**Отчет**

**Лабораторная работа №4**

**С дисциплины: «C#»**

**Тема: «** **LINQ to Objects »**

**Автор: Ярцев Д.А.**

**Группа: ИС61**

**Цель**: - ознакомиться с обработкой данных с использованием библиотеки LINQ to Objects.

**LINQ –** это модель для работы с данными в различных видах источников данных и в различных форматах. Все операции запроса LINQ можно разбить на три различные группы:

1) Получение источника данных.

2) Создание запроса.

3) Выполнение запроса

**Вариант №6**

Разработать структуру данных для хранения информации о троллейбусных маршрутах города. Для каждого маршрута хранится информация:

1. наименование начальной и конечной остановки,
2. количество троллейбусов на маршруте,
3. время проезда от начала маршрута до конца,
4. список номеров троллейбусов на маршруте.

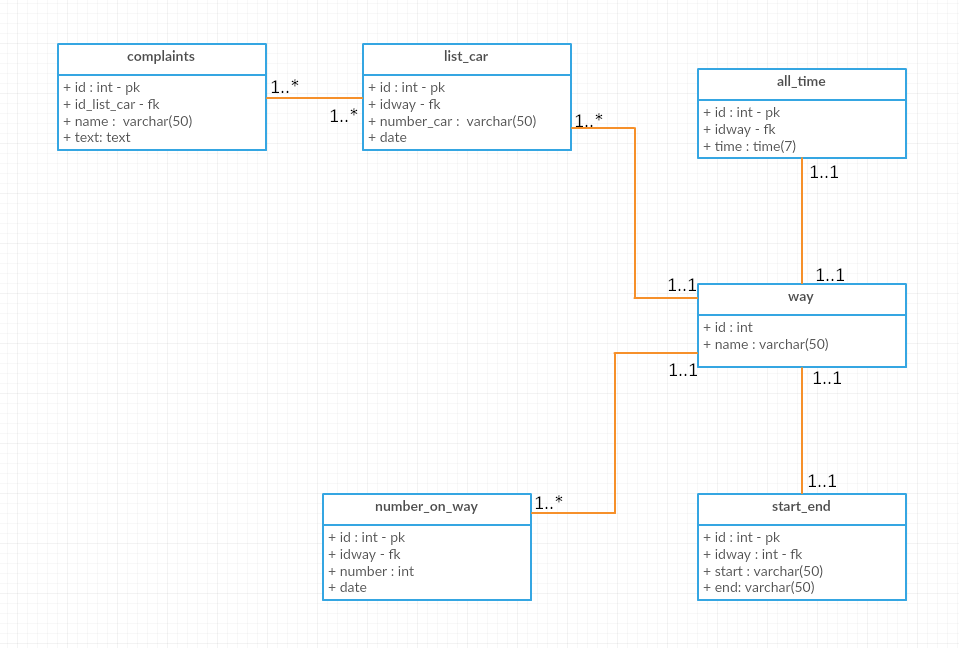
Должно быть как минимум 3-**4** класса.

Должны быть продемонстрированы связи между классами:

1. один-ко-многим
2. много-ко-многим.

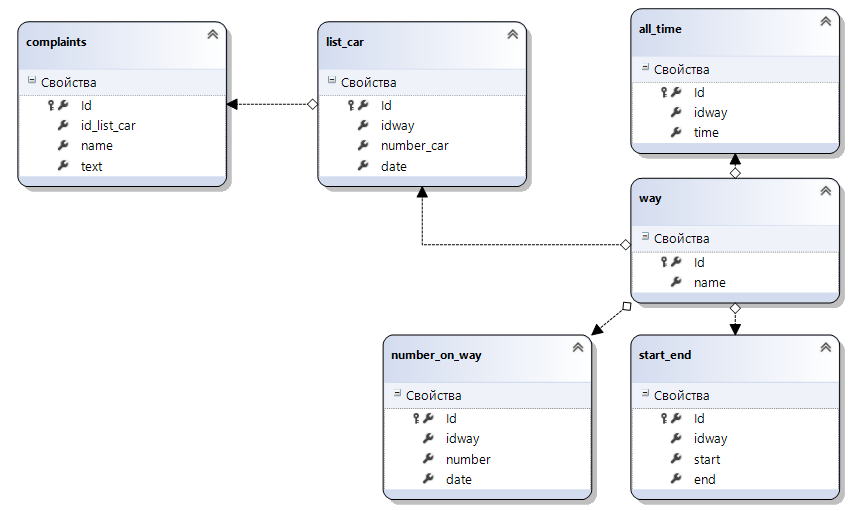
**Ход работы**

Сперва я начертил uml диаграму



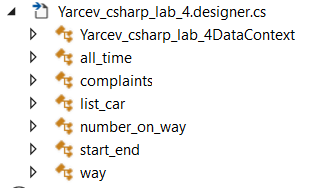
*UML Диаграмма*

Потом посльзуясь этой диаграмой создал локальную базу данных в visual studio 2017(структура данных привидена ниже )



Структура данных

После воспользовался модулем linq to sql и создал экзаемляр контекста с которым буду работат.



В нем для каждой таблицы используется отдельный класс (их 7).

Yarcev\_csharp\_lab\_4DataContext context = new Yarcev\_csharp\_lab\_4DataContext();

1. Всего будет 7 различных запросов, используя различные действия над множествами:
2. группировку,
3. сортировку,
4. фильтрацию,
5. объединение результатов нескольких запросов в один (join, concat)
6. и т.д.

А именно:

1. узнать сколько было машин на пути idway='X' в разные дни.
2. узнать полный список маршрутов в алфавитном порядке.
3. узнать когда на маршруте id='X' было меньше 'Y' машин.
4. узнать Сколько было жалоб поступило на каждую машину.
5. узнать сколько раз машина number\_car='X' была на маршруте 'Y'.
6. Вывести имена маршрутов с временем начала и конца маршрута.
7. Вывести номер маршрутки и имена людей, что на нее пожаловались.

Общий алгоритм работы программы таков:

1. меню запросов
2. выбор запроса
3. корекция запроса
4. выполнение запроса
5. завершение программы

**Список запросов**

1. узнать сколько было машин на пути idway='X' в разные дни.
2. узнать полный список маршрутов в алфавитном порядке.
3. узнать когда на маршруте id='X' было меньше 'Y' машин.
4. узнать Сколько было жалоб поступило на каждую машину.
5. узнать сколько раз машина number\_car='X' была на маршруте 'Y'.
6. Вывести имена маршрутов с временем начала и конца маршрута.
7. Вывести номер маршрутки и имена людей, что на нее пожаловались.

**Разбор программы, реализующей обработку данных с использованием библиотеки LINQ to Objects**

Для управления использовал цикл с меню.

Функционал реализовал через паттерн FAÇADE.

Для составления запросов использовал linQ.

**Реализация: вывод в консоль**

**Lab 4 author: Yarcev D.A., group: IS-61**

**Variant 6**

**================================================================================**

**menu:**

**1 - узнать сколько было машин на пути idway='X' в разные дни:**

**2 - узнать полный список маршрутов в алфавитном порядке:**

**3 - узнать когда на маршруте id='X' было меньше 'Y' машин:**

**4 - узнать Сколько было жалоб поступило на каждую машину:**

**5 - узнать сколько раз машина number\_car='X' была на маршруте 'Y':**

**6 - Вывести имена маршрутов с временем начала и конца маршрута:**

**7 - Вывести номер маршрутки и имена людей, что на нее пожаловались:**

**--------------------------------------------------------------------------------**

**<= Input number: 1**

**--------------------------------------------------------------------------------**

**(Используется фильтрация и сортировка)**

**<= Введите idway пути: 1**

**SELECT ID, IDWAY, NUMBER, DATA**

**FROM NUMBER\_ON\_WAY**

**WHERE IDWAY = '1'**

**ORDER BY DATA ASC;**

**{ID, IDway, NUMBER, DATA}(**

**(1) (1) (2) (01.01.2018 00:00:00),**

**(22) (1) (1) (02.01.2018 00:00:00),**

**);**

**--------------------------------------------------------------------------------**

**<= Are you want to rareat this promram? ('y' - yes/'n' - no): y**

**--------------------------------------------------------------------------------**

**menu:**

**1 - узнать сколько было машин на пути idway='X' в разные дни:**

**2 - узнать полный список маршрутов в алфавитном порядке:**

**3 - узнать когда на маршруте id='X' было меньше 'Y' машин:**

**4 - узнать Сколько было жалоб поступило на каждую машину:**

**5 - узнать сколько раз машина number\_car='X' была на маршруте 'Y':**

**6 - Вывести имена маршрутов с временем начала и конца маршрута:**

**7 - Вывести номер маршрутки и имена людей, что на нее пожаловались:**

**--------------------------------------------------------------------------------**

**<= Input number: 2**

**--------------------------------------------------------------------------------**

**(сортировка по убыванию)**

**SELECT ID, NAME**

**WAY**

**ORDER BY ID ASC;**

**{ID, NAME}(**

**(0) (way\_001),**

**(1) (way\_002),**

**(2) (way\_003),**

**(3) (way\_004),**

**(4) (way\_005),**

**(5) (way\_006),**

**(6) (way\_007),**

**(7) (way\_008),**

**(8) (way\_009),**

**(9) (way\_010),**

**(10) (way\_011),**

**(11) (way\_012),**

**(12) (way\_013),**

**(13) (way\_014),**

**(14) (way\_015),**

**(15) (way\_016),**

**(16) (way\_017),**

**(17) (way\_018),**

**(18) (way\_019),**

**(19) (way\_020),**

**(20) (way\_021),**

**);**

**--------------------------------------------------------------------------------**

**<= Are you want to rareat this promram? ('y' - yes/'n' - no): y**

**--------------------------------------------------------------------------------**

**menu:**

**1 - узнать сколько было машин на пути idway='X' в разные дни:**

**2 - узнать полный список маршрутов в алфавитном порядке:**

**3 - узнать когда на маршруте id='X' было меньше 'Y' машин:**

**4 - узнать Сколько было жалоб поступило на каждую машину:**

**5 - узнать сколько раз машина number\_car='X' была на маршруте 'Y':**

**6 - Вывести имена маршрутов с временем начала и конца маршрута:**

**7 - Вывести номер маршрутки и имена людей, что на нее пожаловались:**

**--------------------------------------------------------------------------------**

**<= Input number: 3**

**--------------------------------------------------------------------------------**

**(фильтрация, сортировка)**

**<= Введите idway пути: 3**

**<= Введите num машин: 3**

**SELECT ID, IDWAY, NUMBER, DATA**

**FROM NUMBER\_ON\_WAY**

**WHERE IDWAY = '3' && NUMBER > 3**

**ORDER BY DATA ASC;**

**{ID, IDway, NUMBER, DATA}(**

**(3) (3) (1) (01.01.2018 