Mandatory Activity. Concurrent Programming. Lab 09

This activity must be autonomously done by the student. **It must be done prior to the following laboratory class**. It will be used as part of the following laboratory.

Activity

The project provided in `lab09.mandatory.activities.students.sample.code.zip` contains Bitcoin value information extracted from a real source. Its code obtains an array of `BitcoinValueData` objects with the value of the bitcoin cryptocurrency at a concrete date and time. With this information and starting from the `vector.modulus` project shown during the laboratory, the student must:

- Implement an application that concurrently computes the number of occurrences of a bitcoin value higher than a fixed value (passed as an extra parameter in the master) in the provided array. The master must return the number of times the bitcoin value has been higher or equal than the provided one. For example, the Bitcoin value has been over 7000 euros 2826 times.
- Test the sequential (one thread) and concurrent (multiple threads) implementation using the Visual Studio testing tool.
- Measure context switch. Analyze when the highest benefit is obtained. Identify the number of threads from which the concurrent version is slower than the sequential one.
- Invoke `GC.Collect()` and `GC.WaitForFullGCComplete()` after the execution of the algorithm.
- Execute it twice to avoid the effects of JIT compilation. Run it in `Release` mode (not in Debug mode).
- In an *Excel* document, show the evolution of execution times relative to the number of threads. Identify the values mentioned in the previous point.

Use the appropriate programming language features learned so far.