Министерство образования Республики Беларусь

Оршанский колледж учреждения образования

«Витебский государственный университет имени П. М. Машерова»

Специальность 5-04-0612-02

«Разработка и сопровождение программного обеспечения информационных систем»

Группа 3ПОИС23

**УЧЕБНАЯ ПРАКТИКА**

**ПО ПРОГРАММИРОВАНИЮ**

с 29 сентября 2025 года по 25 октября 2025 года

**ОТЧЕТ**

Выполнил(а) Шевцова А.Д.

Руководитель практики Алейников М.А.

2025

**СОДЕРЖАНИЕ**

[Модуль 1. Основы C# 4](#_Toc210035478)

[Модуль 2. Объектно-ориентированное программирование (ООП) 4](#_Toc210035479)

[Модуль 3. Задания по делегатам 4](#_Toc210035480)

[Модуль 4. Задания по интерфейсам 4](#_Toc210035481)

[Модуль 5. Графический интерфейс 4](#_Toc210035482)

[Модуль 6. Базы данных 4](#_Toc210035483)

[Модуль 7. Проектная работа 4](#_Toc210035484)

[Модуль 8. Реализация проекта 4](#_Toc210035485)

[8.1. Постановка задачи 4](#_Toc210035486)

[8.1.1. Описание предметной области 4](#_Toc210035487)

[8.1.2. Характеристика решаемой задачи 4](#_Toc210035488)

[8.1.3. Разработка моделей решаемой задачи 4](#_Toc210035489)

[8.2. Проектирование программного модуля 4](#_Toc210035490)

[8.2.1. Сбор исходных материалов 4](#_Toc210035491)

[8.2.2. Проектирование информационной модели 4](#_Toc210035492)

[8.2.3. Описание входных и выходных данных 4](#_Toc210035493)

[8.3. Реализация программного модуля 4](#_Toc210035494)

[8.3.1. Описание диаграммы классов разрабатываемого проекта 4](#_Toc210035495)

[8.3.2 Описание структуры разрабатываемого проекта 4](#_Toc210035496)

[8.3.3. Проектирование и реализация интерфейса программы 4](#_Toc210035497)

[Модуль 9. Совершенствование навыков ООП 4](#_Toc210035498)

[ЗАКЛЮЧЕНИЕ 4](#_Toc210035499)

[Список использованных источников 4](#_Toc210035500)

# Модуль 1. Основы C#

**Цель:** разработать модуль 1.

**Модуль 1.1.**

**Задания. (Вариант 2.)**

1. Напишите программу, которая проверяет, является ли введенное число четным или нечетным, и выводит соответствующее сообщение.
2. Напишите программу, которая запрашивает у пользователя радиус круга и выводит его площадь.
3. Напишите программу, которая запрашивает у пользователя имя и фамилию, а затем выводит их в формате "Фамилия, Имя".
4. Напишите программу, которая создает массив из 20 случайных чисел и находит максимальное и минимальное значение в массиве.
5. Напишите программу, которая запрашивает у пользователя возраст и определяет, может ли он получить водительские права (старше 18 лет).

**Выполнение практических заданий.**

**Выполнение задания 1.**

Чтобы проверить число на четность, необходимо разделить число на 2 и проверить остаток. Если остаток равен 0, то число четное, иначе – нечетное. В программе это проверяется через оператор условия if и else.

**Листинг 1.1.1.** Код программы задания 1.

Console.Write("Введите число: ");

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

if (a % 2 == 0)

{

Console.WriteLine($"{a} - четное");

}

else

{

Console.WriteLine($"{a} - нечетное");

}

Console.WriteLine("Нажмите любую клавишу для выхода...");

Console.ReadKey();

Результат выполнения программы представлен на рисунке 1.1.1. и рисунке 1.1.2

****

Рисунок 1.1.1 – Результат выполнения программы



Рисунок 1.1.2 – Результат выполнения программы

**Задание 2.**

Площадь круга можно вычислить по формуле 1.1.1. Для вычислений в программе используется библиотека Math.