00:00:00),**

**(24) (3) (0) (02.01.2018 00:00:00),**

**);**

**--------------------------------------------------------------------------------**

**<= Are you want to rareat this promram? ('y' - yes/'n' - no): y**

**--------------------------------------------------------------------------------**

**menu:**

**1 - узнать сколько было машин на пути idway='X' в разные дни:**

**2 - узнать полный список маршрутов в алфавитном порядке:**

**3 - узнать когда на маршруте id='X' было меньше 'Y' машин:**

**4 - узнать Сколько было жалоб поступило на каждую машину:**

**5 - узнать сколько раз машина number\_car='X' была на маршруте 'Y':**

**6 - Вывести имена маршрутов с временем начала и конца маршрута:**

**7 - Вывести номер маршрутки и имена людей, что на нее пожаловались:**

**--------------------------------------------------------------------------------**

**<= Input number: 4**

**--------------------------------------------------------------------------------**

**(групировка и сортировка)**

**SELECT NUMBER\_CAR, COUNT(\*)**

**FROM COMPLAINTS S1 JOIN LIST\_CAR S2 ON S1.ID\_LIST\_CAR = S2.ID**

**GROUP BY NUMBER\_CAR;**

**{NUMBER\_CAR, COUNT(\*)}(**

**(00-00-00FB) (3),**

**(00-00-01FB) (3),**

**);**

**--------------------------------------------------------------------------------**

**<= Are you want to rareat this promram? ('y' - yes/'n' - no): 5**

**=> Erorr.**

**<= Please rapeat you answer ('y' - yes/'n' - no): y**

**--------------------------------------------------------------------------------**

**menu:**

**1 - узнать сколько было машин на пути idway='X' в разные дни:**

**2 - узнать полный список маршрутов в алфавитном порядке:**

**3 - узнать когда на маршруте id='X' было меньше 'Y' машин:**

**4 - узнать Сколько было жалоб поступило на каждую машину:**

**5 - узнать сколько раз машина number\_car='X' была на маршруте 'Y':**

**6 - Вывести имена маршрутов с временем начала и конца маршрута:**

**7 - Вывести номер маршрутки и имена людей, что на нее пожаловались:**

**--------------------------------------------------------------------------------**

**<= Input number: 5**

**--------------------------------------------------------------------------------**

**(групировка и фильтр)**

**<= Введите number\_car машины(00-00-06FB): 00-00-06FB**

**<= Введите idway(2): 2**

**SELECT NUMBER\_CAR, COUNT(\*)**

**FROM LIST\_CAR**

**GROUP BY NUMBER\_CAR;**

**{NUMBER\_CAR, COUNT(\*)}(**

**(00-00-06FB) (1),**

**);**

**--------------------------------------------------------------------------------**

**<= Are you want to rareat this promram? ('y' - yes/'n' - no): y**

**--------------------------------------------------------------------------------**

**menu:**

**1 - узнать сколько было машин на пути idway='X' в разные дни:**

**2 - узнать полный список маршрутов в алфавитном порядке:**

**3 - узнать когда на маршруте id='X' было меньше 'Y' машин:**

**4 - узнать Сколько было жалоб поступило на каждую машину:**

**5 - узнать сколько раз машина number\_car='X' была на маршруте 'Y':**

**6 - Вывести имена маршрутов с временем начала и конца маршрута:**

**7 - Вывести номер маршрутки и имена людей, что на нее пожаловались:**

**--------------------------------------------------------------------------------**

**<= Input number: 6**

**--------------------------------------------------------------------------------**

**(join и сортировка, concat)**

**SELECT NAME, START, END**

**FROM WAY S1 JOIN STARY\_END S2 ON S1.IDWAY = S2.ID**

**ORDER BY NAME;**

**{NAME, START\_END}(**

**(way\_001) (station\_1station\_12),**

**(way\_002) (station\_2station\_13),**

**(way\_003) (station\_3station\_12),**

**(way\_004) (station\_4station\_15),**

**(way\_005) (station\_5station\_12),**

**(way\_006) (station\_6station\_17),**

**(way\_007) (station\_7station\_18),**

