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

|  |  |  |
| --- | --- | --- |
| **Факультет/Институт** |  | МФПУ Синергия |
|  |  | (наименование факультета/ Института) |
| **Направление/специальность** |  | Программист |
| **подготовки:** |  | (код и наименование направления /специальности подготовки) |
| **Форма обучения:** |  | очная |
|  |  | (очная, очно-заочная, заочная) |
|  |  |  |

**Отчет по лабораторной работе № 5.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **на тему** |  | Проектирование классов | | |
|  |  | (наименование темы) | | |
|  |  |  | | |
| **по дисциплине** | | |  | Разработка ПО |
|  | | |  | (наименование дисциплины) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Обучающийся** |  | Панцулая Темур Генадьевич |  |  |
|  |  | (ФИО) |  | (подпись) |
| **Группа** |  | ДКИП\_205 ПРОГ |  |  |
|  |  |  |  |  |

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

**Москва 2025 г.**

**Вариант 14**

Описать класс дробей - рациональных чисел, являющиеся отношением двух целых

чисел. Предусмотреть методы сложения, вычитания, умножения и деления дробей.

Написать программу, демонстрирующую все разработанные элементы класс.

**Листинг кода:**

**using System;**

**public class Fraction**

**{**

**private int numerator;   // числитель**

**private int denominator; // знаменатель**

**public Fraction(int numerator, int denominator)**

**{**

**if (denominator == 0)**

**throw new ArgumentException("Знаменатель не может быть равен нулю.");**

**this.numerator = numerator;**

**this.denominator = denominator;**

**Simplify();**

**}**

**// Метод для упрощения дроби**

**private void Simplify()**

**{**

**int gcd = GCD(Math.Abs(numerator), Math.Abs(denominator));**

**numerator /= gcd;**

**denominator /= gcd;**

**// Чтобы знаменатель всегда был положительным**

**if (denominator < 0)**

**{**

**denominator = -denominator;**

**numerator = -numerator;**

**}**

**}**

**// Нахождение наибольшего общего делителя (НОД)**

**private int GCD(int a, int b)**

**{**

**while (b != 0)**

**{**

**int temp = b;**

**b = a % b;**

**a = temp;**

**}**

**return a;**

**}**

**// Сложение дробей**

**public Fraction Add(Fraction other)**

**{**

**int newNumerator = numerator \* other.denominator + other.numerator \* denominator;**

**int newDenominator = denominator \* other.denominator;**

**return new Fraction(newNumerator, newDenominator);**

**}**

**// Вычитание дробей**

**public Fraction Subtract(Fraction other)**

**{**

**int newNumerator = numerator \* other.denominator - other.numerator \* denominator;**

**int newDenominator = denominator \* other.denominator;**

**return new Fraction(newNumerator, newDenominator);**

**}**

**// Умножение дробей**

**public Fraction Multiply(Fraction other)**

**{**

**int newNumerator = numerator \* other.numerator;**

**int newDenominator = denominator \* other.denominator;**

**return new Fraction(newNumerator, newDenominator);**

**}**

**// Деление дробей**

**public Fraction Divide(Fraction other)**

**{**

**if (other.numerator == 0)**

**throw new DivideByZeroException("Деление на нулевую дробь невозможно");**

**int newNumerator = numerator \* other.denominator;**

**int newDenominator = denominator \* other.numerator;**

**return new Fraction(newNumerator, newDenominator);**

**}**

**public override string ToString()**

**{**

**if (denominator == 1)**

**return numerator.ToString();**

**else**

**return $"{numerator}/{denominator}";**

**}**

**}**

**class Program**

**{**

**static void Main()**

**{**

**var frac1 = new Fraction(3, 4);**

**var frac2 = new Fraction(5, 6);**

**Console.WriteLine($"Дробь 1: {frac1}");**

**Console.WriteLine($"Дробь 2: {frac2}");**

**var sum = frac1.Add(frac2);**

**Console.WriteLine($"Сложение: {sum}");**

**var diff = frac1.Subtract(frac2);**

**Console.WriteLine($"Вычитание: {diff}");**

**var product = frac1.Multiply(frac2);**

**Console.WriteLine($"Умножение: {product}");**

**var quotient = frac1.Divide(frac2);**

**Console.WriteLine($"Деление: {quotient}");**

**}**

**}**

**Вариант 14**

В одномерном массиве, состоящем из п вещественных элементов, вычислить:

количество элементов массива, равных нулю;

сумму элементов массива, расположенных после минимального элемента.