(1.1.1)

**Листинг 1.1.2.** Код программы задания 2.

Console.Write("Введите радиус круга: ");

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

Console.WriteLine("Площадь круга равна " + Math.PI \* Math.Pow(radius, 2));

Console.WriteLine("Нажмите любую клавишу для выхода...");

Console.ReadKey();

Результат выполнения программы представлен на рисунке 1.1.3, также на рисунке 1.1.4 показан рисунок результата вычисления сайта https://calc.by/math-calculators/area-circle.html для проверки корректности работы программы.



Рисунок 1.1.3 – Результат выполнения программы



Рисунок 1.1.4 – Результат вычислений на сайте

**Задание 3.**

В программе сначала вводится имя и фамилия в строковый массив. Далее, чтобы отдельно получить имя и фамилию, массив разделяется на два массива по пробелу с помощью Split(). В конце программа выводит сначала вторую часть массива с фамилией, а после первую часть массива с именем.

**Листинг 1.1.3.** Код программы задания 3.

Console.Write("Введите имя и фамилию: ");

string[] str = Console.ReadLine()!.Split(' ');

Console.WriteLine($"Вывод в формате 'Фамилия, Имя': {str[1]}, {str[0]}");

Console.WriteLine("Нажмите любую клавишу для выхода...");

Console.ReadKey();

На рисунке 1.1.5 представлен результат выполнения программы листинга 1.1.3.



Рисунок 1.1.5 – Результат выполнения программы

**Задание 4.**

В программе сначала задается константа – размер массива. После создается объект целочисленного массива, и объект класса Random. В цикле for массив заполняется случайными значениями от 0 до 100. Массив с случайными значениями выводится. С помощью Array.Sort() массив сортируется. Так как первый элемент и последний элемент отсортированного массива имеет наименьшее и наибольшее значение соответственно, то они и выводятся как минимальный и максимальный элементы.

**Листинг 1.1.4.** Код программы задания 4.

const int size = 30;

int[] numbers = new int[size];

Random rand = new Random();

// Заполнение массива случайными значениями

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

numbers[i] = rand.Next(0, 100);

// Вывод массива

Console.WriteLine("Массив случайных чисел: " + string.Join(" ", numbers));

Array.Sort(numbers); // сортировка по возрастанию

int min = numbers[0]; // Первый элемент

int max = numbers[^1]; // Последний элемент

Console.WriteLine($"Минимум: {min}");

Console.WriteLine($"Максимум: {max}");

Console.WriteLine("Нажмите любую клавишу для выхода...");

Console.ReadKey();

Результат выполнения программы представлен на рисунке 1.1.6.



Рисунок 1.1.6 – Результат выполнения программы

**Задание 5.**

Проверка возраста выполняется с помощью тернарного оператора вида [первый операнд - условие] ? [второй операнд] : [третий операнд], где условие – это проверка целочисленной переменной age.

**Листинг 1.1.5.** Код программы задания 5.

Console.Write("Введите возраст: ");

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

Console.WriteLine(age >= 18 ? "Можно получить водительские права" : "Нельзя получить водительские права");

Console.WriteLine("Нажмите любую клавишу для выхода...");

Console.ReadKey();

Результат выполнения программы показан на рисунке 1.1.7 (вводимое значение меньше 18) и рисунке 1.1.8 (вводимое значения больше 18).



Рисунок 1.1.7 – Результат выполнения программы



Рисунок 1.1.8 – Результат выполнения программы

**Задания. Модуль 1.2.**

Задача 1. Ввести размер массива N и значения его элементов. Нормировать элементы массива, разделив их на значение максимального по модулю элемента. Вывести значения элементов измененного массива.

Задача 2. Определить и инициализировать целочисленный массив из 10-ти элементов. Ввести целое число и заменить им значение максимального элемента в массиве.

