Отчет по практической работе №15

Наименование работы: многопоточность.

Индивидуальные задания (вариант №9):

1. Создайте консольное приложение с использованием MS Visual Studio.

Реализуйте метод для запуска в отдельном потоке (в соответствии с индивидуальным вариантом). Создайте делегат для представления метода (если требуется).

В основной программе (функция Main( )) реализуйте создание массива потоков (размер определите самостоятельно). Затем запустите все элементы массива (потоки) на выполнение.

Метод, выполняющийся в параллельных потоках должен выводить информацию о ходе своего выполнения в консоль приложения.

|  |  |
| --- | --- |
| 9 | Метод вычисления статистики (количество для каждого входящего символа) по строке. |

1. Создать приложение, запускающее три дочерних потока. Каждый поток выполняет вывод сообщения о том, что он создан, а затем выводит цифры своего порядкового номера. Для первого потока приоритет не изменять, для второго повысить на 1, для третьего повысить на 2. Вывести значение приоритетов потоков в порядке его убывания.
2. Создайте консольное приложение в среде MS Visual Studio.

С помощью параметризированных потоков выполните решение задачи в соответствии с индивидуальным вариантом.

|  |  |  |
| --- | --- | --- |
| Вариант | Решаемая задача (результат метода) | Входные параметры |
| 9 | Метод находит подмножество четных элементов матрицы случайных чисел | Два параметра: размер матрицы |

Ход работы:

1. Разработал программу в соответствии с заданием №1. Были реализованы методы, делегаты, события, потоки.

Листинг программы:

using System;

using System.Linq;

using System.Threading;

namespace Lab15.\_1

{

class Program

{

static char pos;

static string str;

static void Main(string[] args)

{

Console.Write("Введите строку: ");

str = Console.ReadLine();

Console.Write("Введите искомое в строке: ");

pos = Convert.ToChar(Console.ReadLine());

Thread[] td = new Thread[1];

for(int i = 0; i<1; i++)

{

td[i] = new Thread(Method);

td[i].Start();

}

}

static void Method()

{

int count = 0;

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

{

if(str[i] == pos)

{

count++;

}

}

if(count != 2 && count != 3 && count != 4)

Console.WriteLine($"В веденной строке символ {pos} встречается {count} раз");

else

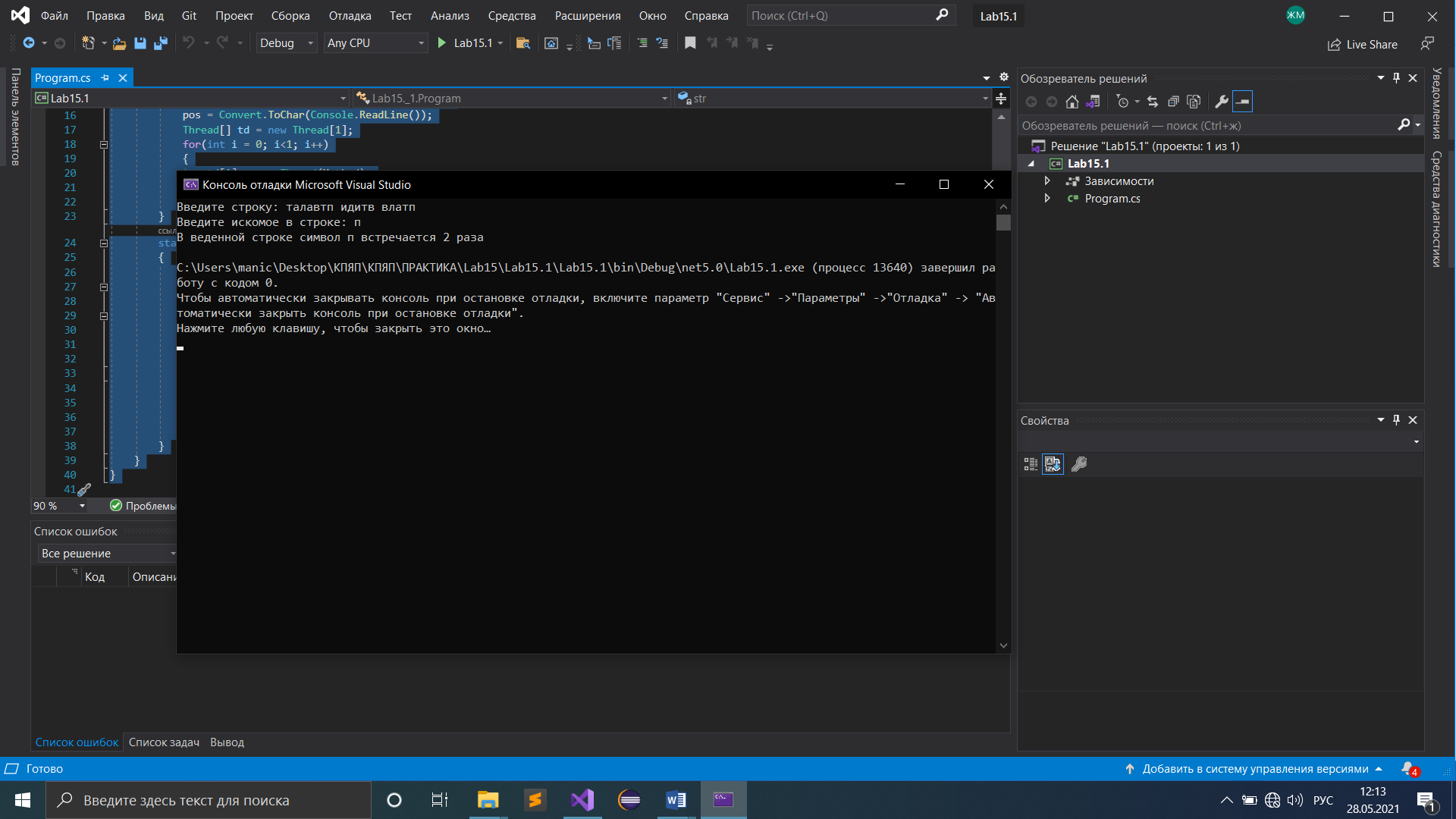
Console.WriteLine($"В веденной строке символ {pos} встречается {count} разa");

