Мусаев Тимур Русланович

**Отчёт.**

**Тема: "Работник" и "Предприятие"**

**Цель работы**: разработка программы для учета работников, компаний, и их взаимодействия

**Задачи**:

Описать классы "Работник" и "Предприятие" позволяющие:

создавать работников,

принимать,

увольнять,

изменять должность,

переводить на другое предприятие,

изменять зарплату.

Предусмотреть защиту от ввода некорректных данных для всех членов всех классов

Предусмотреть повторное предложение Пользователю на ввод данных, в случае их некорректности

Предусмотреть назначение Работникам уникальных идентификаторов

Предусмотреть выбор Предприятия на приема и переода работников

Написать класс для сохранения состояния программы в текстовом файле, на примере стоянки:

при изменении данных перезаписывать файл,

при новом запуске программы восстанавливать данные из файла

написать класс для логирования событий в текстовый файл

using System;

using System.Collections.Generic;

using System.IO;

namespace ConsoleApp3

{

class Работник

{

public string Имя { get; set; }

public string Должность { get; set; }

public double Зарплата { get; set; }

public Работник(string имя, string должность, double зарплата)

{

Имя = имя;

Должность = должность;

Зарплата = зарплата;

}

}

class Предприятие

{

private List<Работник> работники = new List<Работник>();

public void ПринятьРаботника(Работник работник)

{

работники.Add(работник);

}

public void УволитьРаботника(string имя)

{

Работник уволенныйРаботник = работники.Find(r => r.Имя == имя);

if (уволенныйРаботник != null)

{

работники.Remove(уволенныйРаботник);

}

}

public void ИзменитьДолжность(string имя, string новаяДолжность)

{

Работник работник = работники.Find(r => r.Имя == имя);

if (работник != null)

{

работник.Должность = новаяДолжность;

}

}

public void ПеревестиНаДругоеПредприятие(string имя, Предприятие новоеПредприятие)

{

Работник работник = работники.Find(r => r.Имя == имя);

if (работник != null)

{

УволитьРаботника(имя);

новоеПредприятие.ПринятьРаботника(работник);

}

}

public void ИзменитьЗарплату(string имя, double новаяЗарплата)

{

Работник работник = работники.Find(r => r.Имя == имя);

if (работник != null)

{

работник.Зарплата = новаяЗарплата;

}

}

public void ВывестиСписокРаботников()

{

Console.WriteLine("Список работников на предприятии:");

foreach (var работник in работники)

{

Console.WriteLine($"Имя: {работник.Имя}, Должность: {работник.Должность}, Зарплата: {работник.Зарплата}");

}

}

public void СохранитьСостояние(string fileName)

{

using (StreamWriter writer = new StreamWriter(fileName))

{

foreach (var работник in работники)

{

writer.WriteLine($"{работник.Имя}, {работник.Должность}, {работник.Зарплата}р");

}

}

}

public void ЗагрузитьСостояние(string fileName)

{

работники.Clear();

try

{

using (StreamReader reader = new StreamReader(fileName))

{

string line;

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

{

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

if (tokens.Length == 3)

{

string имя = tokens[0].Trim();

string должность = tokens[1].Trim();

double зарплата;

if (double.TryParse(tokens[2].Replace("р", "").Trim(), out зарплата))

{

Работник новыйРаботник = new Работник(имя, должность, зарплата);

ПринятьРаботника(новыйРаботник);

}

else

{

Console.WriteLine($"Ошибка при чтении зарплаты для строки: {line}");

}

}

else

{

Console.WriteLine($"Ошибка при чтении строки: {line}");

}

}

}

}

catch (Exception ex)

{

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

}

}

}

class Program

{

// Метод для получения пути к файлу в корневой папке проекта

private static string GetFilePathInProjectFolder(string fileName)

{

string projectFolder = AppDomain.CurrentDomain.BaseDirectory;

return Path.Combine(projectFolder, fileName);

}

static void Main()

{

Предприятие предприятие = new Предприятие();

// Загрузка состояния из файла при запуске программы

предприятие.ЗагрузитьСостояние(GetFilePathInProjectFolder("состояние.txt"));

while (true)

{

Console.WriteLine("Выберите действие:");

Console.WriteLine("1. Принять работника");

Console.WriteLine("2. Уволить работника");

Console.WriteLine("3. Изменить должность работника");

Console.WriteLine("4. Перевести на другое предприятие");

Console.WriteLine("5. Изменить зарплату работника");

Console.WriteLine("6. Вывести список работников");

Console.WriteLine("7. Сохранить состояние");

Console.WriteLine("8. Выход");

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

switch (выбор)

{

case 1:

Console.Write("Введите имя работника: ");

string имя = Console.ReadLine();

Console.Write("Введите должность: ");

string должность = Console.ReadLine();

Console.Write("Введите зарплату: ");

double зарплата = double.Parse(Console.ReadLine());

Работник новыйРаботник = new Работник(имя, должность, зарплата);

предприятие.ПринятьРаботника(новыйРаботник);

Console.WriteLine($"{имя} принят на работу.");

break;

case 2:

Console.Write("Введите имя увольняемого работника: ");

string имяУвольняемого = Console.ReadLine();

предприятие.УволитьРаботника(имяУвольняемого);

Console.WriteLine($"{имяУвольняемого} уволен.");

break;

case 3:

Console.Write("Введите имя работника, чью должность нужно изменить: ");

string имяРаботника = Console.ReadLine();

Console.Write("Введите новую должность: ");

string новаяДолжность = Console.ReadLine();

предприятие.ИзменитьДолжность(имяРаботника, новаяДолжность);

Console.WriteLine($"Должность работника {имяРаботника} изменена на {новаяДолжность}.");

break;

case 4:

Console.Write("Введите имя работника, которого переводите на другое предприятие: ");

string имяПереводимого = Console.ReadLine();

Console.Write("Введите имя нового предприятия: ");

string имяНовогоПредприятия = Console.ReadLine();

Предприятие новоеПредприятие = new Предприятие();

предприятие.ПеревестиНаДругоеПредприятие(имяПереводимого, новоеПредприятие);

Console.WriteLine($"{имяПереводимого} переведен на предприятие {имяНовогоПредприятия}.");

break;

case 5:

Console.Write("Введите имя работника, у которого нужно изменить зарплату: ");

string имяРаботникаЗарплата = Console.ReadLine();

Console.Write("Введите новую зарплату: ");

double новаяЗарплата = double.Parse(Console.ReadLine());

предприятие.ИзменитьЗарплату(имяРаботникаЗарплата, новаяЗарплата);

Console.WriteLine($"Зарплата работника {имяРаботникаЗарплата} изменена на {новаяЗарплата}.");

break;

case 6:

предприятие.ВывестиСписокРаботников();

break;

case 7:

предприятие.СохранитьСостояние(GetFilePathInProjectFolder("состояние.txt"));

Console.WriteLine("Состояние сохранено.");

break;

case 8:

// Сохранение состояния в файл перед выходом из программы

предприятие.СохранитьСостояние(GetFilePathInProjectFolder("состояние.txt"));

Environment.Exit(0);

break;

default:

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

break;

}

}

}

}

}