\tРазность равна:");

Matrix.Print(Matrix.Subtract(firstMatrix, secondMatrix), 1);

}},

{"Умножение матриц", () => {

var secondMatrix = InputHelpers.ReadIntMatrixWithMaxSize(10);

Console.WriteLine("\tПроизведение равно:");

Matrix.Print(Matrix.Multiply(firstMatrix, secondMatrix), 1);

}},

{"Умножение матрицы на число", () => {

double num = InputHelpers.ReadDouble("Введите число, на которое умножить матрицу");

Console.WriteLine("\tПроизведение равно:");

Matrix.Print(Matrix.MultiplyByNum(firstMatrix, num), 1);

}},

{"Сравнение матриц", () => {

var secondMatrix = InputHelpers.ReadIntMatrixWithMaxSize(10);

bool isEqual = Matrix.Equals(firstMatrix, secondMatrix);

if (isEqual)

Console.WriteLine("\tМатрицы равны");

else

Console.WriteLine("\tМатрицы не равны");

}},

});

}},

{"Перевод из двоичной системы счисления в десятичную", () => {

int num = InputHelpers.ReadInt("Введите число для обработки");

string binNum = "";

if (num == 0)

binNum = "0";

else {

while (num > 0) {

binNum += num % 2;

num /= 2;

}

binNum = string.Join("", binNum.Reverse().ToArray());

}

Console.WriteLine($"Число в десятичной c/c = {num}; Число в двоичной c/c = {binNum}");

if (binNum.Length < 9)

throw new Exception("В двоичной записи числа нет третей триады");

var size = binNum.Length;

var triad1 = binNum[(size - 3)..size];

var triad2 = binNum[(size - 6)..(size - 3)];

var triad3 = binNum[(size - 9)..(size - 6)];

var swappedNum = binNum[0..(size - 9)] + triad1 + triad2 + triad2;

int decimalValue = 0;

for (int i = swappedNum.Length - 1; i >= 0; i--)

{

int bitValue = swappedNum[i] - '0';

int positionValue = (int)Math.Pow(2, swappedNum.Length - 1 - i);

decimalValue += bitValue \* positionValue;

}

Console.WriteLine("Число в двоичной c/c после смены триад = " + swappedNum);

Console.WriteLine("Число в десятичной c/c после смены триад = " + decimalValue);

}},

});

public class ConsoleMenu

{

private bool isContinue = false;

private Action? lastAction = null;

private readonly int tabLevel;

public ConsoleMenu(int tabLevel = 0)

{

this.tabLevel = tabLevel;

}

public void Run(Dictionary<string, Action> menuActions)

{

while (true)

{

while (isContinue)

{

var exitMessage = tabLevel == 0 ? "полностью" : "в предыдущее меню";

WriteLine($"Повторить? (1 - да, 2 - выйти в нынышнее меню, 0 - выйти {exitMessage})");

var result = Console.ReadLine()?.Trim();

if (result == "1")

{

try

{

lastAction?.Invoke();

}

catch (Exception error)

{

WriteLine("Произошла ошибка: " + error.Message);

}

}

else if (result == "0")

return;

else

{

lastAction = null;

isContinue = false;

}

}

isContinue = true;

WriteLine("Выберите один из пунктов меню:");

WriteLine("0 - Выход");

for (int index = 0; index < menuActions.Count; index++)

{

var item = menuActions.ElementAt(index);

WriteLine($"{index + 1} - {item.Key}");

}

string? solution = Console.ReadLine()?.Trim();

bool parsed = int.TryParse(solution, out int solutionNum);

if (!parsed || 0 > solutionNum || solutionNum > menuActions.Count)

{

WriteLine("Не выбран ни один пункт меню\n");

isContinue = false;

}

else if (solutionNum == 0)

return;

else

lastAction = menuActions.ElementAt(solutionNum - 1).Value;

try

{

lastAction?.Invoke();

}

catch (Exception error)

{

WriteLine("Произошла ошибка: " + error.Message);

}

}

}

public void WriteLine(string value)

{

Console.WriteLine("".PadLeft(tabLevel \* 8) + value);

}

}

public class Matrix

{

public static int[,] Add(int[,] matrix1, int[,] matrix2)

{

int rows1 = matrix1.GetLength(0);

int cols1 = matrix1.GetLength(1);

int rows2 = matrix2.GetLength(0);

int cols2 = matrix2.GetLength(1);

if (rows1 != rows2 || cols1 != cols2)

throw new Exception("Матрицы должны иметь одинаковую размерность");

int[,] result = new int[rows1, cols1];

for (int i = 0; i < rows1; i++)

{

for (int j = 0; j < cols1; j++)

{

result[i, j] = matrix1[i, j] + matrix2[i, j];

