Atomic Types Solutions

Atomics

- Write a task function that increments a global variable 100,000 times in a tight loop
- Write a program that runs this task in concurrent threads and prints out the final value of the variable. Increase the number of threads until you see signs of data corruption
- Make the variable atomic. Do you still get data corruption?
- Make the variable volatile. Do you still get data corruption?
- Briefly explain your results

Atomic Types and Operations Solutions

- Non-atomic version
  - There is a data race. Threads interfere with each other and use incorrect values when incrementing the counter
- Atomic version
  - Incrementing the counter is done as an uninterruptible operation and the new value of the counter is immediately visible to all threads. This ensures the correct result
- Volatile version
  - The volatile keyword has no significance in multi-threaded C++ programs. This is equivalent to the non-atomic version