Задача 3. Вычислить К простых чисел. Значение К ввести с клавиатуры. Вывести значения чисел, размещая их по 10 на строке

Задача 4. Определить целочисленный массив из К элементов. Присвоить элементам случайные значения из диапазона [А,В). Найти индексы минимального и максимального элементов массива. Вывести значения элементов, расположенных между найденными (включая найденные)

Задача 5. Определить символьный массив из К элементов. Присвоить элементам случайные значения букв русского алфавита. Создать новый массив, поместив в него только согласные буквы из первого массива. Значение К ввести с клавиатуры. Вывести элементы обоих массивов.

Задача 6. Определить вещественный массив из 10-ти элементов. Присвоить элементам случайные значения из диапазона [–10, 10). Сформировать массив индексов, которые нумеруют элементы первого массива в порядке возрастания их значений

**Выполнение практических заданий.**

**Выполнение задания 1.**

Чтобы нормировать элементы массивы, необходимо разделить каждый элемент на наибольший. Для этого сортируется копия введенного массива. Максимальное значение – последний элемент массива. В цикле происходит деление всех элементов на максимальный.

**Листинг 1.2.1.** Код программы задания 1.

Console.Write("Введите количество элементов массива: ");

int count = Convert.ToInt32(Console.ReadLine()); // Ввод количества элементов в массива

var array = new float[count];

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

{

Console.Write($"Введите {i} элемент: ");

array[i] = Convert.ToInt32(Console.ReadLine()); // Ввод значений массива

}

Console.Write("Массив: ");

Console.WriteLine(string.Join(" ", array)); // Вывод заполненного массива

var cpArr = new float[count];

Array.Copy(array, cpArr, count);

Array.Sort(cpArr); // Сортировка массива

var maxNum = cpArr[^1]; // Максимальный элемент

Console.WriteLine("Максимальный элемент: " + maxNum);

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

{

array[i] = array[i] / maxNum;

}

Console.Write("Нормированный массив: ");

Console.WriteLine(string.Join(" ", array));

Console.ReadKey();

Результат выполнения программы представлен на рисунке 1.2.1.



Рисунок 1.2.1 – Результат выполнения программы

**Задание 2.**

Вводится цикл с клавиатуры и число, используемое в замене. Индекс наибольшего элемента находится с помощью IndexOf(). После элемент с найденным индексом заменяется на введенное число.

**Листинг 1.2.2.** Код программы задания 2.

const int count = 10;

var array = new int[count];

// Цикл для ввода значений элементов

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

{

Console.Write($"Введите {i} элемент: ");

array[i] = Convert.ToInt32(Console.ReadLine()); // Ввод значения элемента массива

}

Console.Write("Введите число для замены: ");

var number = Convert.ToInt32(Console.ReadLine()); // Ввод числа для замены

var maxIndex = Array.IndexOf(array, array.Max()); // Поиск индекса максимального элемента

array[maxIndex] = number; // Замена числа

Console.WriteLine("Массив с замененным числом: " + string.Join(", ", array)); // Вывод массива

Console.ReadKey();

Результат работы программы показан на рисунке 1.2.2.



Рисунок 1.2.2. – Результат выполнения программы

**Задание 3.**

Поиск простых чисел происходит в цикле. После нахождения выводятся найденные числа.

**Листинг 1.2.3.** Код программы задания 3.

Console.Write("Введите К: ");

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

int count = 0, num = 2;

// Цикл для поиска простых чисел

while (count < K)

{

bool prime = true;

// i \* i потому что проверяем только до корня от num

for (int i = 2; i \* i <= num; i++)

if (num % i == 0)

{

prime = false; // Делится на что-то кроме 1 и себя - не простое

}

if (prime)

{

Console.Write($"{num, 5}"); // Вывод с шириной 5

count++;

if (count % 10 == 0) Console.WriteLine();

}

num++;

}

Console.ReadKey();

На рисунке 1.2.3 показан результат работы программы листинга 1.2.3.З



Рисунок 1.2.3 – Результат выполнения программы

**Задание 4.**

Задается массив с помощью случайных значений. Индексы элементов с максимальным и минимальным значением находится используя IndexOf().

