**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

ИШИТР

Отделение информационных технологий

09.03.02 Информационные системы и технологии

Лабораторная работа № 1

по дисциплине  
 **«Архитектура информационных систем»**

Выполнил:

студент гр. 8И11 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Мандриков А.Д

Проверил:

ассистент ОИТ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Коровкин В.А.

Томск 2023

**Основная часть.**

**Задание на лабораторную работу**

В ходе выполнения работы необходимо: Разработать консольное приложение для работы с данными из текстового файла в формате CSV. Используемая среда разработки MS Visual Studio, язык – C#.

**Сущность** – Яхт-клуб

**Описание класса, реализующего сущность.**

internal class YachtClub

{

public string Name { get; set; }

public string Address { get; set; }

public int NumberOfYachts { get; set; }

public int NumberOfPlaces { get; set; }

public bool HasPool { get; set; }

}

Класс имеет 5 полей в которых храняться:

1. Name – название клуба
2. Address – адрес клуба
3. NumberOfYachts – число яхт
4. NumberOfPlaces – число мест
5. HasPool – true/false в зависимости от того есть ли бассейн в клубе

**Код программы.**

**Program.cs:**using lab\_1;

using System;

using System.Collections.Generic;

class Program

{

static void Main(string[] args)

{

string filePath = "D:/учёба 5 сем/архетектуры ИС/lab\_1/yachtclubs.csv"; // Путь к файлу CSV

YachtClubController dataHandler;

List<YachtClub> yachtClubs = new List<YachtClub>();

if (!File.Exists(filePath))

{

Console.WriteLine("Файл yachtclubs.csv не существует.");

Console.Write("Введите путь для создания файла (нажмите Enter для использования пути по умолчанию): ");

string customPath = Console.ReadLine();

if (!string.IsNullOrWhiteSpace(customPath))

{

filePath = customPath;

}

dataHandler = new YachtClubController(filePath);

dataHandler.WriteRecords(yachtClubs);

}

dataHandler = new YachtClubController(filePath);

yachtClubs = dataHandler.ReadAllRecords();

do

{

Console.Clear();

Console.WriteLine("\x1b[3J");

Console.WriteLine("Меню:");

Console.WriteLine("1. Вывести все яхт-клубы");

Console.WriteLine("2. Добавить новый яхт-клуб");

Console.WriteLine("3. Удалить яхт-клуб по индексу");

Console.WriteLine("4. Изменить яхт-клуб по индексу");

Console.WriteLine("5. Загрузить яхт-клубы из файла");

Console.WriteLine("6. Сохранить яхт-клубы в файл");

Console.WriteLine("7. Выйти");

var choice = Console.ReadKey().Key; // Получаем нажатую клавишу

switch (choice)

{

case ConsoleKey.D1:

DisplayYachtClubs(yachtClubs);

break;

case ConsoleKey.D2:

AddYachtClub(dataHandler, yachtClubs);

break;

case ConsoleKey.D3:

RemoveYachtClub(dataHandler, yachtClubs);

break;

case ConsoleKey.D4:

ModifyYachtClub(dataHandler, yachtClubs);

break;

case ConsoleKey.D5:

yachtClubs = dataHandler.ReadAllRecords();

break;

case ConsoleKey.D6:

dataHandler.WriteRecords(yachtClubs);

break;

case ConsoleKey.D7:

Environment.Exit(0);

break;

case ConsoleKey.Escape: // Выход по клавише "Esc"

Environment.Exit(0);

break;

default:

Console.WriteLine("Неверный выбор. Попробуйте снова.");

break;

}

} while (true);

}

static void DisplayYachtClubs(List<YachtClub> yachtClubs)

{

Console.Clear();

Console.WriteLine("\x1b[3J");

Console.WriteLine("Список яхт-клубов:");

for (int i = 0; i < yachtClubs.Count; i++)

{

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

Console.WriteLine($"Индекс: {i}");

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

Console.WriteLine($"Название: {yachtClubs[i].Name}");

Console.WriteLine($"Адрес: {yachtClubs[i].Address}");

Console.WriteLine($"Количество яхт: {yachtClubs[i].NumberOfYachts}");

Console.WriteLine($"Количество мест: {yachtClubs[i].NumberOfPlaces}");

Console.WriteLine($"Наличие бассейна: {yachtClubs[i].HasPool}");

