**3 ПРОЦЕДУРЫ И ФУНКЦИИ – МЕТОДЫ КЛАССА**

Задание 1. Разработать метод f(x), который возвращает младшую цифру натурального числа Х. Вычислить с помощью него значение выражения = f(a) + f(b).

Листинг программы:

class Program

{// возвращает функцию х деленную на 10

public static int F(int x)

{

return x % 10;

}

static void Main(string[] args)

{

// ввод числа и его конвертирование

Console.WriteLine("Введите число a");

int a = Convert.ToInt32(Console.ReadLine());

// ввод числа б и его конвертирование

Console.WriteLine("Введите число b");

int b = Convert.ToInt32(Console.ReadLine());

// сумма функции а и б

int z = F(a) + F(b);

// вывод результата на консоль

Console.WriteLine("Результат работы функции:" + z);

// задержка консоли

Console.ReadKey();

}

}

Таблица 3.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| a = 5  b = 10 | Результат = 5 |

Анализ результатов:

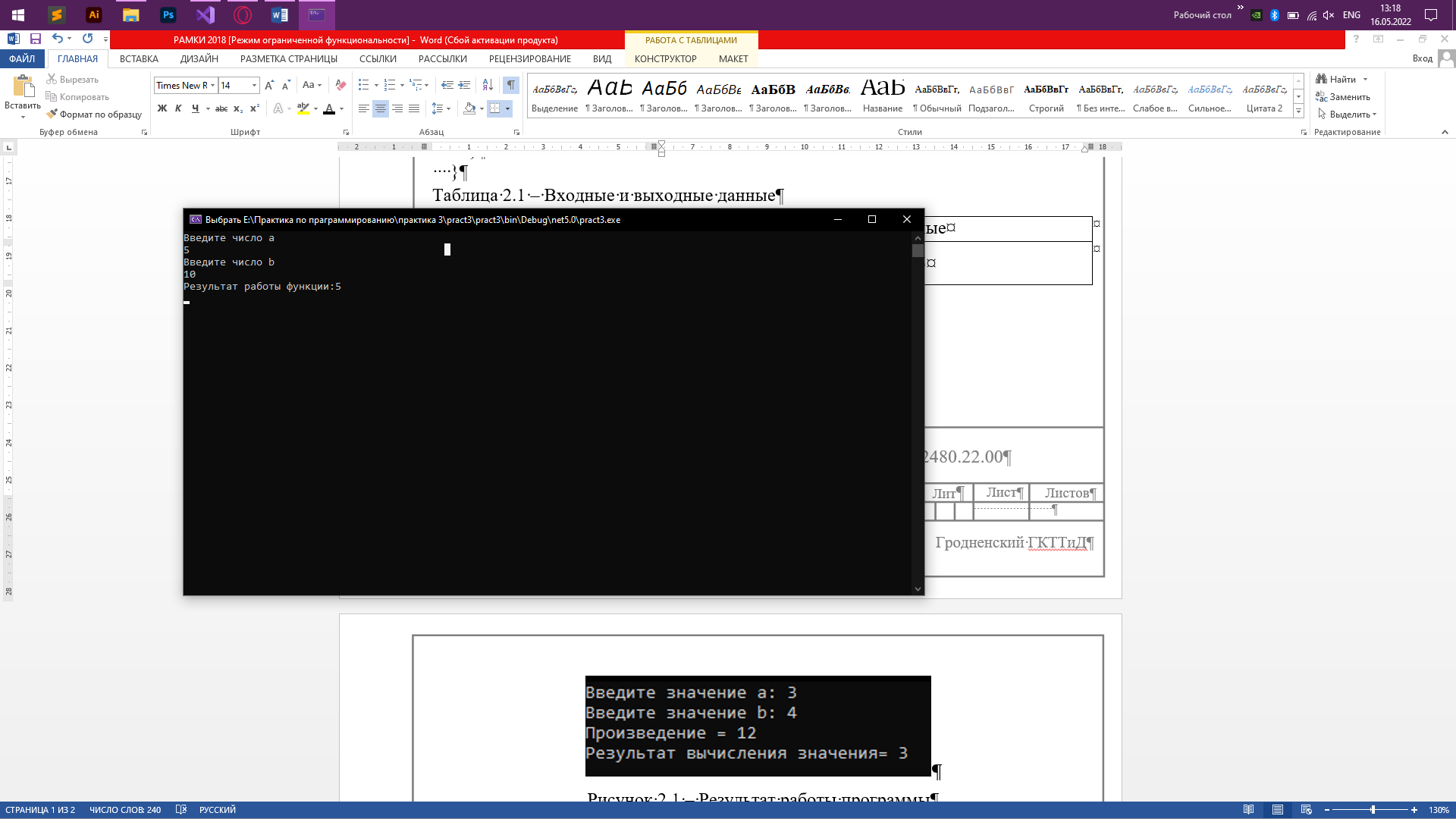
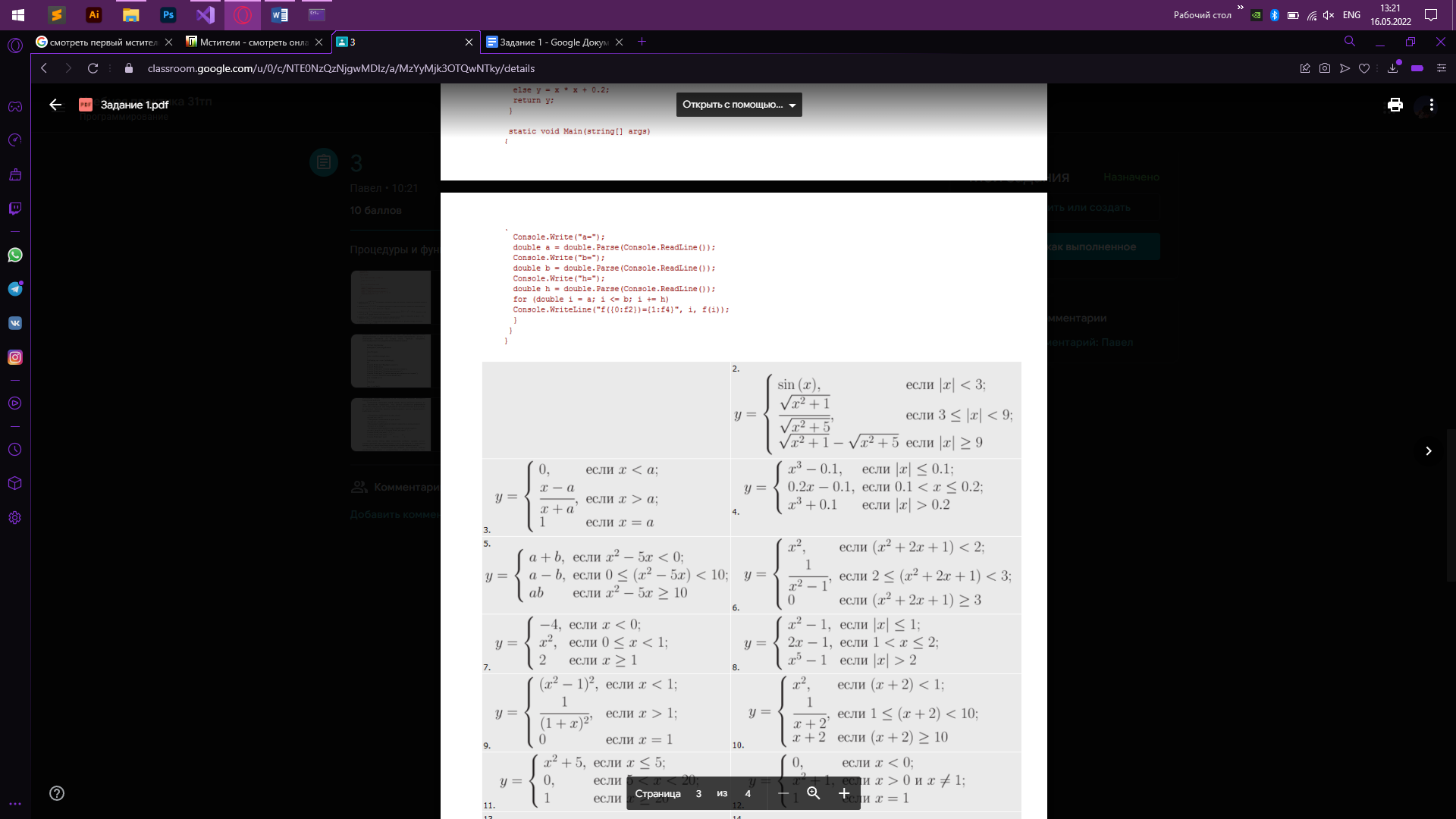


Рисунок 3.1 – Результат работы программы

Источник: собственная разработка

Задание 2. Постройте таблицу значений функции y = f(x) для xe [a, b с шагом h.