**Листинг 1.2.4.** Код программы задания 4.

Console.Write("Введите количество элементов: ");

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

int[] array = new int[K];

var rand = new Random(); // Создание нового объекта класса Random

Console.Write("Введите значение A: ");

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

Console.Write("Введите значение B: ");

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

// Цикл для заполнения массива

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

{

array[i] = rand.Next(A, B);

}

Console.WriteLine(string.Join(", ", array));

var minIndex = Array.IndexOf(array, array.Min()); // Индекс минимального элемента

var maxIndex = Array.IndexOf(array, array.Max()); // Индекс максимального элемента

Console.WriteLine($"Минимальное значение: {array.Min()}, его индекс: {minIndex}\n" +

$"Максимальное значение: {array.Max()}, его индекс: {maxIndex}");

var firstIndex = Math.Min(minIndex, maxIndex); // Первый индекс по счету

var secondIndex = Math.Max(minIndex, maxIndex); // второй индекс по счету

Console.Write("Значения в диапозоне: ");

for (int i = firstIndex; i <= secondIndex; i++)

{

Console.Write(array[i] + " ");

}

Console.ReadKey();



Рисунок 1.2.4 – Результат выполнения программы

**Задание 5.**

Задаются символьные массивы: массив всего русского алфавит и гласных букв. Создается новый символьный массив, который заполняется случайными русскими буквами из соответствующего символьного массива. Далее используя List<> просиходит поиск согласных.

**Листинг 1.2.5.** Код программы задания 5.

Console.Write("Введите количество элементов массива: ");

var K = Convert.ToInt32(Console.ReadLine());

char[] alphabet = "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ".ToCharArray(); // Массив всех букв

char[] vowels = "АЕЁИОУЫЭЮЯ".ToCharArray(); // Массив гласных

Random rand = new Random(); // Создание объекта класса Random

var array = new char[K]; // Массив для случайных букв

// Цикл для заполнения массива случайными буквами

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

array[i] = alphabet[rand.Next(0, alphabet.Length)]; // Добавление случайной буквы

List<char> constants = new List<char>(K); // Лист для согласных (чтобы меньше кода было)

// Поиск согласных

foreach (var letter in array)

{

if(!vowels.Contains(letter))

constants.Add(letter);

}

Console.WriteLine("Массив случайных букв: " + string.Join(' ', array)); // Вывод массива со случайными буквами

Console.WriteLine("Лист согласных из массива случайных букв" + string.Join(' ', constants)); // Вывод массива с согласными

Console.ReadKey();

Результат выполнения программы листинга 1.2.5 представлен на рисунке 1.2.5.



Рисунок 1.2.5 – Результат выполнения программы

**Задание 6.**

Создается массив и заполняется случайными вещественными числами. Далее происходит сортировка индексов по возрастанию значений.

**Листинг 1.2.6.** Код программы задания 6.

const int size = 10;

var array = new float[size];

var indices = new int[size];

var rand = new Random();

// Цикл для заполнения случацными вещественными значениями

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

{

array[i] = (float)rand.NextDouble() \* 20 - 10; // [0,1) > [0,20) > [-10,10)

indices[i] = i; // Начальная нумераия

}

// Цикл сортировки индексов по значениям массива

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

{

for (int j = i + 1; j < size; j++)

{

if (array[indices[i]] > array[indices[j]])

{

(indices[i], indices[j]) = (indices[j], indices[i]);

}

}

}

Console.WriteLine("Исходный массив: " + string.Join(" ", array.Select(n => n.ToString("F2"))));

Console.WriteLine("Массив индексов по возрастанию значений: " + string.Join(' ', indices));

Console.ReadKey();

На рисунке 1.2.6 показан результат выполнения программы листинга 1.2.6.



Рисунок 1.2.6 – Результат выполнения программы

**Задания. Модуль 1.3.**

Задача 1. Определить функцию (статический метод) для вычисления наибольшего общего делителя двух целых натуральных чисел (Greatest Common Measure). В основной программе, используя функцию, сократить неотрицательную обыкновенную дробь. Дробь вводится с клавиатуры в виде неотрицательного числителя и положительного знаменателя

Задача 2. Присваивая последовательным элементам массива случайные значения от 1 до 9, создать массив с минимальным количеством элементов, сумма которых не превышает заданного пользователем числа.