**(way\_008) (station\_8station\_13),**

**(way\_009) (station\_9station\_20),**

**(way\_010) (station\_10station\_1 ),**

**(way\_011) (station\_11station\_1 ),**

**(way\_012) (station\_12station\_2 ),**

**(way\_013) (station\_13station\_13),**

**(way\_014) (station\_14station\_4 ),**

**(way\_015) (station\_15station\_15),**

**(way\_016) (station\_16station\_6 ),**

**(way\_017) (station\_17station\_17),**

**(way\_018) (station\_18station\_8 ),**

**(way\_019) (station\_19station\_9 ),**

**(way\_020) (station\_20station\_10),**

**(way\_021) (station\_21station\_11),**

**);**

**--------------------------------------------------------------------------------**

**<= Are you want to rareat this promram? ('y' - yes/'n' - no): 7**

**=> Erorr.**

**<= Please rapeat you answer ('y' - yes/'n' - no): y**

**--------------------------------------------------------------------------------**

**menu:**

**1 - узнать сколько было машин на пути idway='X' в разные дни:**

**2 - узнать полный список маршрутов в алфавитном порядке:**

**3 - узнать когда на маршруте id='X' было меньше 'Y' машин:**

**4 - узнать Сколько было жалоб поступило на каждую машину:**

**5 - узнать сколько раз машина number\_car='X' была на маршруте 'Y':**

**6 - Вывести имена маршрутов с временем начала и конца маршрута:**

**7 - Вывести номер маршрутки и имена людей, что на нее пожаловались:**

**--------------------------------------------------------------------------------**

**<= Input number: 7**

**--------------------------------------------------------------------------------**

**(join и группировка)**

**SELECT NUMBER\_CAR OVER(PARTITION BY NUMBER\_CAR), NAME, END**

**FROM CON S1 JOIN STARY\_END S2 ON S1.ID\_LIST\_CAR = S2.ID**

**{NUMBER\_CAR, NAME}(**

**(00-00-00FB:**

**Fedor,**

**Dmitry,**

**Oleg,**

**),**

**(00-00-01FB:**

**Konstantin,**

**Maria,**

**Olga,**

**),**

**);**

**--------------------------------------------------------------------------------**

**<= Are you want to rareat this promram? ('y' - yes/'n' - no): n**

**--------------------------------------------------------------------------------**

**--------------------------------------------------------------------------------**

**Program is ending...**

**================================================================================**

**Реализация: код**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Yarcev\_csharp\_lab\_4

{

/// <summary>

/// основной класс программы

/// </summary>

class Program

{

//рисует линию из символа 'ы'

static void LineCh(char s)

{

for (int i = 0; i < 80; i++)

Console.Write(s);

Console.WriteLine();

}

//основная функция программы

static void Main(string[] args)

{

Console.WriteLine(" Lab 4 author: Yarcev D.A., group: IS-61");

Console.WriteLine(" Variant 6");

LineCh('=');

//создаем новый экземпляр контекста

//ключ основного цикла программы и сам цикл

bool keyLoop = true;

while (keyLoop)

{

//основной алгоритм программы

bool key = true;

Facade go = new Facade();

int i1 = 0;

while (key)

{

key = false;

Console.WriteLine(" menu:");

Console.WriteLine(" 1 - узнать сколько было машин на пути idway='X' в разные дни:");

Console.WriteLine(" 2 - узнать полный список маршрутов в алфавитном порядке:");

Console.WriteLine(" 3 - узнать когда на маршруте id='X' было меньше 'Y' машин:");

Console.WriteLine(" 4 - узнать Сколько было жалоб поступило на каждую машину:");

Console.WriteLine(" 5 - узнать сколько раз машина number\_car='X' была на маршруте 'Y':");

Console.WriteLine(" 6 - Вывести имена маршрутов с временем начала и конца маршрута:");

Console.WriteLine(" 7 - Вывести номер маршрутки и имена людей, что на нее пожаловались:");

LineCh('-');

Console.Write(" <= Input number: ");

i1 = Int32.Parse(Console.ReadLine());

LineCh('-');

switch (i1)

