Thread-local Variables Solutions

Thread-local variables

- What happens if the declaration of mt is moved into func?
  - Same functionality (both threads print the same numbers) and performance
- What happens if mt is declared as static instead of thread_local?
  - There is a single engine instance which is shared by all the threads
  - Each thread receives the next numbers from the shared engine's sequence
  - The two threads print out different numbers
- What happens if mt is declared as a normal local variable in func?
  - Each thread has its own engine instance
  - Each thread receives the next numbers from its own engine's sequence
  - Both threads print the same numbers

Performance implications

- What happens if the declaration of mt is moved into func?
  - Should produce the same code - no effect on performance
- What happens if mt is declared as static instead of thread_local?
  - Only one engine instance is created
  - This has less overhead than creating one instance per thread
- What happens if mt is declared as a normal local variable in func?
  - An engine instance is created in each thread (same as thread_local)
  - However, if the engine variable is in a function called repeatedly by the thread, instead of the worker function, there would be one instance per function call
  - In the thread_local version, there would still only be one instance per thread