Упорядочить элементы массива по возрастанию модулей.

**Листинг кода:**

using System;

class Program

{

static void Main()

{

// Пример заполнения массива

double[] array = { 3.5, 0.0, -2.1, 4.6, 0.0, 1.2, -3.0 };

// Подсчет количества нулевых элементов

int zeroCount = CountZeros(array);

Console.WriteLine("Количество элементов, равных нулю: " + zeroCount);

// Сумма элементов после минимального

double sumAfterMin = SumAfterMin(array);

Console.WriteLine("Сумма элементов после минимального: " + sumAfterMin);

// Сортировка массива по возрастанию модулей

SortByAbsoluteValue(array);

Console.WriteLine("Массив после сортировки по возрастанию модулей: ");

PrintArray(array);

}

static int CountZeros(double[] array)

{

int count = 0;

foreach (double num in array)

{

if (num == 0.0)

{

count++;

}

}

return count;

}

static double SumAfterMin(double[] array)

{

double sum = 0.0;

if (array.Length == 0) return sum;

double min = array[0];

int minIndex = 0;

// Находим минимальное значение и его индекс

for (int i = 1; i < array.Length; i++)

{

if (array[i] < min)

{

min = array[i];

minIndex = i;

}

}

// Считаем сумму элементов после минимального

for (int i = minIndex + 1; i < array.Length; i++)

{

sum += array[i];

}

return sum;

}

static void SortByAbsoluteValue(double[] array)

{

Array.Sort(array, (x, y) => Math.Abs(x).CompareTo(Math.Abs(y)));

}

static void PrintArray(double[] array)

{

foreach (double num in array)

{

Console.Write(num + " ");

}

Console.WriteLine();

}

}

**Вариант 14**

Описать класс «автостоянка» для хранения сведений об автомобилях. Для каждого

автомобиля записываются госномер, цвет, фамилия владельца и признак присутствия на

стоянке. Обеспечить возможность поиска автомобиля по разным критериям, вывода

списка присутствующих и отсутствующих на стоянке автомобилей, доступа к

имеющимся сведениям по померу места. Написать программу, демонстрирующую все

разработанные элементы класса.

**Листинг кода:**

using System;

using System.Collections.Generic;

class Car

{

public string LicensePlate { get; set; }

public string Color { get; set; }

public string Owner { get; set; }

public bool IsPresent { get; set; }

public Car(string licensePlate, string color, string owner, bool isPresent)

{

LicensePlate = licensePlate;

Color = color;

Owner = owner;

IsPresent = isPresent;

}

public override string ToString()

{

return $"{LicensePlate}, {Color}, {Owner}, {(IsPresent ? "На стоянке" : "Не на стоянке")}";

}

}

class ParkingLot

{

private List<Car> cars;

public ParkingLot()

{

cars = new List<Car>();

}

public void AddCar(Car car)

{

cars.Add(car);

}

public Car FindCarByLicensePlate(string licensePlate)

{

return cars.Find(car => car.LicensePlate.Equals(licensePlate, StringComparison.OrdinalIgnoreCase));

}

public List<Car> GetPresentCars()

{

return cars.FindAll(car => car.IsPresent);

}

public List<Car> GetAbsentCars()

{

return cars.FindAll(car => !car.IsPresent);

}

public void DisplayCars()

{

foreach (var car in cars)

{

Console.WriteLine(car);

}

}

}

class Program

{

static void Main()

{

ParkingLot parkingLot = new ParkingLot();

// Добавление автомобилей

parkingLot.AddCar(new Car("A123BC", "Красный", "Иванов", true));

parkingLot.AddCar(new Car("B456CD", "Синий", "Петров", false));

parkingLot.AddCar(new Car("C789EF", "Зеленый", "Сидоров", true));

Console.WriteLine("Список всех автомобилей на стоянке:");

parkingLot.DisplayCars();

// Поиск по госномеру

string searchPlate = "A123BC";

Car foundCar = parkingLot.FindCarByLicensePlate(searchPlate);

if (foundCar != null)

{

Console.WriteLine($"\nАвтомобиль с госномером {searchPlate}: {foundCar}");

}

else

{

Console.WriteLine($"\nАвтомобиль с госномером {searchPlate} не найден.");

