Лабораторная 16. Многопоточность IV

Сдать до 28.12

Задание 1. Ноль-Чет-Нечет (3 балла)

Пусть у вас есть функция printNumber, которую можно вызвать с целочисленным параметром и вывести этот параметр на консоль. Например, при вызове printNumber (7) на консоль выводится 7.

Вам предоставляется экземпляр класса ZeroEvenOdd, который имеет три функции:

```
public class ZeroEvenOdd {
   private int n;

   public ZeroEvenOdd(int n) {
      this.n = n;
   }

   // printNumber(x) outputs "x", where x is an integer.
   public void Zero(Action<int> printNumber) {
   }

   public void Even(Action<int> printNumber) {
   }

   public void Odd(Action<int> printNumber) {
   }
}
```

Один и тот же экземпляр ZeroEvenOdd передается трем разным потокам:

Поток A: вызывает Zero(), который должен выводить только 0.
Поток B: вызывает Even(), который должен выводить только четные числа.
Поток C: вызывает Odd(), который должен выводить только нечетные числа.
Измените данный класс, чтобы вывести серию «010203040506 ...», где длина серии должна быть равна 2n.

Реализуйте класс ZeroEvenOdd:
ZeroEvenOdd(int n) - объект инициализируется числом n, на основе которого определяются числа, которые должны быть напечатаны.
void Zero(printNumber) Вызывает printNumber для вывода одного нуля.
void Even(printNumber) Вызывает printNumber для вывода одного четного числа.
void Odd(printNumber) Вызывает printNumber для вывода одного нечетного числа.

Пример 1

Ввод: n = 2
Выход: «0102»
Объяснение: Асинхронно запускаются три потока. Один из них вызывает Zero(), другой вызывает Even(), а последний вызывает Odd().
«0102» - правильный вывод.

Пример 2

Ввод: n = 5
Выход: «0102030405»

Задание 2. Многопоточный массив (3 балла)

Пусть есть массив целых чисел, с которым одновременно работают четыре типа потоков.

Поток TMin ищет минимальный элемент в массиве, поток TAvg ищет среднее значение, поток TSort сортирует массив по возрастанию, поток TSwap меняет случайным образом два элемента местами. Потоков может быть запущено больше четырех. Реализуйте безопасную и справедливую работу с массивом, чтобы каждый из типов потоков получал равные права на доступ к массиву.

Задание 3. Молекулы H2O (4 балла)

Есть два вида потоков: кислородный (H) и водородный (O). Ваша цель - сгруппировать эти потоки в молекулы воды.

```
public class H2O {
   public H2O() {
 }

   public void Hydrogen(Action releaseHydrogen) {
      // releaseHydrogen() outputs "H". Do not change or remove this line.
      releaseHydrogen();
   }

   public void Oxygen(Action releaseOxygen) {
      // releaseOxygen() outputs "O". Do not change or remove this line.
      releaseOxygen();
   }
}
```

Существует некоторый барьер, на котором каждый поток должен ждать, пока не сформируется полная молекула. Водородным и кислородным потокам даны методы releaseHydrogen и releaseOxygen соответственно.
Потоки должны проходить через барьер группами по три, после чего считается, что они образуют молекулу воды. Вы должны гарантировать, что все потоки для одной молекулы связываются раньше, чем любые другие потоки следующей молекулы.
Другими словами:
Если кислородный поток достигает барьера при отсутствии водородных потоков, то он должен ждать двух других водородных потока.
Если водородный поток достигает барьера, когда нет других потоков, то он должен ждать кислородный поток и другой водородный поток.
Нам не нужно беспокоиться о явном сопоставлении потоков. Потоки не обязательно знают, с какими другими потоками они связаны. Потоки проходят барьер в полном комплекте. Таким образом, если мы рассматриваем последовательность потоков и разделяем их на группы по три, то каждая группа должна содержать один кислородный и два водородных потока.
Напишите код синхронизации для молекул кислорода и водорода, который обеспечивает соблюдение этих ограничений.
Для демонстрации работы программы вам необходим генератор H и O потоков.

Пример 1

Ввод: "HOH"
Выход: «HHO»
Пояснение: «HOH» и «OHH» также являются допустимыми ответами.

Пример 2

Ввод: "OOHHHH"

Выход: "HHOHHO"
Пояснение: «HOHHHO», «OHHHHO», «HHOHOH», «HOHHOH», «OHHHOH», «HHOOHH», «HOHOHH» и «OHHOHH» также являются допустимыми ответами.

Задание 4. Умножение матриц (3 балла)

Разработайте многопоточное приложение, выполняющее вычисление произведения матриц A (m×n) и B (n×k). Элементы c[i,j] матрицы произведения C = A×B вычисляются параллельно p однотипными потоками. Если некоторый поток уже вычисляет элемент c[i,j] матрицы C, то следующий приступающий к вычислению поток выбирает для расчета элемент c[i,j+1], если j<k, и c[i+1,k], если j=k. Выполнив вычисление элемента матрицы-произведения, поток проверяет, нет ли элемента, который еще не рассчитывается. Если такой элемент есть, то приступает к его расчету. В противном случае отправляет сообщение о завершении своей работы и приостанавливает своё выполнение. Главный поток, получив сообщения о завершении вычислений от всех потоков, выводит результат на экран. В каждом потоке должна быть искусственная задержка на выполнение вычислений (чтобы дать возможность поработать всем потокам).

Задание 5. MyBarrier (4 балла)

Реализуйте с помощью стандартных примитивов аналог Barrier.

Пример сигнатуры класса:

```
public class CMyBarrier : IDisposable
{
    public CMyBarrier(int participantCount);

    public bool SignalAndWait(TimeSpan timeout);
    public void Dispose();
}
```

При создании указывается participantCount. Метод SignalAndWait(TimeSpan timeout) декрементирует счетчик на 1 и ждет, пока все другие потоки вызовут SignalAndWait. При вызове SignalAndWait после достижения 0, выкидывается исключение. Если Timeout истекает до достижения 0, то возвращается false, иначе true.

Класс должен работать в многопоточном приложении. Выделяемые ресурсы должны освободиться в методе Dispose. Реализовать обработку различных передаваемых параметров. Разрешается использовать только AutoResetEvent, ManualResetEvent, Mutex, Semaphore, lock().

Реализуйте проверку работы создаваемого класса в различных ситуациях.