**Практическая работа №7 (15 баллов)**

Технологии: C#, .NET 8.0, Task, async/await, PLINQ, Parallel, потокобезопасные коллекции.

Требуется Самостоятельно изучить PLINQ:

https://learn.microsoft.com/ru-ru/dotnet/standard/parallel-programming/introduction-to-plinq https://metanit.com/sharp/tutorial/17.1.php

Реализовать оконное приложение на языке C# с 3 вкладками:

1. На главном окне 3 области (каждая отвечает за свою задачу). Вкладки содержат:
   1. Элементы для отображения результата (label, textbox, таблицы и т.п. на выбор студента)
   2. Кнопку запуска задачи
2. Методы решения задач должны работать асинхронно с основным потоком (окно). После нажатия одной кнопки должна быть возможность нажать кнопку в другой вкладке, пока не закончилась работа в прочих вкладках. При работе методов должна иметь место возможность взаимодействия с окном (например, при переносе окна по экрану оно не должно «зависать»).

Задачи:

1. Подсчет вхождений списка слов в набор текстов.

Дан список текстов и список слов. Необходимо найти кол-во вхождений каждого слова из списка во всех текстах.

* Кол-во слов для поиска взять фиксированным, а кол-во текстов – разным, для того чтобы изучить как влияет кол-во текстов на время обработки. Можно сформировать один набор файлов, а затем каждый файл обрабатывать по несколько раз.
* После обработки одного кол-ва файлов сразу выводить время работы и список частот вхождения каждого слова.
* Реализовать параллельную обработку с помощью модели MapReduce.

1. Обход сайта в глубину.

Реализовать Crawler - приложение, позволяющее обходить все внутренние ссылки по заданному корневому адресу web-ресурса и получать размеченные кортежи некоторых объектов (изображения). Требуется восстановить исходное дерево по которому были собраны ресурсы.

* Реализация предполагает разделение обхода сайта на (не)зависимые подзадачи (Task).
* Реализовать изменяемость глубины обхода.
* В качестве ресурсов взять например, новостные сайты.
* После обработки каждой глубины сразу выводить время обработки данной глубины и информацию о дереве (высота, кол-во узлов, несколько листьев). Для вывода информации можно использовать таблицу.

1. Создать классы:

Сотрудник (ФИО, список заказов)

Заказ (дата заказа, сумма заказа)

Добавить на вкладку формы 3 таблицы.

Добавить кнопку, при нажатии на которую для определенного количества сотрудников будут сгенерированы данные. Для генерации данных можно использовать библиотеки для тестирования, например: **https://habr.com/ru/articles/673674/**

Общее количество заказов берется из расчета 1 сотрудник = минимум 50 заказов (количество заказов для каждого отдельного сотрудника может варьироваться). Например, на 1000 сотрудников должно приходиться минимум 50 000 заказов (1000 \* 50).

После генерации данных вывести первые 100 сотрудников в I таблицу.

После генерации данных вывести первые 500 заказов во II таблицу.

Необходимо выполнить следующие запросы и вывести результаты работы в III таблицу:

1. Получить список всех заказов для сотрудников с некоторыми фильтрами по ФИО (содержит …, начинается с …).

2. Получить список всех заказов по всем сотрудникам до/после указанного числа.

3. Отсортировать сотрудников по средней сумме заказа.

Реализовать каждый из запросов с помощью LINQ (https://metanit.com/sharp/tutorial/15.1.php) и PLINQ (https://metanit.com/sharp/tutorial/17.1.php) и Добавить на форму время выполнения каждого из запросов.

Работу желательно выполнять в группе из 3 студентов.