}

}

return result;

}

public static int[,] Subtract(int[,] matrix1, int[,] matrix2)

{

int rows1 = matrix1.GetLength(0);

int cols1 = matrix1.GetLength(1);

int rows2 = matrix2.GetLength(0);

int cols2 = matrix2.GetLength(1);

if (rows1 != rows2 || cols1 != cols2)

throw new Exception("Матрицы должны иметь одинаковую размерность");

int[,] result = new int[rows1, cols1];

for (int i = 0; i < rows1; i++)

{

for (int j = 0; j < cols1; j++)

{

result[i, j] = matrix1[i, j] - matrix2[i, j];

}

}

return result;

}

public static int[,] Multiply(int[,] matrix1, int[,] matrix2)

{

int rows1 = matrix1.GetLength(0);

int cols1 = matrix1.GetLength(1);

int rows2 = matrix2.GetLength(0);

int cols2 = matrix2.GetLength(1);

if (rows1 != rows2 || cols1 != cols2)

throw new Exception("Матрицы должны иметь одинаковую размерность");

int[,] result = new int[rows1, cols2];

for (int i = 0; i < rows1; i++)

{

for (int j = 0; j < cols2; j++)

{

result[i, j] = 0;

for (int k = 0; k < cols1; k++)

{

result[i, j] += matrix1[i, k] \* matrix2[k, j];

}

}

}

return result;

}

public static double[,] MultiplyByNum(int[,] matrix, double num)

{

int rows = matrix.GetLength(0);

int cols = matrix.GetLength(1);

double[,] result = new double[rows, cols];

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

{

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

{

result[i, j] = matrix[i, j] \* num;

}

}

return result;

}

public static bool Equals(int[,] matrix1, int[,] matrix2)

{

int rows1 = matrix1.GetLength(0);

int cols1 = matrix1.GetLength(1);

int rows2 = matrix2.GetLength(0);

int cols2 = matrix2.GetLength(1);

if (rows1 != rows2 || cols1 != cols2)

{

return false;

}

for (int i = 0; i < rows1; i++)

{

for (int j = 0; j < cols1; j++)

{

if (matrix1[i, j] != matrix2[i, j])

{

return false;

}

}

}

return true;

}

public static int[,] Transpose(int[,] matrix)

{

int rows = matrix.GetLength(0);

int cols = matrix.GetLength(1);

int[,] result = new int[cols, rows];

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

{

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

{

result[j, i] = matrix[i, j];

}

}

return result;

}

public static void Print<T>(T[,] matrix, int tabLevel = 0)

{

int rows = matrix.GetLength(0);

int cols = matrix.GetLength(1);

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

{

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

{

Console.Write("".PadLeft(tabLevel \* 8) + matrix[i, j] + "\t");

}

Console.WriteLine();

}

}

}

public class InputHelpers

{

public static double ReadDouble(string message)

{

Console.WriteLine(message);

bool parsed = double.TryParse(Console.ReadLine(), out double value);

if (!parsed)

{

throw new Exception("Вы ввели некорректное значение");

}

return value;

}

public static int ReadInt(string message)

{

Console.WriteLine(message);

bool parsed = int.TryParse(Console.ReadLine(), out int value);

if (!parsed)

{

throw new Exception("Вы ввели некорректное значение");

}

return value;

}

public static int[,] ReadIntMatrixWithMaxSize(int maxSize)

{

var matrixSize = ReadInt("Введите размерность матрицы");

if (matrixSize < 1)

throw new Exception("Размер матрицы должен быть больше 0");

if (matrixSize > maxSize)

throw new Exception("Размер матрицы должен быть меньше " + maxSize);

Console.WriteLine("Введите строки матрицы через пробел:");

return ReadIntMatrix(matrixSize, matrixSize);

}

public static int[,] ReadIntMatrix(int rows, int cols)

{

int[,] matrix = new int[rows, cols];

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

{

Console.Write($"{i + 1} строка: ");

var matrixRow = Console.ReadLine()!.Trim();

var values = matrixRow.Split();

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

{

if (j < values.Length && int.TryParse(values[j], out matrix[i, j])) { }

else

matrix[i, j] = 0;

}

}

return matrix;

}

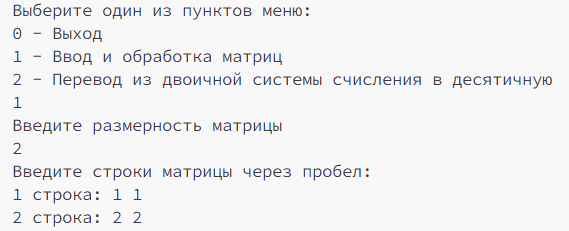
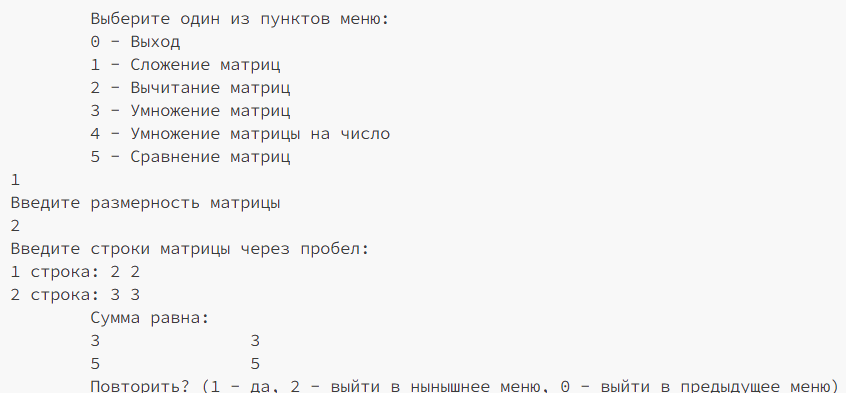
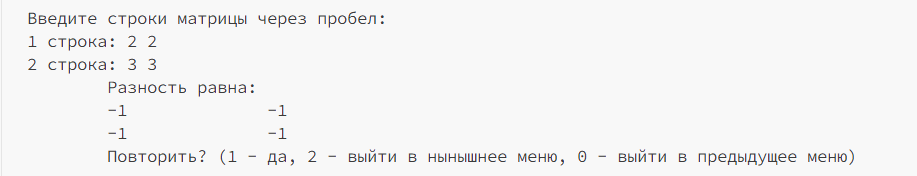
}

Рисунок 1 - Ввод матриц

Рисунок 2 – Действия над матрицами (сложение)

Рисунок 3 – Действия над матрицами (вычитание)

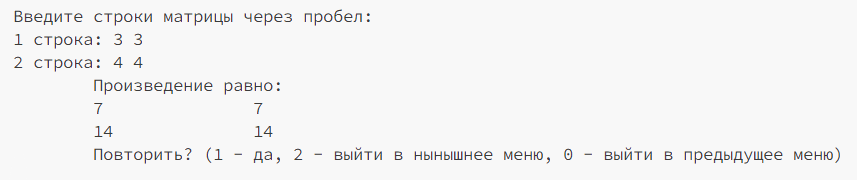


Рисунок 4 – Действия над матрицами (умножение)

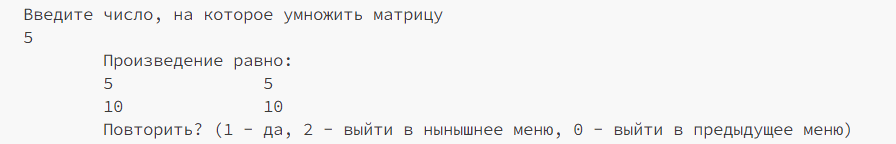


Рисунок 5 – Действия над матрицами (умножение на число)

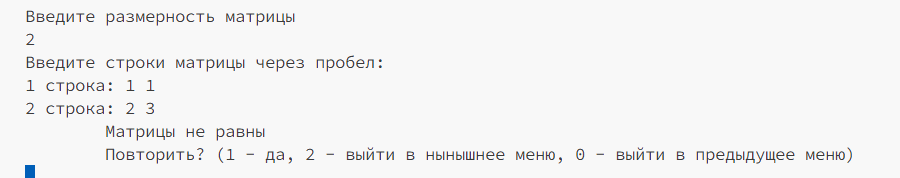


Рисунок 6 – Действия над матрицами (сравнение)

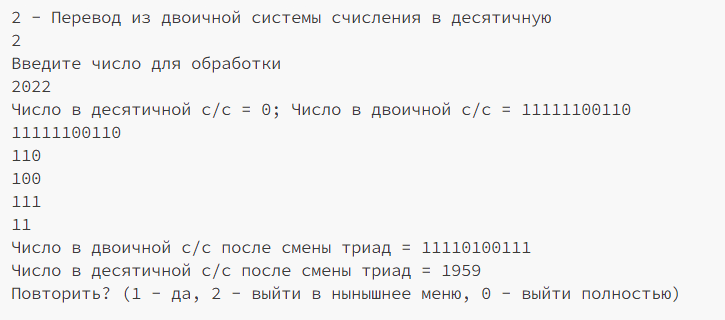


Рисунок 7 - Системы счисления и смена триад

ВЫВОДЫ

В лабораторной работе были использованы возможности языка C#:

* цикл while;
* двумерные массивы;
* цикл for;
* ветвления if;
* методы;
* классы;
* лямбда-выражения.

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: 04.10.23).