Задача 3. Сформировав квадратную целочисленную матрицу со случайными значениями элементов, упорядочить ее строки по возрастанию сумм их элементов. Пусть элементы принимают значения в диапазоне от –50 до +50.

**Задание 1.**

Наибольший общий делитель находится с помощью алгоритма Евклида – большее число делится на меньшее, если делится без остатка, то меньшее число и есть НОД. Если остаток есть, то большее число заменяется на остаток от деления.

**Листинг 1.3.1.** Код программы задания 1.

using System;

class Program

{

static int NOD(int a, int b)

{

// Алгоритм Евклида

while (a != 0 && b != 0)

{

if (a > b)

a %= b;

else b %= a;

}

return a + b;

}

static void Main(string[] args)

{

Console.Write("Введите неотрицательный числитель: ");

int a = int.Parse(Console.ReadLine()!);

Console.Write("Введите положительный знаменатель: ");

int b = int.Parse(Console.ReadLine()!);

int nod = NOD(a, b);

Console.WriteLine("Наибольший общий делитель: " + nod);

Console.WriteLine($"Сокращенная дробь: {a/nod} / {b/nod}");

Console.ReadKey();

}

}

На рисунке 1.3.1. представлен результат выполнения программы листинга 1.3.1.



Рисунок 1.3.1 – Результат выполнения программы

**Задание 2.**

Значение максимальной сумме вводится с клавиатуры. Далее пока сумма значений элементов массива, которые генерируются случайно в диапазоне, не привышает введенную сумму – элементы продолжают генерироваться.

**Листинг 1.3.2.** Код программы задания 2.

using System;

class Program

{

static void Main()

{

Console.Write("Введите значение максимальной суммы: ");

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

int sum = 0;

var array = new int[1];

var temp = new int[maxSum];

var count = 0;

var rand = new Random();

while (sum < maxSum)

{

int value = rand.Next(1, 10);

if (sum + value > maxSum) break;

temp[count++] = value;

sum += value;

}

// Создание итогового массива нужной длины

var result = new int[count];

Array.Copy(temp, result, count);

Console.WriteLine("Сформированный массив: " + string.Join(" ", result) + "\nСумма элементов: " + sum);

Console.ReadKey();

}

}

Результат выполнения программы листинга 1.3.2 показан на рисунке 1.3.2.



Рисунок 1.3.2 – Результат выполнения программы

**Задание 3.**

Вводится размер квадратной матрицы. Значения двумерного массива случайные в заданном диапозоне. Исходная матрица выводится с помощью функции PrintMatrix(). Далее проихсходит сортировка матрицы по сумме значений в строке. В конце выводится отсортированная по строкам матрица.

**Листинг 1.3.3.** Код программы задания 3.

using System;

class Program

{

static void Main(string[] args)

{

Console.Write("Введите размер квадратной матрицы: ");

var size = Convert.ToInt32(Console.ReadLine());

int[,] matrix = new int[size, size];

var rand = new Random();

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

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

matrix[i, j] = rand.Next(-50, 51);

Console.WriteLine("Исходная матрица: ");

PrintMatrix(matrix);

//Сортировка по сумме

// Перебирает строки матрицы

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

{

// Сравнивает текущую строку i с каждой последующей строкой k

for (int k = i + 1; k < size; k++)

{

// Если сумма одной строки больше суммы другой

if (RowSum(matrix, i) > RowSum(matrix, k))

{

// Поменять местами

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

{

(matrix[i, j], matrix[k, j]) = (matrix[k, j], matrix[i, j]);

}

}

}

}

Console.WriteLine("Отсортированная матрица:");

PrintMatrix(matrix);

Console.ReadKey();

}

// Метод для поиска суммы ряда

static int RowSum(int[,] matrix, int row)

{

int sum = 0;

for (int i = 0; i < matrix.GetLength(1); i++)

sum += matrix[row, i];

return sum;

}

// Метод для вывода матрицы

static void PrintMatrix(int[,] matrix)

{

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($"{matrix[i, j], 5}");

Console.WriteLine();

}

}

}

Результат выполнения программы листинга 1.3.3 представлен на рисунке 1.3.3.



