**3 Процедуры и функции – методы класса**

Задание 1. Разработать метод f(n,x), который для заданного натурального числа n и вещественного x находит значение выражения . Вычислить с помощью данного метода значение выражения

namespace task1

{

internal class Program

{

static void Main(string[] args)

{

Console.Write("Введите х: ");

double x = double.Parse(Console.ReadLine()); //Ввод х

Console.Write("Введите n: ");

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

double newX = Function(x, n); //Вызов методы

double result = Math.Pow(newX,2) / 2 + Math.Pow(newX, 4) / 4 + Math.Pow(newX, 6) / 6;

Console.WriteLine("Отвтет: " + Math.Round(result,2));

}

static double Function(double x, double n) //Метод который вычесляет функция

{

return Math.Pow(x,n)/n;

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 4, 3 | 15762909 |

Анализ результатов:

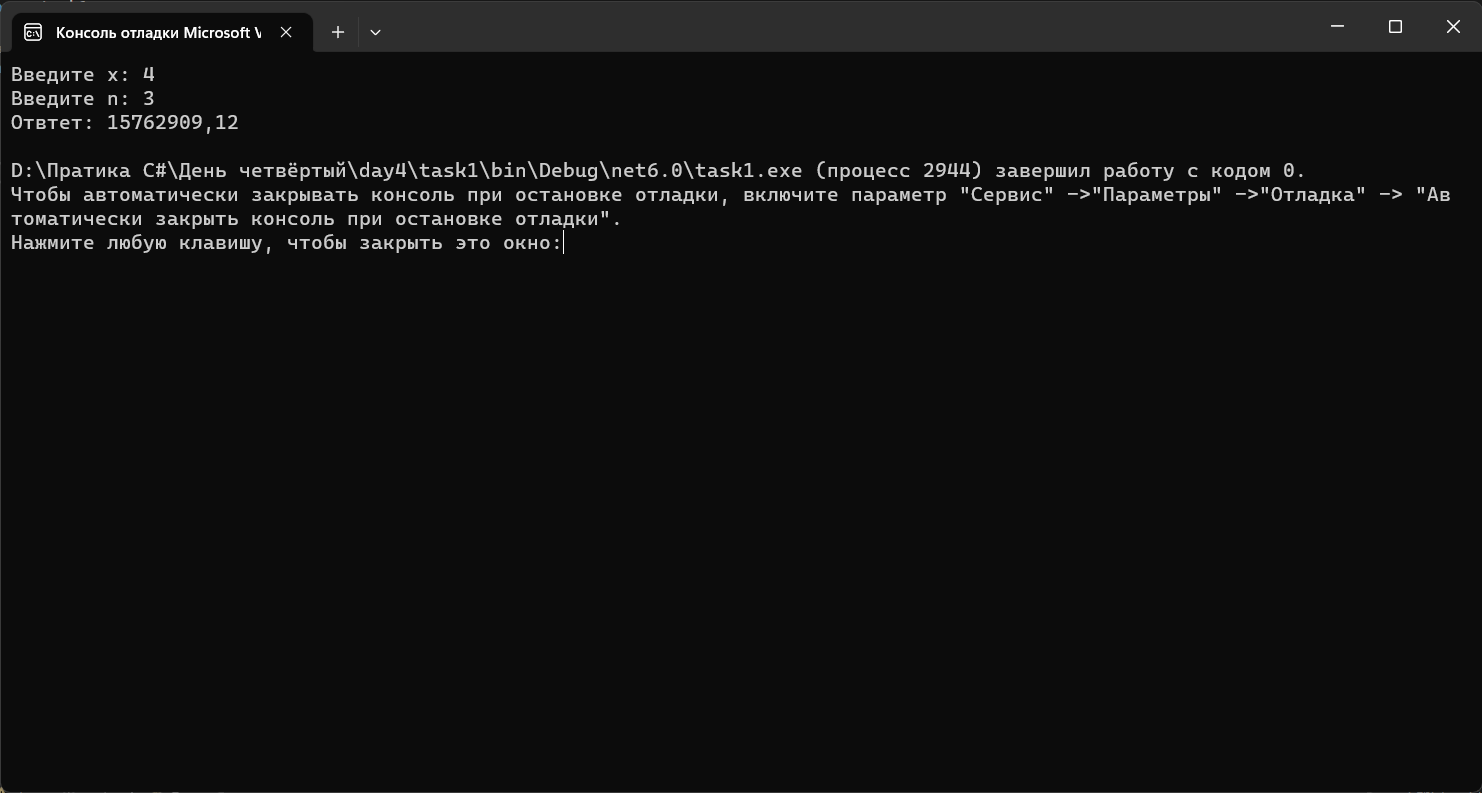


Рисунок 1.1 – Результат работы программы

Задание 2. Описать класс для работы с двумерным массивом вещественных

чисел. Обеспечить добавление к первому столбцу столбца заданного номера

(перегрузка операции +).

namespace task5

{

internal class Program

{

static void Main(string[] args)

{

Record first = new Record("Вова", "7153040", "11.04.2004");

Record second = new Record("Попа", "7159040", "20.04.2004");

Record third = new Record("Боба", "7532040", "13.10.2004");

Notebook notebook = new Notebook();

notebook.AddRecord(first);

notebook.AddRecord(second);

notebook.AddRecord(third);

Console.WriteLine("Найдём людей по 3 цифрам номера 715: ");

var peoples = notebook.GetRecordByPhone("715");

foreach(var people in peoples)

{

Console.WriteLine(people);

}

Console.WriteLine("--------");

Console.WriteLine("Найдём человека по номеру 7153040: ");

peoples = notebook.GetRecordByPhone("7153040");

foreach (var people in peoples)

{

Console.WriteLine(people);

}

Console.WriteLine("--------");

Console.WriteLine("Найдём человека по дню рождению сегодня: ");

peoples = notebook.GetRecordByBirthday();

foreach (var people in peoples)

{

Console.WriteLine(people);

}