}

// Список присутствующих

Console.WriteLine("\nПрисутствующие на стоянке автомобили:");

foreach (var car in parkingLot.GetPresentCars())

{

Console.WriteLine(car);

}

// Список отсутствующих

Console.WriteLine("\nОтсутствующие на стоянке автомобили:");

foreach (var car in parkingLot.GetAbsentCars())

{

Console.WriteLine(car);

}

}

}

**Вариант 14**

Описать структуру с именем NOTE, содержащую следующие поля:

фамилия, имя;

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 трем первым цифрам номера телефона);

вывод на экран информации о человеке, чья фамилия введена с клавнатуры (если

такого нет, вывести соответствующее сообщение)

**Листинг кода:**

**#include <stdio.h>**

**#include <string.h>**

**#define MAX\_NOTES 8**

**#define MAX\_NAME\_LEN 50**

**#define PHONE\_LEN 15**

**typedef struct {**

**char lastName[MAX\_NAME\_LEN];**

**char firstName[MAX\_NAME\_LEN];**

**char phoneNumber[PHONE\_LEN];**

**int birthDate[3]; // день, месяц, год**

**} NOTE;**

**// Функция для ввода одной записи**

**void inputNote(NOTE\* note) {**

**printf("Введите фамилию: ");**

**fgets(note->lastName, MAX\_NAME\_LEN, stdin);**

**note->lastName[strcspn(note->lastName, "\n")] = 0; // удаляем символ новой строки**

**printf("Введите имя: ");**

**fgets(note->firstName, MAX\_NAME\_LEN, stdin);**

**note->firstName[strcspn(note->firstName, "\n")] = 0;**

**printf("Введите номер телефона: ");**

**fgets(note->phoneNumber, PHONE\_LEN, stdin);**

**note->phoneNumber[strcspn(note->phoneNumber, "\n")] = 0;**

**printf("Введите дату рождения (день месяц год): ");**

**scanf("%d %d %d", &note->birthDate[0], &note->birthDate[1], &note->birthDate[2]);**

**getchar(); // чтобы убрать символ новой строки после scanf**

**}**

**// Функция сравнения по первым трём цифрам номера телефона для сортировки**

**int compareFirstThreeDigits(const void\* a, const void\* b) {**

**const NOTE\* n1 = (const NOTE\*)a;**

**const NOTE\* n2 = (const NOTE\*)b;**

**// Сравним первые три символа номера телефона как строки**

**char prefix1[4] = {0};**

**char prefix2[4] = {0};**

**strncpy(prefix1, n1->phoneNumber, 3);**

**strncpy(prefix2, n2->phoneNumber, 3);**

**return strcmp(prefix1, prefix2);**

**}**

**// Функция поиска по фамилии**

**int findByLastName(NOTE array[], int size, const char\* lastName) {**

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

**if (strcmp(array[i].lastName, lastName) == 0) {**

**return i;**

**}**

**}**

**return -1;**

**}**

**// Функция вывода записи**

**void printNote(const NOTE\* note) {**

**printf("Фамилия: %s\n", note->lastName);**

**printf("Имя: %s\n", note->firstName);**

**printf("Номер телефона: %s\n", note->phoneNumber);**

**printf("Дата рождения: %02d.%02d.%04d\n", note->birthDate[0], note->birthDate[1], note->birthDate[2]);**

**}**

**int main() {**

**NOTE notes[MAX\_NOTES];**

**printf("Введите данные %d человек:\n", MAX\_NOTES);**

**for (int i = 0; i < MAX\_NOTES; i++) {**

**printf("\nЧеловек %d:\n", i + 1);**

**inputNote(&notes[i]);**

**}**

**// Сортировка по первым трём цифрам номера телефона**

**qsort(notes, MAX\_NOTES, sizeof(NOTE), compareFirstThreeDigits);**

**char searchLastName[MAX\_NAME\_LEN];**

**printf("\nВведите фамилию для поиска: ");**

**fgets(searchLastName, MAX\_NAME\_LEN, stdin);**

**searchLastName[strcspn(searchLastName, "\n")] = 0;**

**int index = findByLastName(notes, MAX\_NOTES, searchLastName);**

**if (index >= 0) {**

**printf("\nИнформация о человеке:\n");**

**printNote(&notes[index]);**

**} else {**

**printf("Человек с фамилией \"%s\" не найден.\n", searchLastName);**

**}**

**return 0;**

**}**