}

}

}

Результат выполнения программы Рис 2.1.



**Рисунок 2.1 – Пример работы программы c потоками**

1. Разработал программу в соответствии с заданием №1. Были реализованы методы, делегаты, события, потоки.

Листинг программы:

using System;

using System.Threading;

using System.Linq;

namespace Lab15.\_2

{

class Program

{

public delegate void Threading(string message);

static event Threading Trids;

static Thread[] td = new Thread[3];

static void Main(string[] args)

{

Trids += Mess;

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

{

td[i] = new Thread(Method);

td[i].Priority = ThreadPriority.Normal+i;

Trids?.Invoke($"Поток №{i + 1} запущен с приоритетом {td[i].Priority}!");

}

td[0].Name = "Valera";

td[1].Name = "Valero4ka";

td[2].Name = "Valeron4ik";

Trids -= Mess;

Trids += Mess2;

Console.ForegroundColor = ConsoleColor.Blue;

Console.WriteLine("Отсортированные потоки!");

Console.ResetColor();

Array.Reverse(td);

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

{

td[i].Start();

Trids?.Invoke($"Поток {td[i].Name} приоритет {td[i].Priority}");

}

}

static void Method()

{

}

static void Mess(string message)

{

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine(message);

Console.ResetColor();

}

static void Mess2(string message)

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine(message);

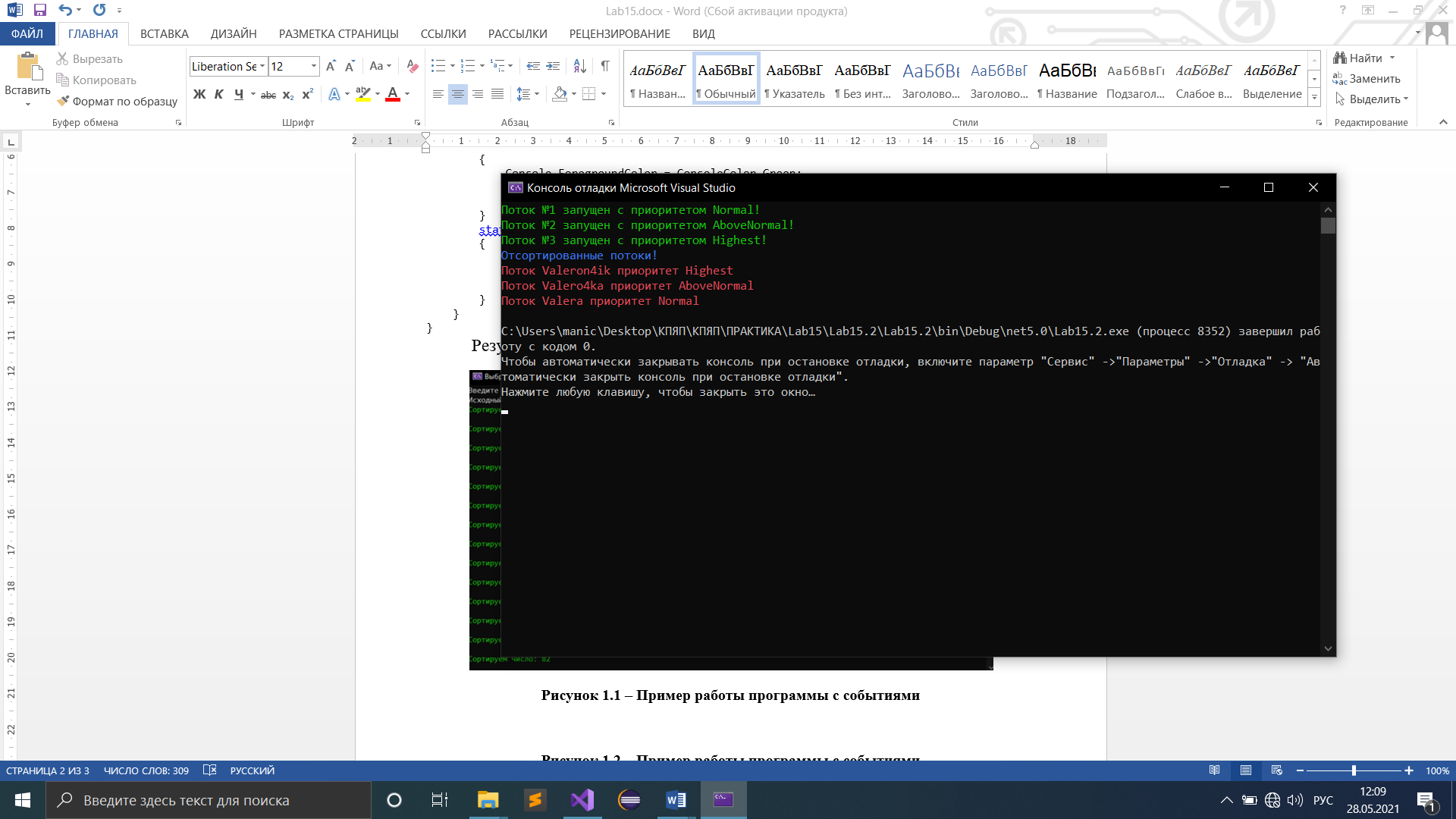
Console.ResetColor();

}

}

}

Результат выполнения программы Рис 2.1.



**Рисунок 2.1 – Пример работы программы c потоками**

1. Разработал программу в соответствии с заданием №1. Были реализованы методы, потоки.

Листинг программы:

using System;

using System.Threading;

namespace Lab15.\_3

{

class Program

{

static Thread td = new Thread(new ParameterizedThreadStart(Matr));

static void Main(string[] args)

{

Zna zn = new Zna();

Console.Write("Введите длину матрицы: ");

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

Console.Write("Введите ширину матрицы: ");

zn.m = int.Parse(Console.ReadLine());

td.Start(zn);

}

static void Matr(object obj)

{

Zna c = (Zna)obj;

int pod = 1;

int[,] arr = new int[c.n, c.m];

Random rnd = new Random();

for (int i = 0; i < c.n; i++)

{

for (int j = 0; j < c.m; j++)

{

arr[i, j] = rnd.Next(-100, 100);

}

}

Console.WriteLine("Исходная мтарица: ");

for (int i = 0; i < c.n; i++)

{

for (int j = 0; j < c.m; j++)

{

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

}

Console.WriteLine();

}

for (int i = 0; i < c.n; i++)

{

for (int j = 0; j < c.m; j++)

{

if(arr[i,j] % 2 == 0)