Листинг программы:

class Program

{

public static int f(int x)

{

// возвращает результат операций

return x \* x - 5 \* x;

}

static void Main(string[] args)

{

// ввод числа а и его конвертирование

Console.WriteLine("Введите число a");

int A = Convert.ToInt32(Console.ReadLine());

// ввод числа б и его конвертирование

Console.WriteLine("Введите число b");

int B = Convert.ToInt32(Console.ReadLine());

// ввод числа h и его конвертирование

Console.WriteLine("Введите число h");

int h = Convert.ToInt32(Console.ReadLine());

// число count равно 1

int count = 1;

// цикл, где считается функция, после этого происходит операция (сложение, вычитание, умножение)

for (int i = A; i <= B; i += h)

{

if (f(i) < 0)

{

Console.WriteLine("РЕЗУЛЬТАТ ИТЕРАЦИИ №" + count + "=" + (A + B));

}

else if (f(i) >= 0 && f(i) < 10)

{

Console.WriteLine("РЕЗУЛЬТАТ ИТЕРАЦИИ №" + count + "=" + (A - B));

}

else

{

Console.WriteLine("РЕЗУЛЬТАТ ИТЕРАЦИИ №" + count + "=" + (A \* B));

}

// увеличение на 1

count += 1;

}

// задержка консоли

Console.ReadLine();

}

}

Таблица 3.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Число a: 7  Число b: 9  Число h: 15 | РЕЗУЛЬТАТ ИТЕРАЦИИ №1=63 |

Анализ результатов:

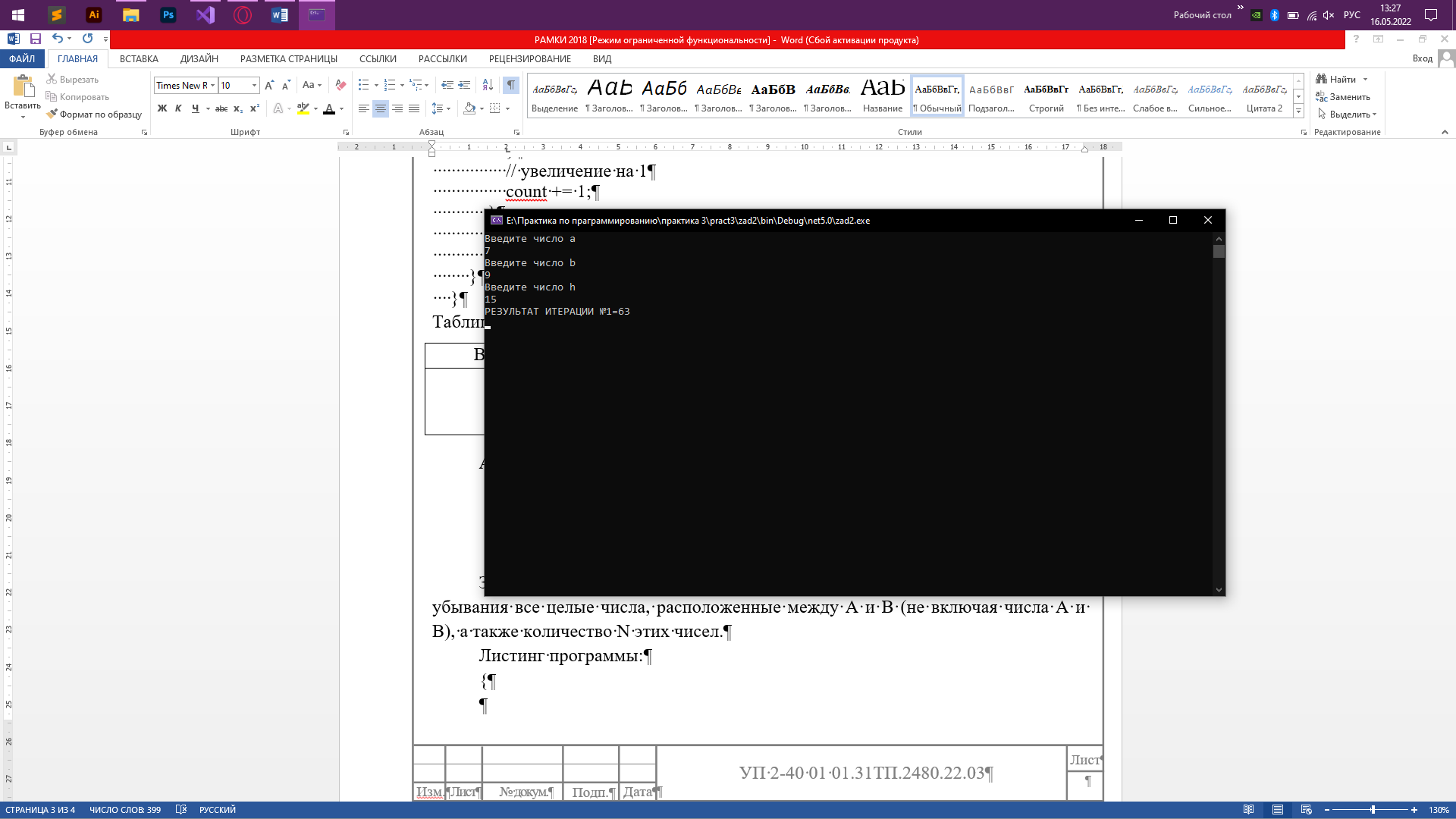


Рисунок 3.2 – Результат работы программы

Источник: собственная разработка

Задание 3. Перегрузите метод  из предыдущего раздела так, чтобы его сигнатура (заголовок) соответствовала виду зtatic void f (double x, out double y}. Продемонстрируйте работу перегруженных методов.

Листинг программы:

class Program

{

public static void f(double x, out double y)

{// решение примера и ввод у

y = x \* x - 5 \* x;

Console.WriteLine(y);

}

public static int f(int x)

{ // возвращает результат

return x \* 2 - 5 \* x;

}

static void Main(string[] args)

{// число z

double z;

f(5.3, out z);

// вывод на консоль

Console.WriteLine(f(10));

// задержка консоли

Console.ReadKey();

}

Таблица 3.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Число a: 7  Число b: 9  Число h: 15 | 1,5899999999999999  -30 |

Анализ результатов:

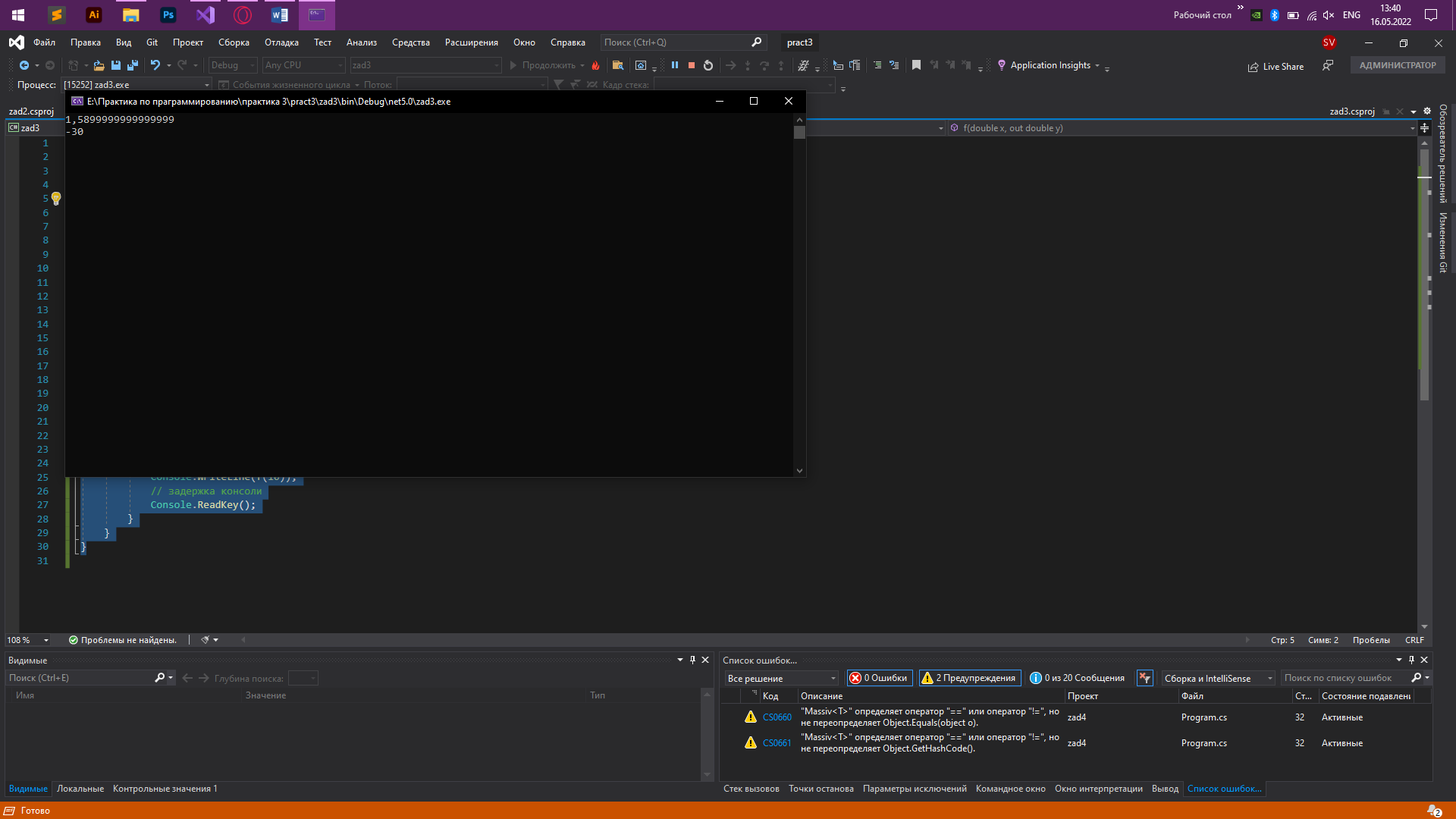


Рисунок 3.3 – Результат работы программы

Источник: собственная разработка

Задание 4. Описать класс для работы с двумерным массивом строк фиксированной длины. Обеспечить сравнение массивов на равенство (перегрузку операции == для поэлементного сравнения).

Листинг программы:

public class Programm

{

static void Main(string[] args)

{

var arrToFill = new string[] { "one", "two", "three", "four", "five", "six", "seven", "eight", "nine", "ten" };

var arr1 = new Massiv<string>(5, 5);

var arr2 = new Massiv<string>(5, 5);

arr1.Fill(arrToFill);

arr2.Fill(arrToFill/\*.OrderBy(x=>x).ToArray()\*/);

arr2[0, 0] = "KAWAII";

arr1.Print();

Console.WriteLine();

arr2.Print();

if (arr1 == arr2) Console.WriteLine("Arrays are equal!");

else Console.WriteLine("Arrays are not equal!");

Console.ReadLine();

}

}

public class Massiv<T>

{

private readonly T[,] \_arr;

public int RowCount { get; set; }

public int ColumnCount { get; set; }

public Massiv(int rows, int columns)

{

RowCount = rows;

ColumnCount = columns;

\_arr = new T[RowCount, ColumnCount];

}

public void Print(int coloredRow = -1, int coloredColumn = -1)

{

var maxLength = MaxLength();

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

{

Console.Write("|");

if (coloredRow == i) Console.ForegroundColor = ConsoleColor.Red;

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

{

if (coloredColumn == j) Console.ForegroundColor = ConsoleColor.Red;

var anz = maxLength - \_arr[i, j].ToString().Length;

var l = Spaces(anz);

if (coloredRow != i && coloredColumn != j) Console.ResetColor();

Console.Write(" " + \_arr[i, j] + l + " ");

}

Console.ResetColor();

Console.Write("|\n");

}

}

private static string Spaces(int count)

{

var result = "";

for (int i = 0; i < count; i++) result += " ";

return result;

}

private int MaxLength()

{

int max = int.MinValue;

