Thread-Safe Initialization of a Singleton

The singleton is a typical candidate for thread-safe initialization.

  - You initialize it once but read it very often.
  - You only want to apply expensive synchronization when you initialize it.

  ➡ Using an expensive locking for all read and write operations is pessimization.

- The double-checked locking pattern should solve this performance issue.

- Analyze the different strategies for the thread-safe initialization of a singleton. For your analysis, perform the following steps:
  1. Compile each program with maximum optimization
  2. Execute each program locally
  3. Note the execution time of each program

Thread-Safe Initialization of a Singleton

- Thread-Safe Initialization
  - `singletonSingleThreaded.cpp`
  - `singletonMeyers.cpp`
  - `singletonLock.cpp`
  - `singletonCallOnce.cpp`
  - `singletonSequentialConsistency.cpp`
  - `singletonAcquireRelease.cpp`