{

pod \*= arr[i, j];

}

}

Console.WriteLine();

}

Console.WriteLine($"Подмножество равно: {pod}");

}

}

class Zna

{

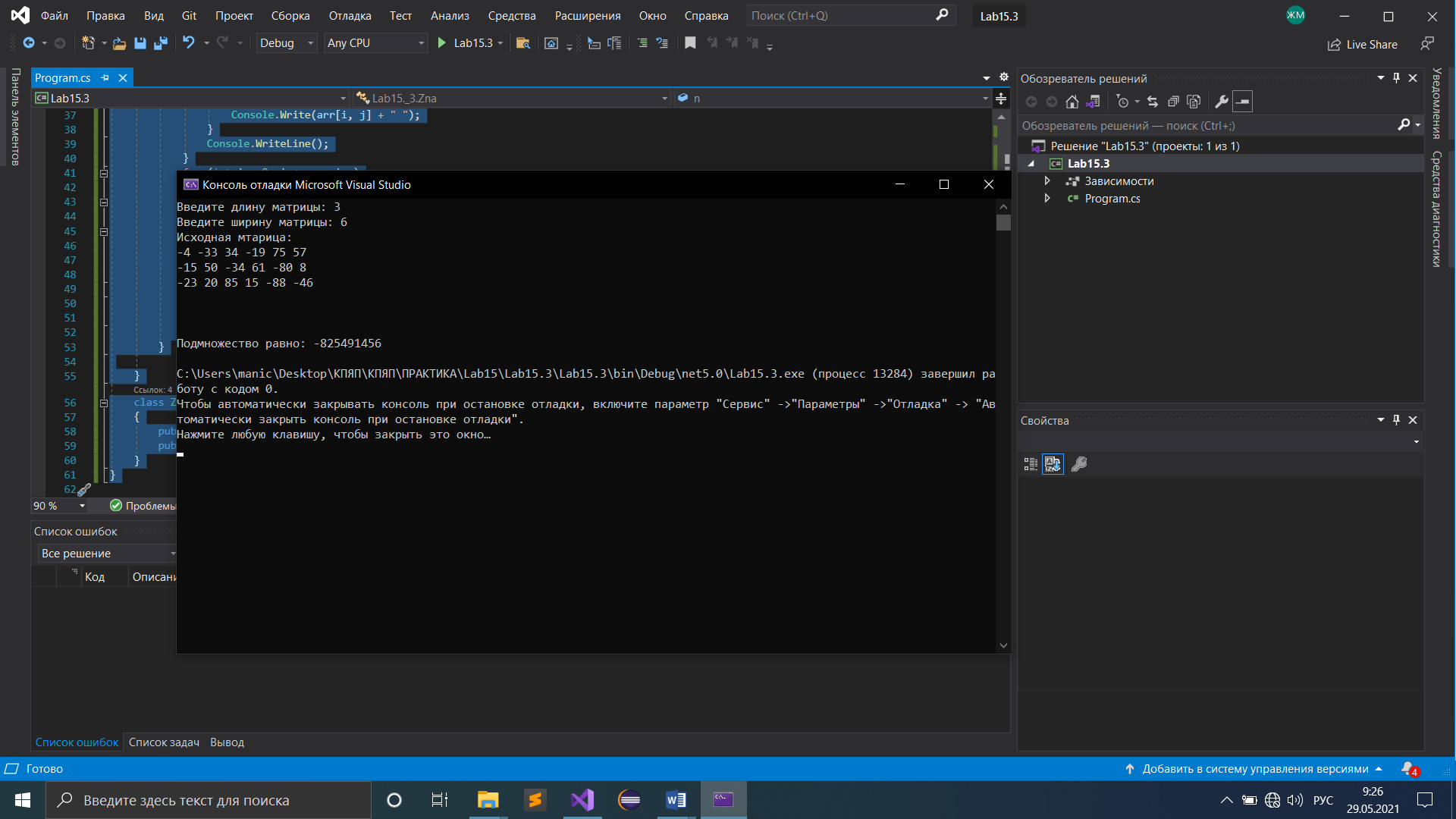
public int n;

public int m;

}

}

Результат выполнения программы Рис 3.1.



**Рисунок 3.1 – Пример работы программы c потоками**

Отметка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подпись преподавателя\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_