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

{

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

{

if (\_arr[i, j].ToString().Length > max) max = \_arr[i, j].ToString().Length;

}

}

return max;

}

public void Fill(T[] toFill)

{

var rnd = new Random();

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

{

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

{

T value = toFill[rnd.Next(toFill.Length)];

\_arr[i, j] = value;

}

}

}

public static bool operator ==(Massiv<T> arr1, Massiv<T> arr2)

{

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

{

for (int j = 0; j < arr2.ColumnCount; j++)

{

if (!arr1.\_arr[i, j].Equals(arr2.\_arr[i, j])) return false;

}

}

return true;

}

public static bool operator !=(Massiv<T> arr1, Massiv<T> arr2)

{

return !(arr1 == arr2);

}

public T this[int row, int column]

{

get { return \_arr[row, column]; }

set { \_arr[row, column] = value; }

}

}

Таблица 3.4 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| "one", "two", "three", "four", "five", "six", | ten three one nine two |  | seven four three seven ten |  | eight two eight seven seven | |

Окончание таблицы 3.4

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| "seven", "eight", "nine", "ten","KAWAII". | | six two two six ten |  | two four four four five |  | KAWAII eight five two two |  | eight two nine seven three |  | eight eight three five one |  | two eight two six six |  | four one ten three three | |

Анализ результатов:

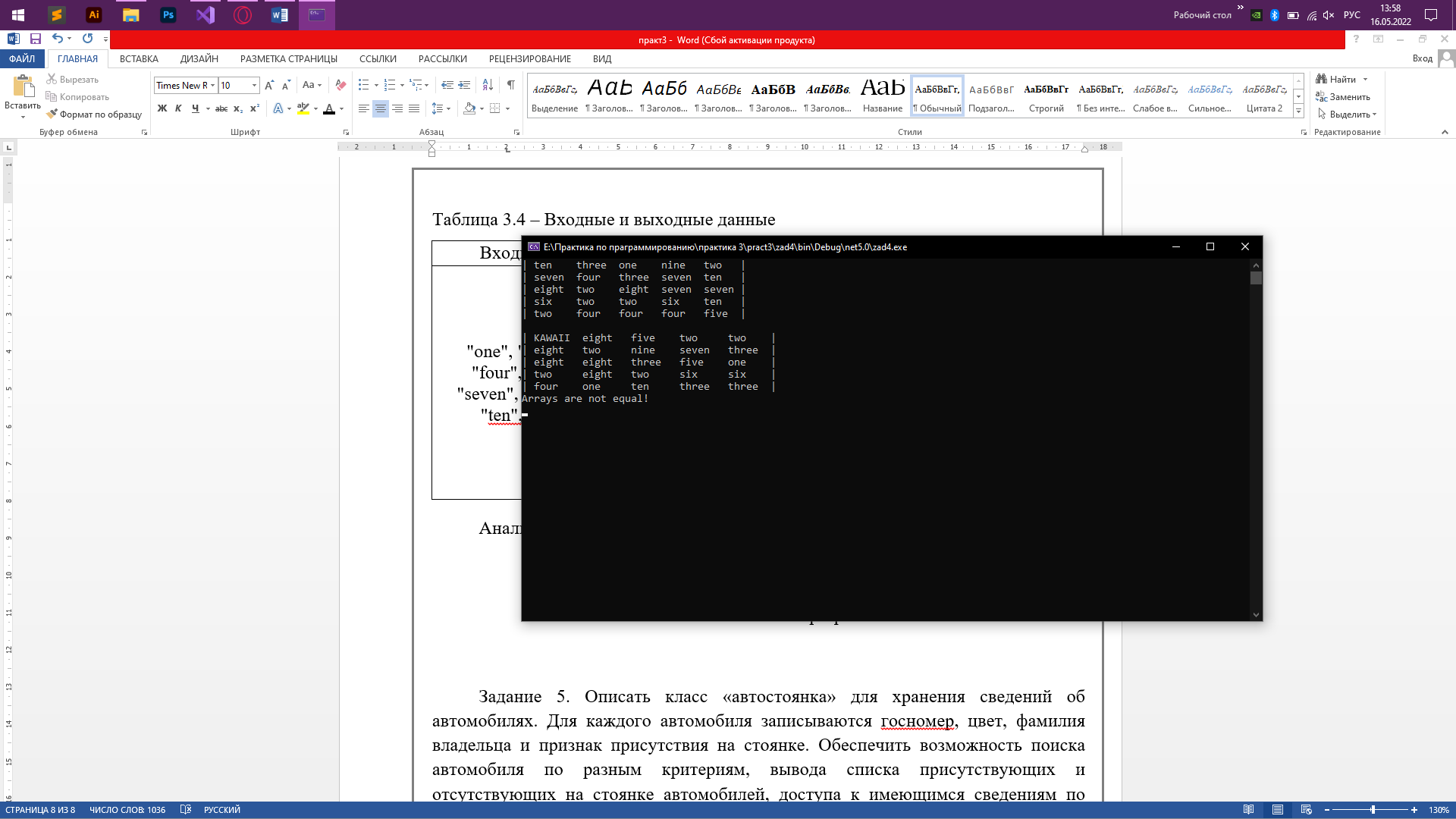


Рисунок 3.4 – Результат работы программы

Источник: собственная разработка

Задание 5. Описать класс «записная книжка». Предусмотреть возможность работы с произвольным числом записей, поиска записи по какому-либо признаку (например, по фамилии, дате рождения или номеру телефона), добавления и удаления записей, сортировки по фамилии и доступа к записи по номеру. Написать программу, демонстрирующую все разработанные элементы класса.

Листинг программы:

public class Record

{

public int id;

public string Name;

public string Surname;

public DateTime Birthday;

public string PhoneNumber;

public static bool brake = false;

public Record(int id, string Name, string Surname, DateTime Birthday, string PhoneNumber)

{

this.id = id;

this.Name = Name;

this.Surname = Surname;

this.Birthday = Birthday;

this.PhoneNumber = PhoneNumber;

}

}

public class Notebook

{

private List<Record> notebookRecords;

public List<Record> NotebookRecords

{

get { return notebookRecords; }

set { notebookRecords = value; }

}

public Notebook()

{

notebookRecords = new List<Record>();

}

public List<Record> SearchRecords(string Name, string Surname, DateTime? Birthday, string PhoneNumber)

{

List<Record> ret = new List<Record>();

foreach (Record rec in notebookRecords)

if (rec.Name == Name || rec.Surname == Surname || rec.Birthday == Birthday || rec.PhoneNumber == PhoneNumber)

ret.Add(rec);

return ret;

}

public void AddRecord(int id, string Name, string Surname, DateTime Birthday, string PhoneNumber)

{

id = Convert.ToInt32(Console.ReadLine());

Name = Convert.ToString(Console.ReadLine());

Birthday = Convert.ToDateTime(Console.ReadLine());

PhoneNumber = Convert.ToString(Console.ReadLine());

Record rec = new Record(id, Name, Surname, Birthday, PhoneNumber);

notebookRecords.Add(rec);

Console.WriteLine(Convert.ToString(id), Name, Surname, Birthday, PhoneNumber);

}

public void DeleteRecords(string Name, string Surname, DateTime Birthday, string PhoneNumber)

{

List<Record> records = SearchRecords(Name, Surname, Birthday, PhoneNumber);

foreach (Record rec in records)

notebookRecords.Remove(rec);

}

public void SortBySurname()

{

notebookRecords.OrderBy(x => x.Surname);

}

private void test()

{

SearchRecords(null, null, null, null); //вернёт все записи

}

}

class Program

{

static void Main(string[] args)

{

var needExit = false;

while (!needExit)

{

char resp = ReadMenuSelection("Выберите: [Q]=Выход, [A]=генерация случайного числа", new char[] { 'Q', 'A' });

switch (resp)

{

case 'Q':

Console.WriteLine("Завершение программы");

needExit = true;

break;

case 'A':

Console.WriteLine("Генерация случайного числа:");

Console.WriteLine(new Random().Next(100));

break;

default:

Console.WriteLine("Не предусмотренный ответ.");

break;

}

}

}

private static char ReadMenuSelection(string Mesage, char[] ValidResponces)

{

while (true)

{

Console.WriteLine(Mesage);

var key = Console.ReadKey();

if (ValidResponces.Contains(key.KeyChar))

{

Console.WriteLine();

return key.KeyChar;