Console.WriteLine("--------");

Console.WriteLine("Найдём человека по дню рождения 13.10.2004: ");

peoples = notebook.GetRecordByBirthday("13.10.2004");

foreach (var people in peoples)

{

Console.WriteLine(people);

}

Console.WriteLine("--------");

Console.WriteLine("Найдём человека по дню рождения на следующей неделе (сегодня - 11.04.2023): ");

peoples = notebook.GetRecordByBirthdayOnNextWeek();

foreach (var people in peoples)

{

Console.WriteLine(people);

}

Console.WriteLine("--------");

}

}

namespace task5

{

internal class Notebook

{

private List<Record> \_records;

public Notebook()

{

\_records = new List<Record>();

}

public void AddRecord(Record record)

{

\_records.Add(record);

}

public void DeleteRecord(Record record)

{

\_records.Remove(record);

}

public List<Record> GetRecordByPhone(string phoneNumber)

{

List<Record> finded = \_records.Where(x => x.PhoneNumber.StartsWith(phoneNumber)).ToList();

if (finded.Count == 0)

{

throw new Exception("Не найден человек с таким номером");

}

return finded;

}

public List<Record> GetRecordByBirthday(string date = "now")

{

if (date == "now")

{

return \_records.Where(x => x.Date.Month.Equals(DateTime.Today.Month) && x.Date.Day.Equals(DateTime.Today.Day)).ToList();

}

else

{

DateOnly parsedDate = DateOnly.Parse(date);

return \_records.Where(x => x.Date.Month.Equals(parsedDate.Month) && x.Date.Day.Equals(parsedDate.Day)).ToList();

}

}

public List<Record> GetRecordByBirthdayOnNextWeek()

{

DateTime sundayOfWeek = DateTime.Today.AddDays(-(int)DateTime.Today.DayOfWeek + 6);

return \_records.Where(x => x.Date.Month.Equals(sundayOfWeek.Month) && Enumerable.Range(sundayOfWeek.Day, 7).Contains(x.Date.Day)).ToList();

}

}

class Record

{

private string \_name;

private string \_phoneNumber;

private DateOnly \_date;

public string Name

{

get { return \_name; }

}

public string PhoneNumber

{

get { return \_phoneNumber; }

}

public DateOnly Date

{

get { return \_date; }

}

public Record(string name, string phoneNumber, string date)

{

\_name = name;

\_phoneNumber = phoneNumber;