Console.WriteLine();

}

Console.WriteLine("Нажмите Enter для продолжения...");

Console.ReadLine();

}

static void AddYachtClub(YachtClubController dataHandler, List<YachtClub> yachtClubs)

{

Console.Clear();

Console.WriteLine("Добавление нового яхт-клуба:");

try

{

Console.Write("Название: ");

string name = Console.ReadLine();

Console.Write("Адрес: ");

string address = Console.ReadLine();

Console.Write("Количество яхт: ");

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

Console.Write("Количество мест: ");

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

Console.Write("Наличие бассейна (true/false): ");

bool hasPool = bool.Parse(Console.ReadLine());

YachtClub newClub = new YachtClub

{

Name = name,

Address = address,

NumberOfYachts = numberOfYachts,

NumberOfPlaces = numberOfPlaces,

HasPool = hasPool

};

yachtClubs.Add(newClub);

Console.WriteLine("Яхт-клуб успешно добавлен.");

}

catch (FormatException)

{

Console.WriteLine("Ошибка: Некорректный формат ввода.");

}

catch (Exception ex)

{

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

}

Console.WriteLine("Нажмите Enter для продолжения...");

Console.ReadLine();

}

static void RemoveYachtClub(YachtClubController dataHandler, List<YachtClub> yachtClubs)

{

Console.Clear();

Console.WriteLine("Удаление яхт-клуба по индексу:");

Console.Write("Введите индекс яхт-клуба для удаления: ");

if (int.TryParse(Console.ReadLine(), out int index) && index >= 0 && index < yachtClubs.Count)

{

yachtClubs.RemoveAt(index);

Console.WriteLine("Яхт-клуб успешно удален.");

}

else

{

Console.WriteLine("Некорректный индекс для удаления яхт-клуба.");

}

Console.WriteLine("Нажмите Enter для продолжения...");

Console.ReadLine();

}

static void ModifyYachtClub(YachtClubController dataHandler, List<YachtClub> yachtClubs)

{

Console.Clear();

Console.WriteLine("Изменение яхт-клуба по индексу:");

Console.Write("Введите индекс яхт-клуба для изменения: ");

try

{

if (int.TryParse(Console.ReadLine(), out int index) && index >= 0 && index < yachtClubs.Count)

{

Console.WriteLine("Текущие данные для яхт-клуба:");

Console.WriteLine($"Название: {yachtClubs[index].Name}");

Console.WriteLine($"Адрес: {yachtClubs[index].Address}");

Console.WriteLine($"Количество яхт: {yachtClubs[index].NumberOfYachts}");

Console.WriteLine($"Количество мест: {yachtClubs[index].NumberOfPlaces}");

Console.WriteLine($"Наличие бассейна: {yachtClubs[index].HasPool}");

Console.WriteLine();

Console.Write("Новое название (оставьте пустым для сохранения текущего): ");

string name = Console.ReadLine();

if (!string.IsNullOrWhiteSpace(name))

{

yachtClubs[index].Name = name;

}

Console.Write("Новый адрес (оставьте пустым для сохранения текущего): ");

string address = Console.ReadLine();

if (!string.IsNullOrWhiteSpace(address))

{

yachtClubs[index].Address = address;

}

Console.Write("Новое количество яхт (оставьте пустым для сохранения текущего): ");

string numberOfYachtsStr = Console.ReadLine();

if (!string.IsNullOrWhiteSpace(numberOfYachtsStr))

{

if (int.TryParse(numberOfYachtsStr, out int numberOfYachts))

{

yachtClubs[index].NumberOfYachts = numberOfYachts;

}

else

{

Console.WriteLine("Некорректное значение для количества яхт.");

}

}

Console.Write("Новое количество мест (оставьте пустым для сохранения текущего): ");

string numberOfPlacesStr = Console.ReadLine();

if (!string.IsNullOrWhiteSpace(numberOfPlacesStr))

{

if (int.TryParse(numberOfPlacesStr, out int numberOfPlaces))

{

yachtClubs[index].NumberOfPlaces = numberOfPlaces;

}

else

{

Console.WriteLine("Некорректное значение для количества мест.");

}

}

Console.Write("Наличие бассейна (true/false) (оставьте пустым для сохранения текущего): ");

string hasPoolStr = Console.ReadLine();

if (!string.IsNullOrWhiteSpace(hasPoolStr))