}

else

{

Console.WriteLine("Неправильный ввод. Попробуйте исчо рас-рас.");

}

}

}

}

Таблица 3.5 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| public string Name; public string Surname; public DateTime Birthday; public string PhoneNumber | Генерация случайного числа: 72 |

Анализ результатов:

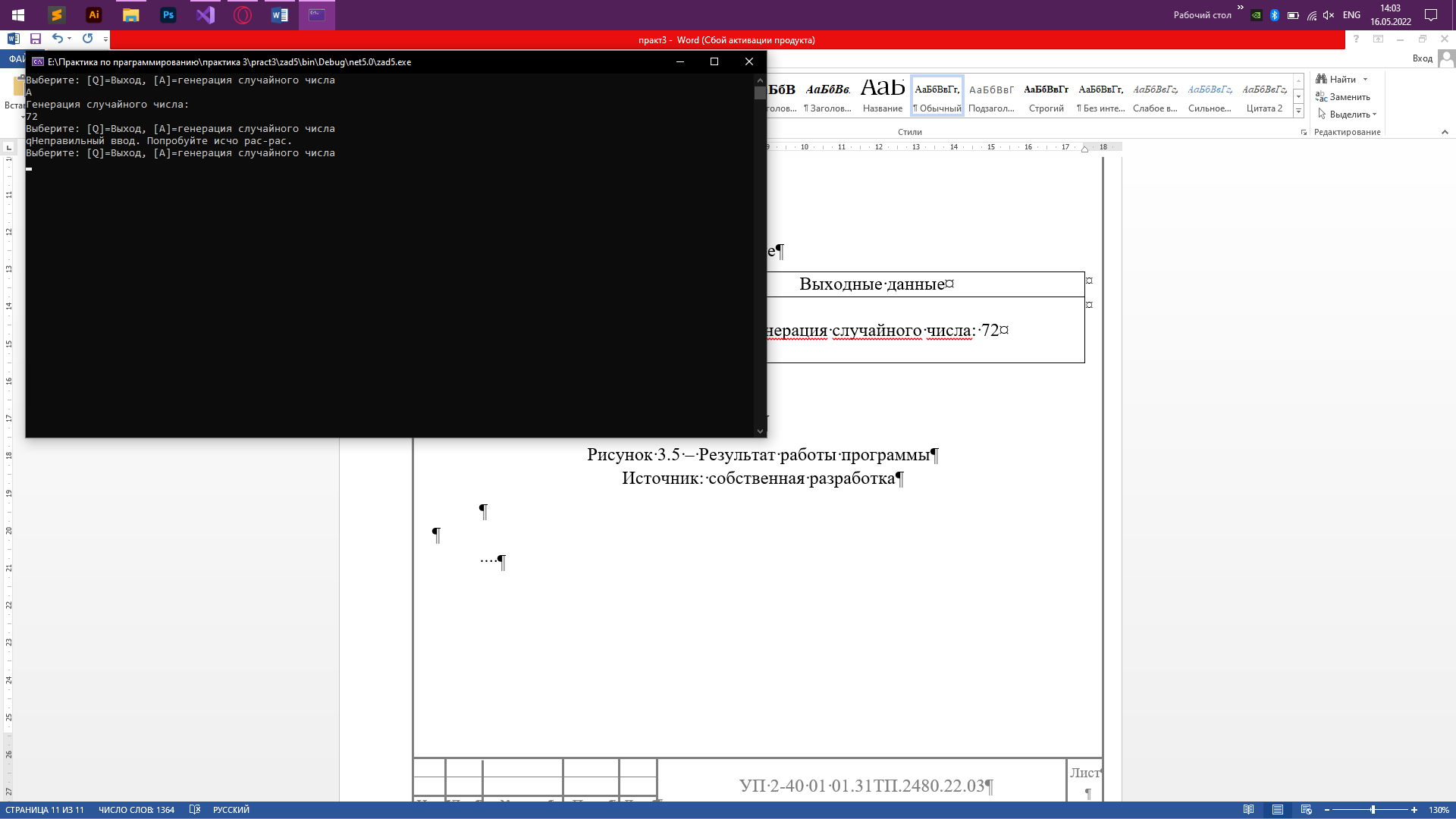


Рисунок 3.5 – Результат работы программы

Источник: собственная разработка