\_date = DateOnly.Parse(date);

}

public override string ToString()

{

return $"Имя: {\_name}, Номер телефона: {\_phoneNumber}, Дата рождения: {\_date}";

}

}

Таблица 1.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Данные людей | Обработанные данные |

Анализ результатов:

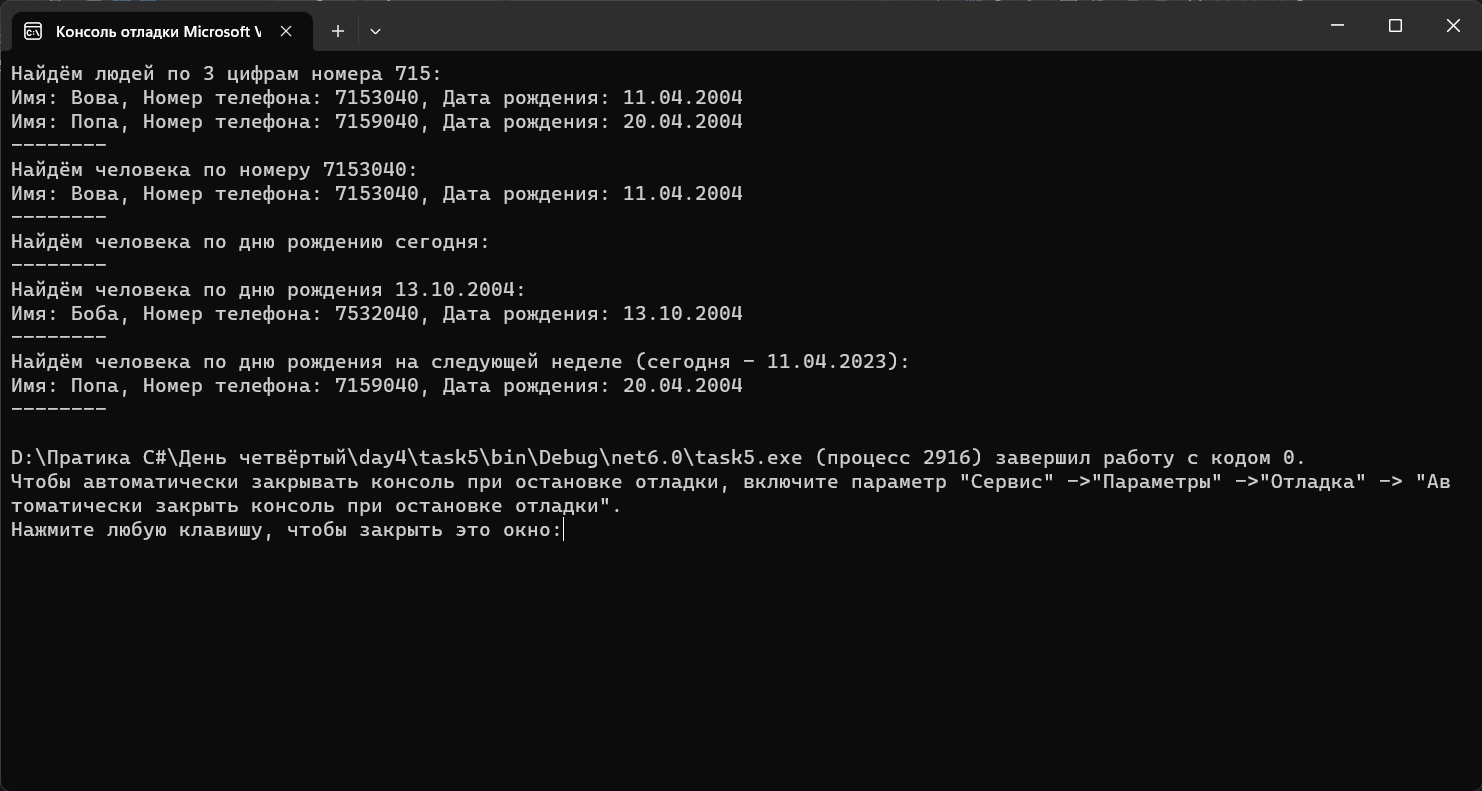
**

Рисунок 1.2 – Результат работы программы

работы программы