№ 15 Платформа параллельных вычислений

Задание

https://docs.microsoft.com/ru-ru/dotnet/standard/parallel-programming/task-parallel-library-tpl

1. Используя TPL создайте длительную по времени задачу (на основе ***Task***) на выбор:

   - ✓ поиск простых чисел (желательно взять «решето Эратосфена»),
   - ✓ перемножение матриц,
   - ✓ умножение вектора размера 100000 на число,
   - ✓ создание множества Мандельброта
   - ✓ другой алгоритм.

   1) Выведите идентификатор текущей задачи, проверьте во время выполнения – завершена ли задача и выведите ее статус.
   2) Оцените производительность выполнения используя объект `Stopwatch` на нескольких прогонах.

   Дополнительно:
   Для сравнения реализуйте последовательный алгоритм.

2. Реализуйте второй вариант этой же задачи с токеном отмены ***CancellationToken*** и отмените задачу.
3. Создайте три задачи с возвратом результата и используйте их для выполнения четвертой задачи. Например, расчет по формуле.
4. Создайте задачу продолжения (*continuation* task) в двух вариантах:
   1) С ***ContinueWith*** - планировка на основе завершения множества предшествующих задач
   2) На основе объекта ожидания и методов ***GetAwaiter***(),***GetResult***();
5. Используя Класс ***Parallel*** распараллельте вычисления циклов ***For***(), ***ForEach***(). Например, на выбор: обработку (преобразования) последовательности, генерация нескольких массивов по 1000000 элементов, быстрая сортировка последовательности, обработка текстов (удаление, замена). Оцените производительность по сравнению с обычными циклами
6. Используя ***Parallel.Invoke***() распараллельте выполнение блока операторов.
7. Используя Класс ***BlockingCollection*** реализуйте следующую задачу:

   Есть 5 поставщиков бытовой техники, они завозят уникальные товары на склад (каждый по одному) и 10 покупателей – покупают все подряд, если товара нет - уходят. В вашей задаче: спрос превышает предложение. Изначально склад пустой. У каждого поставщика своя

скорость завоза товара. Каждый раз при изменении состоянии склада выводите наименования товаров на складе.

8. Используя ***async*** и ***await*** организуйте асинхронное выполнение любого метода.

Вопросы

1. Что такое *TPL*? Как и для чего используется тип *Task*
2. Почему эффект от распараллеливания наблюдается на большом количестве элементов?
3. В чем основные достоинства работы с задачами по сравнению с потокми?
4. Приведите три способа создания и/или запуска Task?
5. Как и для чего используют методы *Wait*(), *WaitAll*() и *WaitAny*()?
6. Приведите пример синхронного запуска *Task*?
7. Как создать задачу с возвратом результата?
8. Как обработать исключение, если оно произошло при выполнении *Task*?
9. Что такое *CancellationToken* и как с его помощью отменить выполнение задач?
10. Как организовать задачу продолжения (continuation task) ?
11. Как и для чего используется объект ожидания при создании задач продолжения?
12. Поясните назначение класса *System.Threading.Tasks.Parallel*?
13. Приведите пример задачи с *Parallel.For(int, int, Action<int>)*
14. Приведите пример задачи с *Parallel.ForEach*
15. Приведите пример с *Parallel.Invoke()*
16. Как с использованием *CancellationToken* отменить параллельные операции?
17. Для чего используют *BlockingCollection*<T>, в чем ее особенность?
18. Как используя *async* и *await* организовать асинхронное выполенение метода?