Алексеев Иван

1. Опишите классы "Автомобиль" и "Стоянка".

Класс "Автомобиль" должен содержать информацию о машине: Марку, Модель, Цвет, Номер, Время прибытия.

Добавьте возможность добавления машин на стоянку, просмотра информации о них и их вывоз.

1. Для всех программ предусмотреть текстовое меню, обеспечивающее доступ ко всем ее функциям
2. Предусмотреть защиту от ввода некорректных данных для всех членов всех классов
3. Предусмотреть повторное предложение Пользователю на ввод данных, в случае их некорректности
4. Предусмотреть назначение Работникам и Автомобилям уникальных идентификаторов
5. Предусмотреть выбор Предприятия на прием и перевод работников

3) Доработать программу, написать класс для сохранения состояния программы в текстовом файле, на примере стоянки:

место; модель; цвет; номер; дата прибытия

1; ваз 2107; белый; а123бв; 20:00 05.10.2023

2; ...

при изменении данных перезаписывать файл,

при новом запуске программы восстанавливать данные из файла

4) написать класс для логирования событий в текстовый файл, например:

20:00 05.10.2023: прибытие автомобиля ваз 2107

21:30 05.10.2023: убытие автомобиля ваз 2107

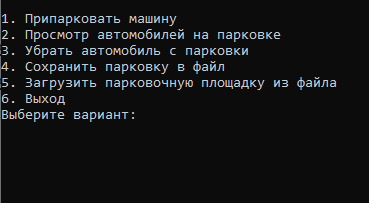


Рисунок 2 – текстовое меню

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace cars

{

class Car

{

public string Brand { get; set; }

public string Model { get; set; }

public string Color { get; set; }

public string Number { get; set; }

public DateTime ArrivalTime { get; set; }

public Car(string brand, string model, string color, string number)

{

Brand = brand;

Model = model;

Color = color;

Number = number;

ArrivalTime = DateTime.Now;

}

}

class ParkingLot

{

private Dictionary<int, Car> cars = new Dictionary<int, Car>();

public void AddCar(Car car)

{

int id = car.GetHashCode();

cars.Add(id, car);

}

public void ViewCars()

{

foreach (var pair in cars)

{

Console.WriteLine($"ID: {pair.Key}, Марка: {pair.Value.Brand}, Модель: {pair.Value.Model}, Цвет: {pair.Value.Color}, Номер: {pair.Value.Number}, Время прибытия: {pair.Value.ArrivalTime}");

}

}

public void RemoveCar(int id)

{

if (cars.ContainsKey(id))

{

cars.Remove(id);

Console.WriteLine($"Автомобиль с идентификатором {id} был удален с парковки.");

}

else

{

Console.WriteLine($"Автомобиль с идентификатором {id} не найден на парковке.");

}

}

public void SaveToFile(string filename)

{

using (StreamWriter writer = new StreamWriter(filename))

{

foreach (var pair in cars)

{

writer.WriteLine($"{pair.Key}; {pair.Value.Model}; {pair.Value.Color}; {pair.Value.Number}; {pair.Value.ArrivalTime}");

}

}

}

public void LoadFromFile(string filename)

{

if (File.Exists(filename))

{

string[] lines = File.ReadAllLines(filename);

foreach (string line in lines)

{

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

if (data.Length == 5)

{

int id = int.Parse(data[0].Trim());

string model = data[1].Trim();

string color = data[2].Trim();

string number = data[3].Trim();

DateTime arrivalTime = DateTime.Parse(data[4].Trim());

Car car = new Car("", model, color, number)

{

ArrivalTime = arrivalTime

};

cars.Add(id, car);

}

}

}

}

}

class Program

{

static void Main()

{

ParkingLot parkingLot = new ParkingLot();

while (true)

{

Console.WriteLine("\n1. Припарковать машину");

Console.WriteLine("2. Просмотр автомобилей на парковке");

Console.WriteLine("3. Убрать автомобиль с парковки");

Console.WriteLine("4. Сохранить парковку в файл");

Console.WriteLine("5. Загрузить парковочную площадку из файла");

Console.WriteLine("6. Выход");

Console.Write("Выберите вариант: ");

string choice = Console.ReadLine();

switch (choice)

{

case "1":

try

{

Console.Write("Введите марку автомобиля: ");

string brand = Console.ReadLine();

Console.Write("Введите модель автомобиля: ");

string model = Console.ReadLine();

Console.Write("Введите цвет автомобиля: ");

string color = Console.ReadLine();

Console.Write("Введите номер автомобиля: ");

string number = Console.ReadLine();

Car car = new Car(brand, model, color, number);

parkingLot.AddCar(car);

Console.WriteLine("Автомобиль успешно добавлен на парковку.");

}

catch (Exception e)

{

Console.WriteLine($"Ошибка при добавлении автомобиля: {e.Message}");

}

break;

case "2":

parkingLot.ViewCars();

break;

case "3":

try

{

Console.Write("Введите идентификатор автомобиля для удаления: ");

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

parkingLot.RemoveCar(id);

}

catch (FormatException)

{

Console.WriteLine("Неверный ввод. Пожалуйста, введите правильное исло.");

}

break;

case "4":

Console.Write("Введите имя файла для сохранения парковки: ");

string saveFileName = Console.ReadLine();

parkingLot.SaveToFile(saveFileName);

break;

case "5":

Console.Write("Введите имя файла для загрузки парковки: ");

string loadFileName = Console.ReadLine();

parkingLot.LoadFromFile(loadFileName);

break;

case "6":

Environment.Exit(0);

break;

default:

Console.WriteLine("Неверный выбор. Пожалуйста, выберите правильный вариант.");

break;

}

}

}

}

}