1. Difference between classic and modern
   1. toring a static instance and checking its existence upon request to ensure only one instance is created.
   2. instantiates the Singleton in the static function that returns the instance (lazy initialization)
2. Static member variables in singleton
   1. Static member variables can mimic a Singleton by allowing shared state and methods that can be accessed globally without creating class instances. However, unlike Singletons, they don't maintain an independent lifecycle or allow for lazy initialization
3. Correct file at compile time
   1. It uses the directives in the instance()
4. Global
   1. You have to potentially hunt through the entire code base to find what interacts with the global data
5. Encouraging coupling
   1. They're universally accessible across different files and all parts of an application, creating dependencies that are often hard to track
6. Bugs that it can create:
   1. Concurrency issues
   2. Name errors: mistakenly creating conflicts with variables that have the same name
   3. Uncontrolled access and modification to parts of program
   4. Hidden dependencies: used in a variety of places in the program, can unexpectedly break, causing chaos.
7. Poorly designed singletons are often masking that only one of the singleton pattern's problems needs to be solved, not both ensuring a single instance and global access.