**НЕГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ ЧАСТНОЕ** **УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ** **«МОСКОВСКИЙ ФИНАНСОВО-ПРОМЫШЛЕННЫЙ УНИВЕРСИТЕТ** **“СИНЕРГИЯ”»**

|  |  |  |
| --- | --- | --- |
| **Факультет/Институт** |  | Информационных технологий |
|  |  | (наименование факультета/ Института) |
| **Направление/специальность** |  | ИСиТ |
| **подготовки:** |  | (код и наименование направления /специальности подготовки) |
| **Форма обучения:** |  | Очная |
|  |  | (очная, очно-заочная, заочная) |
|  |  |  |

**Отчет по лабораторной работе №3**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **на тему** |  | Лабораторная работа №3 «Функции и рекурсия».  [2018]Беспалов\_Основы\_Алгоритмизации\_CPP  Лабораторная работа №1. Рекурсивные функции | | |
|  |  | (наименование темы) | | |
|  |  |  | | |
| **по дисциплине** | | |  | Разработка модулей обеспечения для компьютерных систем |
|  | | |  | (наименование дисциплины) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Обучающийся** |  | Истомин М.А |  |  |
|  |  | (ФИО) |  | (подпись) |
| **Группа** |  | ДКИП 206-прог |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Преподаватель** |  | Сибирев И.В |  |  |
|  |  | (ФИО) |  | (подпись) |

**Москва 2024 г.**

**Лабораторная работа №.3 «Функции и рекурсия»**

Лабораторная работа 4. Простейшие классы Теоретический материал: глава 4, раздел «Обработка исключительных ситуаций», глава 5. Каждый разрабатываемый класс должен, как правило, содержать следующие элементы: скрытые поля, конструкторы с параметрами и без параметров, методы, свойства. Методы и свойства должны обеспечивать непротиворечивый, полный, минимальный и удобный интерфейс класса. При возникновении ошибок должны выбрасываться исключения. В программе должна выполняться проверка всех разработанных элементов класса.

Вариант 12 Описать структуру с именем NOTE, содержащую следующие поля: номер телефона; дата рождения (массив из трех чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в массив, состоящий из восьми элементов типа NOTE (записи должны быть упорядочены по дате рождения); вывод на экран информации о человеке, номер телефона которого введен с клавнатуры (если такого нет, вывести соответствующее сообщение)

using System;

class Program

{

// Определение структуры NOTE

struct NOTE

{

// Поля

private string phoneNumber;

private int[] birthDate; // [день, месяц, год]

// Конструктор с параметрами

public NOTE(string phoneNumber, int day, int month, int year)

{

this.phoneNumber = phoneNumber;

this.birthDate = new int[3] { day, month, year };

}

// Свойства для доступа к полям

public string PhoneNumber

{

get => phoneNumber;

set => phoneNumber = value;

}

public int[] BirthDate

{

get => birthDate;

set => birthDate = value;

}

// Метод для сравнения дат рождения

public static bool CompareByBirthDate(NOTE note1, NOTE note2)

{

if (note1.birthDate[2] != note2.birthDate[2]) // Сравнение по году

return note1.birthDate[2] < note2.birthDate[2];

if (note1.birthDate[1] != note2.birthDate[1]) // Сравнение по месяцу

return note1.birthDate[1] < note2.birthDate[1];

return note1.birthDate[0] < note2.birthDate[0]; // Сравнение по дню

}

// Переопределение метода ToString для вывода информации

public override string ToString()

{

return $"Телефон: {phoneNumber}, Дата рождения: {birthDate[0]}.{birthDate[1]}.{birthDate[2]}";

}

}

static void Main()

{

// Создание массива из 8 элементов типа NOTE

NOTE[] notes = new NOTE[8];

try

{

// Ввод данных с клавиатуры

Console.WriteLine("Введите данные для 8 человек:");

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

{

Console.WriteLine($"Человек {i + 1}:");

Console.Write("Номер телефона: ");

string phoneNumber = Console.ReadLine();

Console.Write("Дата рождения (день): ");

int day = int.Parse(Console.ReadLine());

Console.Write("Дата рождения (месяц): ");

int month = int.Parse(Console.ReadLine());

Console.Write("Дата рождения (год): ");

int year = int.Parse(Console.ReadLine());

// Инициализация структуры через конструктор с параметрами

notes[i] = new NOTE(phoneNumber, day, month, year);

}

// Сортировка массива по дате рождения

Array.Sort(notes, (note1, note2) => NOTE.CompareByBirthDate(note1, note2) ? -1 : 1);

// Вывод отсортированного массива

Console.WriteLine("\nОтсортированный список:");

foreach (var note in notes)

{

Console.WriteLine(note);

}

// Поиск по номеру телефона

Console.Write("\nВведите номер телефона для поиска: ");

string searchPhone = Console.ReadLine();

bool found = false;

foreach (var note in notes)

{

if (note.PhoneNumber == searchPhone)

{

Console.WriteLine($"Найден: {note}");

found = true;

break;

}

}

if (!found)

{

Console.WriteLine("Человек с таким номером телефона не найден.");

}

}

catch (Exception ex)