Рисунок 1.3.3. – Результат выполнения программы

# Модуль 2. Объектно-ориентирование программирование (ООП).

**Задание 1. Создание классов.**

Создайте класс Person, представляющий человека. У этого класса должны быть поля для хранения имени, возраста и адреса. Добавьте методы для установки и получения значений этих полей. Затем создайте объекты этого класса и выведите информацию о них.

**Задание 2. Наследование и полиморфизм.**

Создайте базовый класс Shape, представляющий геометрическую фигуру, и производные классы Circle и Rectangle. В базовом классе определите метод Area(), который возвращает площадь фигуры, и метод Perimeter(), который возвращает периметр фигуры. В производных классах переопределите эти методы для соответствующих фигур (круг и прямоугольник). Создайте объекты всех классов и выведите их площади и периметры.

**Задание 3. Композиция.**

Создайте классы Author и Book. Класс Author должен содержать информацию об авторе (имя и год рождения). Класс Book должен содержать информацию о книге (название, год выпуска и автора). Используйте композицию, чтобы связать объекты Author и Book. Создайте несколько объектов Author и Book и выведите информацию о них.

**Задание 4. Интерфейсы и абстрактные классы.**

Создайте интерфейс IDrawable с методом Draw(), который выводит информацию о рисуемом объекте. Создайте классы Circle, Rectangle и Triangle, реализующие этот интерфейс. Создайте массив объектов, реализующих интерфейс IDrawable, и вызовите метод Draw() для каждого из них.

**Задание 5. События.**

Создайте класс TemperatureSensor, который генерирует событие TemperatureChanged, когда измеренная температура меняется. Создайте класс Thermostat, который подписывается на событие TemperatureChanged и реагирует на изменение температуры, включая или выключая отопление.

**Задания по вариантам. (Вариант 2)**

**Задание 1.**

Создайте класс "Автомобиль" с полями: марка, модель, год выпуска и цена. Реализуйте методы для расчета стоимости автомобиля с учетом скидки и налога на добавленную стоимость.

**Задание 2.**

Создайте структуру с именем student, содержащую поля: фамилия и инициалы, номер группы, успеваемость (массив из пяти элементов). Создать массив из десяти элементов такого типа, упорядочить записи по возрастанию среднего балла. Добавить возможность вывода фамилий и номеров групп студентов, имеющих оценки, равные только 4 или 5.

**Задание 3.**

Создание класса "Геометрическая фигура" (Shape) и его производных: Создайте абстрактный класс "Геометрическая фигура" с методом для вычисления площади. Создайте производные классы, например, "Круг" (Circle), "Прямоугольник" (Rectangle), "Треугольник" (Triangle) и т. д. Реализуйте методы для вычисления площади у каждого класса. Создайте объекты различных классов и протестируйте их функциональность.

**Выполнение практических заданий.**

**Задание 1.**

Создается класс Person с публичными полями Name, Age, Address. Данные вводятся с помощью конструктара класса, выводятся с помощью метода Print().

**Листинг 2.1.** Код программы задания 1.

using System;

class Person

{

public string Name { get; set; }

public int Age { get; set; }

public string Address { get; set; }

// Конструктор класса Person

public Person(string name = "Не введено!", int age = 0, string address = "Не введено!")