{

case 1: go.Subsystem\_1(); break;

case 2: go.Subsystem\_2(); break;

case 3: go.Subsystem\_3(); break;

case 4: go.Subsystem\_4(); break;

case 5: go.Subsystem\_5(); break;

case 6: go.Subsystem\_6(); break;

case 7: go.Subsystem\_7(); break;

default:

Console.WriteLine(" Error");

LineCh('-');

key = true;

break;

}

}

//вопрос о завершении программы и его ключ

LineCh('-');

string answer;

bool keyEndCycle = true;

Console.Write(" <= Are you want to rareat this promram? ('y' - yes/'n' - no): ");

while (keyEndCycle)

{

answer = Console.ReadLine().ToString();

if (answer == "y" || answer == "yes" || answer == "Yes" || answer == "YES")

{

keyLoop = true;

keyEndCycle = false;

}

else if (answer == "n" || answer == "no" || answer == "No" || answer == "NO")

{

keyLoop = false;

keyEndCycle = false;

}

else

{

Console.Write(" => Erorr.\n <= Please rapeat you answer ('y' - yes/'n' - no): ");

keyEndCycle = true;

}

}

LineCh('-');

}

LineCh('-');

Console.WriteLine(" Program is ending...");

LineCh('=');

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Yarcev\_csharp\_lab\_4

{

/// <summary>

/// управляющий класс

/// </summary>

class Facade

{

private Subsystem\_1 sub1 = new Subsystem\_1();

private Subsystem\_2 sub2 = new Subsystem\_2();

private Subsystem\_3 sub3 = new Subsystem\_3();

private Subsystem\_4 sub4 = new Subsystem\_4();

private Subsystem\_5 sub5 = new Subsystem\_5();

private Subsystem\_6 sub6 = new Subsystem\_6();

private Subsystem\_7 sub7 = new Subsystem\_7();

public void Subsystem\_1() { this.sub1.Procedutre(); }

public void Subsystem\_2() { this.sub2.Procedutre(); }

public void Subsystem\_3() { this.sub3.Procedutre(); }

public void Subsystem\_4() { this.sub4.Procedutre(); }

public void Subsystem\_5() { this.sub5.Procedutre(); }

public void Subsystem\_6() { this.sub6.Procedutre(); }

public void Subsystem\_7() { this.sub7.Procedutre(); }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Yarcev\_csharp\_lab\_4

{

/// <summary>

/// узнать сколько было машин на пути idway='X' в разные дни

/// </summary>

class Subsystem\_1

{

public void Procedutre()

{

Yarcev\_csharp\_lab\_4DataContext context = new Yarcev\_csharp\_lab\_4DataContext();

Console.WriteLine(" (Используется фильтрация и сортировка)");

int int\_idway = 0;

Console.Write(" <= Введите idway пути: ");

int\_idway = Int32.Parse(Console.ReadLine());

Console.WriteLine("");

var Groups2 = from n in context.number\_on\_way

where n.idway == int\_idway

select new {

ID = n.Id,

IDway = n.idway,

NUMBER = n.number,

DATA = n.date

};

Console.WriteLine(" SELECT ID, IDWAY, NUMBER, DATA");

Console.WriteLine(" FROM NUMBER\_ON\_WAY");

Console.WriteLine(" WHERE IDWAY = '"+int\_idway+"'");

Console.WriteLine(" ORDER BY DATA ASC;\n");

Console.WriteLine(" {ID, IDway, NUMBER, DATA}(");

foreach (var group in Groups2)

{

Console.WriteLine(" (" + group.ID.ToString()

+ ") (" + group.IDway.ToString() + ") (" + group.NUMBER.ToString()

+ ") (" + group.DATA.ToString() + "),");

}

Console.WriteLine(");");

//------------------------------------------------------------------

/\*

Yarcev\_csharp\_lab\_4DataContext context = new Yarcev\_csharp\_lab\_4DataContext();

Console.WriteLine(" (Используется фильтрация и сортировка по убыванию)");

int int\_idway = 0;

Console.Write(" Введите id пути: ");

int\_idway = Int32.Parse(Console.ReadLine());

var Groups2 = from n in context.number\_on\_way

where n.idway == int\_idway

group n by n.idway into g

select new { SUM = g.Sum(d => d.number), ID = g.Key };

foreach (var group in Groups2)

Console.WriteLine(" id={0} : sum={1}", group.SUM, group.ID);

\*/

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Yarcev\_csharp\_lab\_4

{

/// <summary>

/// узнать полный список маршрутов в алфавитном порядке:

/// </summary>

class Subsystem\_2

{

public void Procedutre()

{

Yarcev\_csharp\_lab\_4DataContext context = new Yarcev\_csharp\_lab\_4DataContext();

Console.WriteLine(" (сортировка по убыванию)");

var rows = from n in context.way

orderby n.Id

select new {ID = n.Id, NAME = n.name };

Console.WriteLine("\n SELECT ID, NAME");

Console.WriteLine(" WAY");

Console.WriteLine(" ORDER BY ID ASC;\n");

Console.WriteLine(" {ID, NAME}(");

foreach (var i in rows)

{

Console.WriteLine(" ("+i.ID.ToString()+") ("+i.NAME.ToString()+"),");

}

Console.WriteLine(");");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Yarcev\_csharp\_lab\_4

{

/// <summary>

/// узнать когда на маршруте id='X' было меньше 'Y' машин

/// </summary>

class Subsystem\_3

{

public void Procedutre()

{

Yarcev\_csharp\_lab\_4DataContext context = new Yarcev\_csharp\_lab\_4DataContext();

Console.WriteLine(" (фильтрация, сортировка)");

Console.Write(" <= Введите idway пути: ");

int in\_idway = Int32.Parse(Console.ReadLine());

Console.Write(" <= Введите num машин: ");

int in\_num = Int32.Parse(Console.ReadLine());

var rows = from n in context.number\_on\_way

where n.idway == in\_idway && n.number < in\_num

orderby n.Id

select new

{

ID = n.Id,

IDway = n.idway,

NUMBER = n.number,

DATA = n.date

};

Console.WriteLine(" SELECT ID, IDWAY, NUMBER, DATA");

Console.WriteLine(" FROM NUMBER\_ON\_WAY");

Console.WriteLine(" WHERE IDWAY = '" + in\_idway + "' && NUMBER > "+in\_num);

Console.WriteLine(" ORDER BY DATA ASC;\n");

Console.WriteLine(" {ID, IDway, NUMBER, DATA}(");

foreach (var group in rows)

{

Console.WriteLine(" (" + group.ID.ToString()

+ ") (" + group.IDway.ToString() + ") (" + group.NUMBER.ToString()

+ ") (" + group.DATA.ToString() + "),");

}

Console.WriteLine(");");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Yarcev\_csharp\_lab\_4

{

/// <summary>

/// узнать Сколько было жалоб поступило на каждую машину

/// </summary>

class Subsystem\_4

{

public void Procedutre()

{

Yarcev\_csharp\_lab\_4DataContext context = new Yarcev\_csharp\_lab\_4DataContext();

Console.WriteLine(" (групировка и сортировка)");

var rows = from n in context.complaints

join s in context.list\_car on n.id\_list\_car equals s.Id

group n by s.number\_car into g

select new {

NUMBER\_CAR = g.Key,

COUNT = g.Count()

};

Console.WriteLine("\n SELECT NUMBER\_CAR, COUNT(\*)");

Console.WriteLine(" FROM COMPLAINTS S1 JOIN LIST\_CAR S2 ON S1.ID\_LIST\_CAR = S2.ID");

Console.WriteLine(" GROUP BY NUMBER\_CAR;\n");

Console.WriteLine(" {NUMBER\_CAR, COUNT(\*)}(");

foreach (var group in rows)

{

Console.WriteLine(" (" + group.NUMBER\_CAR.ToString()

+ ") (" + group.COUNT.ToString() + "),");

}

Console.WriteLine(");");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Yarcev\_csharp\_lab\_4

{

/// <summary>

/// узнать сколько раз машина number\_car='X' была на маршруте 'Y'

/// </summary>

class Subsystem\_5

{

public void Procedutre()

{

Yarcev\_csharp\_lab\_4DataContext context = new Yarcev\_csharp\_lab\_4DataContext();

Console.WriteLine(" (групировка и фильтр)");

Console.Write(" <= Введите number\_car машины(00-00-06FB): ");

string in\_number\_car = Console.ReadLine();

Console.Write(" <= Введите idway(2): ");

int in\_idway = Int32.Parse(Console.ReadLine());

var rows = from n in context.list\_car

where n.idway == in\_idway && n.number\_car == in\_number\_car

group n by n into g

select new

{

NUMBER\_CAR = g.Key.number\_car,

COUNT = g.Count()

};

Console.WriteLine("\n SELECT NUMBER\_CAR, COUNT(\*)");

Console.WriteLine(" FROM LIST\_CAR");

Console.WriteLine(" GROUP BY NUMBER\_CAR;\n");

Console.WriteLine(" {NUMBER\_CAR, COUNT(\*)}(");

foreach (var group in rows)

{

Console.WriteLine(" (" + group.NUMBER\_CAR

+ ") (" + group.COUNT.ToString() + "),");

}

Console.WriteLine(");");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Yarcev\_csharp\_lab\_4

{

/// <summary>

/// Вывести имена маршрутов с временем начала и конца маршрута:

/// </summary>

class Subsystem\_6

{

public void Procedutre()

{

Yarcev\_csharp\_lab\_4DataContext context = new Yarcev\_csharp\_lab\_4DataContext();

Console.WriteLine(" (join и сортировка, concat)");

var rows = from se in context.start\_end

join w in context.way on se.idway equals w.Id

select new

{

WAY = w.name,

START\_END = string.Concat(se.start,se.end)

};

Console.WriteLine("\n SELECT NAME, START, END");

Console.WriteLine(" FROM WAY S1 JOIN STARY\_END S2 ON S1.IDWAY = S2.ID");

Console.WriteLine(" ORDER BY NAME;\n");

Console.WriteLine(" {NAME, START\_END}(");

foreach (var group in rows)

{

Console.WriteLine(" (" + group.WAY.ToString()

+ ") (" + group.START\_END + "),");

}

Console.WriteLine(");");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Yarcev\_csharp\_lab\_4

{

class Subsystem\_7

{

/// <summary>

/// Вывести номер маршрутки и имена людей, что на нее пожаловались

/// </summary>

public void Procedutre()

{

Yarcev\_csharp\_lab\_4DataContext context = new Yarcev\_csharp\_lab\_4DataContext();

Console.WriteLine(" (join и группировка)");

var rows = from n in context.complaints

join s in context.list\_car on n.id\_list\_car equals s.Id

group n by s.number\_car

;

Console.WriteLine("\n SELECT NUMBER\_CAR OVER(PARTITION BY NUMBER\_CAR), NAME, END");

Console.WriteLine(" FROM CON S1 JOIN STARY\_END S2 ON S1.ID\_LIST\_CAR = S2.ID\n");

Console.WriteLine(" {NUMBER\_CAR, NAME}(");

foreach (var g in rows)

{

Console.WriteLine(" ("+g.Key+":");

foreach (var t in g)

{

Console.WriteLine(" "+t.name+",");

}

Console.WriteLine(" ),");

}

Console.WriteLine(" );");

}

}

}

**Вивод:**

Паттерны довольно сильно ускоряют процесс разработки ПО. Обеспечивают четко прослеживаемую структуру.