{

if (bool.TryParse(hasPoolStr, out bool hasPool))

{

yachtClubs[index].HasPool = hasPool;

}

else

{

Console.WriteLine("Некорректное значение для наличия бассейна.");

}

}

Console.WriteLine("Яхт-клуб успешно изменен.");

}

else

{

Console.WriteLine("Некорректный индекс для изменения яхт-клуба.");

}

}

catch (FormatException)

{

Console.WriteLine("Ошибка: Некорректный формат ввода.");

}

catch (Exception ex)

{

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

}

Console.WriteLine("Нажмите Enter для продолжения...");

Console.ReadLine();

}

}

**YachtClubController.cs:**

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

namespace lab\_1

{

internal class YachtClubController

{

private string filePath; // Путь к файлу CSV

public YachtClubController(string filePath)

{

this.filePath = filePath;

}

// Метод для чтения всех записей из файла и возврата списка объектов YachtClub

public List<YachtClub> ReadAllRecords()

{

List<YachtClub> yachtClubs = new List<YachtClub>();

try

{

using (StreamReader reader = new StreamReader(filePath))

{

string line;

while ((line = reader.ReadLine()) != null)

{

string[] data = line.Split(',');

// Проверяем, что данные в строке корректны, иначе пропускаем строку

if (data.Length >= 4)

{

YachtClub club = new YachtClub

{

Name = data[0],

Address = data[1],

NumberOfYachts = int.Parse(data[2]),

NumberOfPlaces = int.Parse(data[3]),

HasPool = bool.Parse(data[4])

};

yachtClubs.Add(club);

}

}

}

}

catch (IOException e)

{

Console.WriteLine($"Ошибка чтения файла: {e.Message}");

Console.WriteLine("Нажмите Enter для продолжения...");

Console.ReadLine();

}

return yachtClubs;

}

// Метод для записи списка объектов YachtClub в файл CSV

public void WriteRecords(List<YachtClub> yachtClubs)

{

try

{

using (StreamWriter writer = new StreamWriter(filePath))

{

foreach (YachtClub club in yachtClubs)

{

string line = $"{club.Name},{club.Address},{club.NumberOfYachts},{club.NumberOfPlaces},{club.HasPool}";

writer.WriteLine(line);

}

}

}

catch (IOException e)

{

Console.WriteLine($"Ошибка записи в файл: {e.Message}");

}

}

// Метод для добавления новой записи в список и сохранения в файле

public void AddRecord(YachtClub club)

{

List<YachtClub> yachtClubs = ReadAllRecords();

yachtClubs.Add(club);

WriteRecords(yachtClubs);

}

// Метод для удаления записи по индексу

public void RemoveRecord(int index)

{

List<YachtClub> yachtClubs = ReadAllRecords();

if (index >= 0 && index < yachtClubs.Count)

{

yachtClubs.RemoveAt(index);

WriteRecords(yachtClubs);

}

else

{

Console.WriteLine("Неверный индекс для удаления записи.");

}

}

}

}

**Пример выполнения программы**

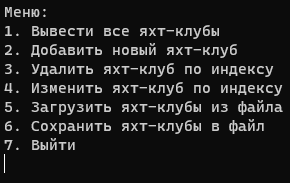


Рисунок - Меню.

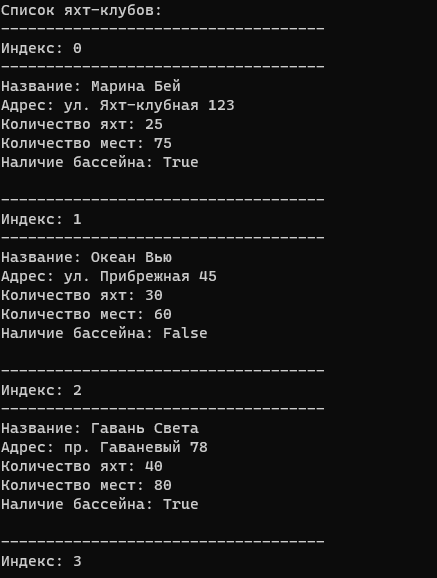


Рисунок - Список клубов.

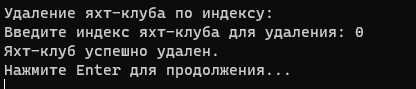


Рисунок - Удаление элемента.