{

Name = name;

Age = age;

Address = address;

}

public void Print() => Console.WriteLine($"Имя: {Name}, Возраст: {Age}, Адрес: {Address}");

}

class Program

{

static void Main(string[] args)

{

Console.Write("Введите имя: ");

var name = Console.ReadLine()!;

Console.Write("Введите возраст: ");

var age = int.Parse(Console.ReadLine()!);

Console.Write("Введите адрес: ");

var address = Console.ReadLine()!;

var person = new Person(name, age, address);

person.Print();

Console.ReadKey();

}

}



Рисунок 2.1 – Результат выполнения программы

**Задание 2.**

**Листинг 2.2.** Код программы задания 2.

**Задание 3.**

**Листинг 2.3.** Код программы задания 3.

**Задание 4.**

**Листинг 2.4.** Код программы задания 4.

**Задание 5.**

**Листинг 2.5.** Код программы задания 5.

**Выполнение заданий по вариантам.**

**Задание 1.**

**Листинг 2.6.** Код программы задания 1.

**Задание 2.**

**Листинг 2.7.** Код программы задания 2.

**Задание 3.**

**Листинг 2.8.** Код программы задания 3.

# Модуль 2. Объектно-ориентированное программирование (ООП)

# Модуль 3. Задания по делегатам

# Модуль 4. Задания по интерфейсам

# Модуль 5. Графический интерфейс

# Модуль 6. Базы данных

# Модуль 7. Проектная работа

# Модуль 8. Реализация проекта

8.1. Постановка задачи

## 8.1.1. Описание предметной области

## 8.1.2. Характеристика решаемой задачи

## 8.1.3. Разработка моделей решаемой задачи

8.2. Проектирование программного модуля

## 8.2.1. Сбор исходных материалов

## 8.2.2. Проектирование информационной модели

## 8.2.3. Описание входных и выходных данных

8.3. Реализация программного модуля

## 8.3.1. Описание диаграммы классов разрабатываемого проекта

## 8.3.2 Описание структуры разрабатываемого проекта

## 8.3.3. Проектирование и реализация интерфейса программы

# Модуль 9. Совершенствование навыков ООП

# ЗАКЛЮЧЕНИЕ

# Список использованных источников

Заключение

Список использованных источников

Карпова, И.Б. Основы программирования на языке Object Pascal / И.Б. Карпова. М.: Академия, 2015. 352 с.

Кормен, Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ / Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн. М.: Вильямс, 2019. 1312 с.

Круг, С. Не заставляйте меня думать / С. Круг. М.: Манн, Иванов и Фербер, 2014. 256 с.

Мартин, Р. Чистая архитектура. Искусство разработки программного обеспечения / Р. Мартин. М.: ДМК Пресс, 2018. 400 с.

Осипов, Д. Delphi. Программирование для Windows, OS X, iOS и Android / Д. Осипов. М.: БХВ-Петербург, 2014. 464 с.

ГОСТ 19.102-77. ЕСПД. Требования к программной документации.

ГОСТ 34.601-90. Автоматизированные системы. Стадии создания.

Boehm, B.W. Экономика программной инженерии / Б.В. Боэм. М.: Мир, 1987. 528 с.

Delphi (software) — Википедия [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Delphi\_(%D1%81%D0%BE%D1%84%D1%82%D0%B2%D0%B5%D1%80). Дата доступа: 06.04.2025.

GIF — Википедия [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/GIF. Дата доступа: 06.04..2025.

ImageEn component suite for image processing in Delphi [Электронный ресурс]. Режим доступа: https://www.imageen.com/. Дата доступа: 07.04.2025.

Median cut - Wikipedia [Электронный ресурс]. Режим доступа: https://en.wikipedia.org/wiki/Median\_cut. Дата доступа: 10.04.2025.

Р 50.1.028-2001. Информационные технологии поддержки жизненного цикла продукции. Методология функционального моделирования.

The Median Cut Algorithm for Color Quantization / Gabriel Ytterberg [Электронный ресурс]. Режим доступа: https://medium.com/@gytterberg\_14295/the-median-cut-algorithm-for-color-quantization-cc1128a0c534. Дата доступа: 15.05.2025.

Video Editing Software Market Size, Share & Growth Report by 2033 [Электронный ресурс]. Режим доступа: https://straitsresearch.com/report/video-editing-software-market. Дата доступа: 15.05.20