Факультет компьютерных технологий

Лабораторная работа №4

«Современные платформы программирования»

Вариант №10

Выполнил: студент гр. 981063 Ефименко Павел Викторович

Проверил: Воробей В.А.

Минск 2021

Условие:

Решить задачу, в соответствии с вариантом подгруппы. В качестве среды разработки использовать IntelliJ IDEA. В разработке алгоритма использовать строго типизированный объектно-ориентированный язык программирования Java. 10. В базе bus.dat хранятся записи, содержащие: номер маршрута, начало маршрута, конец маршрута, тип автобуса, количество автобусов на линии, номер автобазы. Обеспечить выдачу следующих справок: - определить общее число городских автобусов; - по номеру автобуса определить его маршрут; - выдать список номеров автобусов, маршруты которых проходят через заданный пункт; - выдать список маршрутов, которые обслуживает автобаза с указанным номером.

Код:

namespace Laba\_4

{

public interface IRoute

{

int Number { get; set; }

}

}

using System.Collections.Generic;

namespace Laba\_4

{

public interface IBusDat

{

List<IBus> Buses { get; set; }

string CountsBusOnRoute { get; }

IEnumerable<int> GetBusNumberByRoute(IRoute route);

int GetCountCityBus();

IRoute GetRouteByBusNumber(int busNumber);

IEnumerable<IRoute> GetRoutesByAutoBaseNumber(int autoBaseNumber);

}

}

namespace Laba\_4

{

public interface IBus

{

int AutoBaseNumber { get; set; }

int Number { get; set; }

IRoute Route { get; set; }

string RouteEnd { get; set; }

string RouteStart { get; set; }

BusType Type { get; set; }

}

}

using System;

namespace Laba\_4

{

internal class Route : IRoute

{

public int Number { get; set; }

public override string ToString()

{

return $"{Number}";

}

public static Route Fake()

{

var rnd = new Random();

return new Route { Number = rnd.Next(10) };

}

}

}

using Newtonsoft.Json;

using System;

using System.IO;

namespace Laba\_4

{

public static class Helper

{

public static string ToJson(this IBus model)

{

return JsonConvert.SerializeObject(model);

}

public static void Save(this IBusDat busDat)

{

var serializeData = JsonConvert.SerializeObject(busDat, Formatting.Indented);

File.WriteAllText(BusDat.BusDatFilePath, serializeData);

Console.WriteLine("Успешно сохранено");

}

public static void PrintDatFile(this IBusDat busDat)

{

var fileData = File.ReadAllText(BusDat.BusDatFilePath);

Console.WriteLine(fileData);

}

}

}

namespace Laba\_4

{

public enum BusType

{

City,

Region,

Country

}

}

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

namespace Laba\_4

{

internal class BusDat : IBusDat

{

public const string BusDatFilePath = "bus\_dat.json";

public List<IBus> Buses { get; set; }

public string CountsBusOnRoute { get => CountBusOnRoute(); }

public int GetCountCityBus()

{

return Buses.Count(bus => bus.Type == BusType.City);

}

public IRoute GetRouteByBusNumber(int busNumber)

{

var bus = Buses.First(bus => bus.Number == busNumber);

return bus.Route;

}

public IEnumerable<int> GetBusNumberByRoute(IRoute route)

{

return Buses.Where(bus => bus.Route == route).Select(bus => bus.Number);

}

public IEnumerable<IRoute> GetRoutesByAutoBaseNumber(int autoBaseNumber)

{

return Buses.Where(bus => bus.AutoBaseNumber == autoBaseNumber).Select(bus => bus.Route);

}

public static IList<IBus> FakeBuses(byte count)

{

return Enumerable.Range(0, count).Select(bus => Bus.Fake()).ToList();

}

private string CountBusOnRoute()

{

var result = string.Empty;

var query = Buses.GroupBy(bus => bus.Route.Number, bus => bus);

foreach (var busgroup in query)

{

int busCountInGroup = busgroup.Count();

result += $"{busgroup.Key}:{busCountInGroup}\n";

Console.WriteLine($"Номер маршрута:\t{busgroup.Key}\tКоличество автобусов на маршруте:\t{busCountInGroup}");

}

return result;

}

}

}

using System;

namespace Laba\_4

{

internal class Bus : IBus

{

public int Number { get; set; }

public IRoute Route { get; set; }

public string RouteStart { get; set; }

public string RouteEnd { get; set; }

public BusType Type { get; set; }

public int AutoBaseNumber { get; set; }

public override string ToString()

{

return $"Номер автобуса:\t{Number}\n" +

$"Маршрут:\t{Route}\n" +

$"Начало маршрута:{RouteStart}\nКонец маршрута:\t{RouteEnd}\n" +

$"Тип автобуса:\t{Type}\n" +

$"Номер автобазы:\t{AutoBaseNumber}\n";

}

public static IBus Fake()

{

var rnd = new Random();

return new Bus

{

AutoBaseNumber = Faker.NumberFaker.Number(),

Type = (BusType)rnd.Next((int)BusType.Country + 1),

RouteEnd = Faker.LocationFaker.Street(),

RouteStart = Faker.LocationFaker.Street(),

Route = Laba\_4.Route.Fake(),

Number = Faker.NumberFaker.Number()

};

}

}

}

using System;

using System.Linq;

namespace Laba\_4

{

class Program

{

static void Main(string[] args)

{

var buses = BusDat.FakeBuses(10).ToList();

var busDat = new BusDat { Buses = buses };

Console.WriteLine(string.Join('\n', buses));

Console.WriteLine($"Количество автобусов на маршруте:\n{busDat.CountsBusOnRoute}\n");

Console.WriteLine($"Cписок маршрутов, которые обслуживает автобаза с указанным номером \"{buses.First().AutoBaseNumber}\":" +

$"\n{string.Join('\n', busDat.GetRoutesByAutoBaseNumber(buses.First().AutoBaseNumber))}\n");

Console.WriteLine($"Выдать список номеров автобусов, маршруты которых проходят через заданный пункт \"{buses.First().Route}\":" +

$"\n{string.Join('\n', busDat.GetBusNumberByRoute(buses.First().Route))}\n");

Console.WriteLine($"По номеру автобуса определить его маршрут \"{buses.Last().Number}\":" +

$"\n{busDat.GetRouteByBusNumber(buses.Last().Number)}\n");

Console.WriteLine($"Определить общее число городских автобусов:" +

$"\n{busDat.GetCountCityBus()}\n");

busDat.Save();

busDat.PrintDatFile();

}

}

}

Результат:

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |