#11 LINQ

Задание:

1. Используя класс из лабораторной работы #2 создайте коллекцию **List<T>** из минимум 10 элементов. Реализуйте необходимые интерфейсы, если необходимо.
2. Напишите **LINQ**-запросы по вариантам.
3. Придумайте минимум 3 запроса, комбинирующих различные операторы (минимум 3), например, *first* + *min* + *select*.

Варианты:

Если в задании указано свойство, которым ваш класс не обладает, то его нужно расширить, чтобы класс соответствовал условию. Необходимо реализовать 2 вида каждого запроса: используя язык LINQ и используя методы расширения LINQ.

#1

1. Найти все деревья с возрастом старше 200 лет;
2. Найти все деревья с красной листвой;
3. Отсортировать деревья по убыванию возраста.

#2

1. Вывести все автомобили с объемом двигателя больше 3 литров;
2. Выбрать первые 3 автомобиля с наименьшим расходом топлива;
3. Отсортировать автомобили по возрастанию объема двигателя и вывести их марки.

#3

1. Найти все столы желтого цвета.
2. Вывести 5 первых столов с наименьшей площадью столешницы.
3. Вывести цвет стола с самой большой площадью столешницы.

#4

1. Найти самое высокое здание.
2. Отсортировать здания по возрастанию высоты, выбрать 4 первых здания с конца и вывести их высоту.
3. Найти здания, с высотой больше заданной.

#5

1. Найти ручку с заданным цветом.
2. Отсортировать ручки по толщине стержня.
3. Найти ручку с самым тонким стержнем

#6

1. Вывести все автобусы красного цвета.
2. Найти номер автобуса с самым длинным маршрутом (кол-вом остановок)
3. Вывести номера всех автобусов, который останавливаются на остановке с заданным названием.

#7

1. Найти все товары, объемом больше заданного.
2. Найти три товара с наименьшим объемом.
3. Вывести товары, отсортировав их по возрастанию веса.

#8

1. Вывести все марки телефонов-смартфонов.
2. Отсортировать список телефонов, оставив лишь те, которые являются смартфонами и которые обладают оперативной памятью > 1 GB.
3. Отсортировать телефоны по убыванию общего объема памяти.

#9

1. Найти все книги заданного автора.
2. Найти книги объем > 300 и < 500 страниц.
3. Вывести все книги жанра «фантастики».

#10

1. Отсортировать студентов по возрастанию фамилии и вывести только фамилии на консоль.
2. Найти студентов с заданным именем.
3. Вывести топ-3 студентов с наивысшим средним баллом.

#11

1. Найти всех абитуриентов, которые сдавали математику.
2. Вывести абитуриента с самой большой суммой баллов.
3. Вывести всех абитуриентов, фамилия которых начинается на «Ва».

#12

1. Вывести самый быстрый самолет.
2. Найти 5 самых медленных самолетов.
3. Вывести номера самолетов, которые летают из Минска в любом направлении.

#13

1. Отсортировать заказчиков по имени и вывести первых 3 с самым крупным бюджетом.
2. Найти заказчика, у которого есть 3 активных проекта.
3. Вывести заказчиков, чья фамилия начинается на «В».

#14

1. Найти 3 магазина с самой большой выручкой.
2. Вывести названия магазинов, которые торгуют компьютерными комплектующими.
3. Отсортировать магазины по названию (asc) и выручке (desc) одновременно.

#15

1. Отсортировать университеты по позиции в мировом рейтинге.
2. Найти все университеты, которые предлагают обучение для программиста на платной основе.
3. Выбрать самый престижный университет Беларуси. (на основании международного рейтинга)

#16

1. Найти все аккаунты, с датой рождения не позднее 13.05.1995
2. Вывести аккаунт с самым длинным никнеймом.
3. Отсортировать аккаунты по возрастаю никнейма и вывести имена связанных с ними пользователей.

#17

1. Найти топ-5 самых популярных альбома (по кол-ву проданных копий).
2. Отсортировать альбомы по дате выпуска, начиная с самых свежих.
3. Отсортировать альбомы по исполнителю, по убыванию.

#18

1. Вывести самого популярного исполнителя (по кол-ву проданных альбомов).
2. Вывести 5 самых богатых исполнителей.
3. Найти исполнителя с буквой “a” в имени.

#19

1. Найти все приложения с тематикой финансов.
2. Найти самое популярное приложение.
3. Отсортировать приложения по количеству скачиваний (по убыванию).

#20

1. Найти самую популярную студию (по конечным продуктам).
2. Отсортировать студии по объему годовой выручки.
3. Найти студии, работающие в жанре «инди».

Повышенный уровень:

1. Ознакомьтесь с основными понятиями многопоточности в .NET.
2. Создайте в вашей библиотеке класс **ThreadManager** и наследуйте его от интерфейса **IThreadManager**.
3. Подключите класс с помощью этого интерфейса к **DI** как *синглтон*.
4. Добавьте в интерфейс метод: **StartInNewThread**, принимающий параметром **Action**.
5. Реализуйте метод в классе. Метод должен запускать новый поток, передавая ему в работу делегат-параметр.
6. * Добавьте в интерфейс перегрузку метода **StartInNewThread**, принимающий параметрами:
    a. **Action<T>** - делегат, принимающий параметр;
    b. **T** – объект параметра, который будет передан в делегат.
7. * Реализуйте вызов делегата, передав ему параметр двумя методами:
    a. Используя **ParameterizedThreadStart**
    b. Используя **DynamicInvoke**
8. Оба раза делегат должен выполниться в новом потоке.
9. Продемонстрируйте работу вашего менеджера, подключив его к вашей лабораторной и вызвав 2 перегрузки **StartInNewThread**.

Полезные ссылки:

1. https://docs.microsoft.com/ru-ru/dotnet/api/system.delegate.dynamicinvoke?view=netframework-4.8
2. https://www.pluralsight.com/guides/how-to-write-your-first-multi-threaded-application-with-c
3. https://metanit.com/sharp/tutorial/11.1.php
4. http://www.albahari.com/threading/

Вопросы:

1. Что такое **LINQ**?
2. В чем разница между отложенными и не отложенными операциями **LINQ to Object**?
3. Что такое *лямбда*-выражения?
4. Какова структура типичного запроса **LINQ**? Охарактеризуйте ключевые слова, которые при этом применяются (*from*, *where*, *select*, etc.)
5. Ознакомьтесь с методами расширения **LINQ**: https://docs.microsoft.com/en-us/dotnet/api/system.linq.enumerable?view=netframework-4.8
6. Будьте готовы к вопросу применения любого из них. (обратите внимание, что в списке присутствует множество перегрузок одного и того же метода).

Повышенный уровень:

1. Что такое многопоточность? Что такое параллелизм?
2. Как мы можем создать новый поток в .NET?
3. Можем ли мы передать передать параметр в делегат потока? Как?
4. Для чего используется **DynamicInvoke**?
5. В чем разница между **DynamicInvoke** и Invoke?