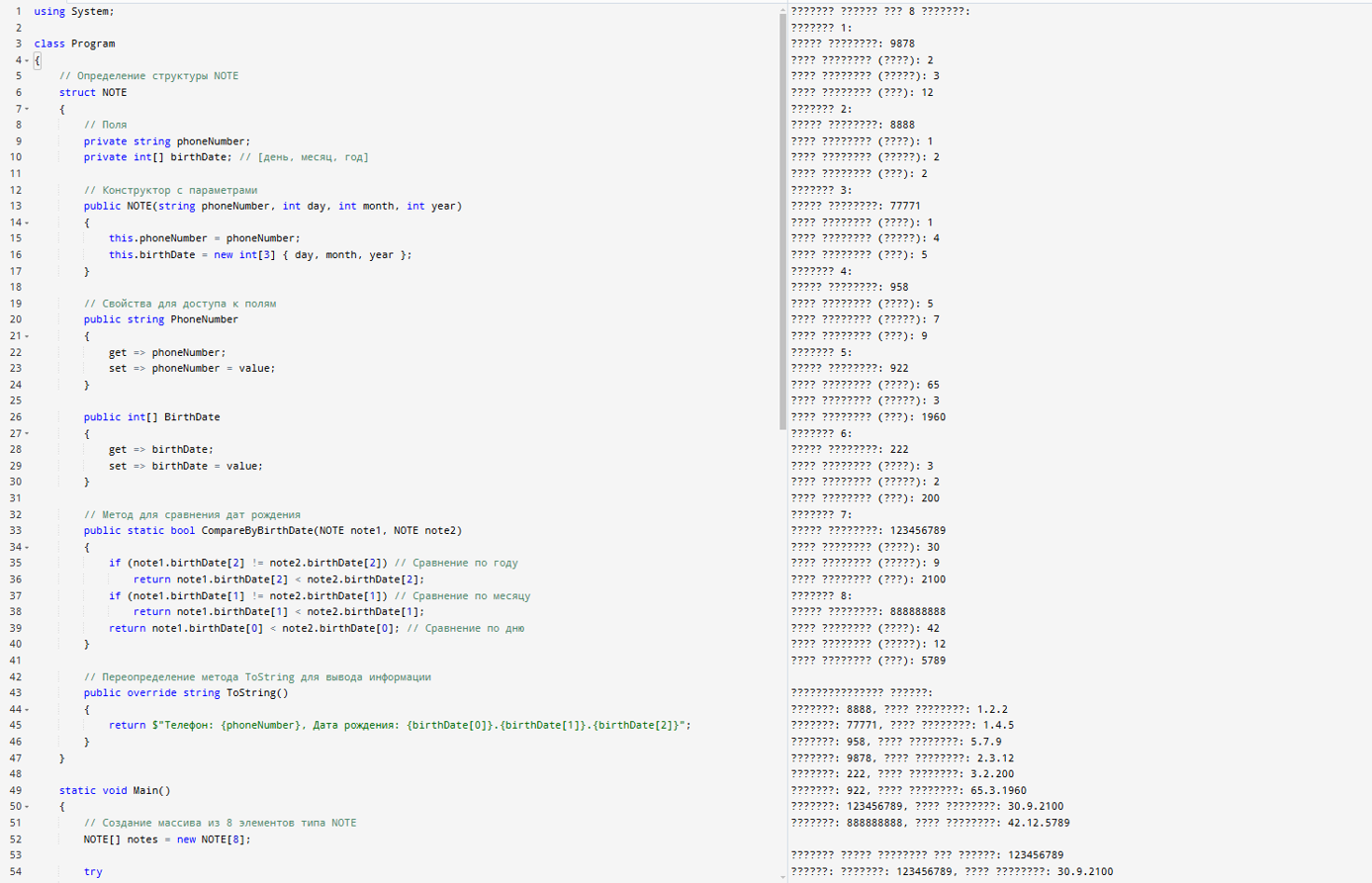
{

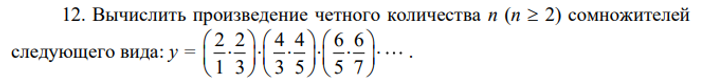
Console.WriteLine($"Ошибка: {ex.Message}");

}

}

}

****

****

using System;

class Program

{

static void Main()

{

// Ввод значения n

Console.Write("Введите значение n (n >= 2): ");

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

if (n < 2)

{

Console.WriteLine("Ошибка: n должно быть больше или равно 2.");

return;

}

// Вычисление произведения

double product = 1.0; // Инициализация произведения

for (int k = 1; k <= n; k++)

{

// Вычисление k-го сомножителя

double term = (4.0 \* k \* k) / (4.0 \* k \* k - 1);

product \*= term;

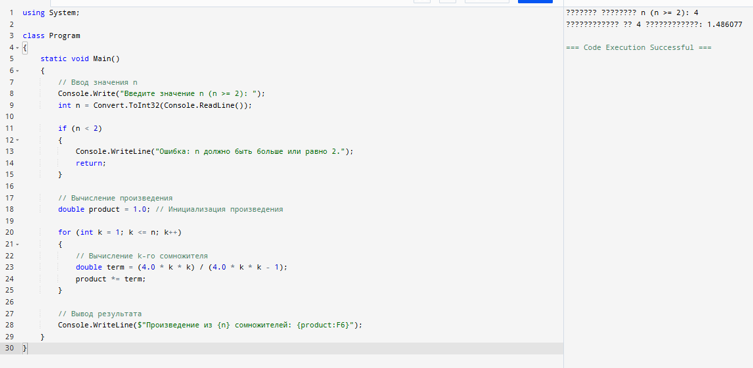
}

// Вывод результата

Console.WriteLine($"Произведение из {n} сомножителей: {product:F6}");

}

}

****