Two versions of the AsyncLog have been created. The first one is the one provided by you but refactored. The second one is using a thread-safe data structure for the Loglines as well as a different, simpler infinite logging loop approach. Both of the versions implement the ILog interface and use the FileHandler class for I/O operations.

The main take from this revision of the code is the decoupling of the folder/file management/writing and the async logger itself. This separation of responsibilities aims to improve maintainability, reusability and also make the code more readable and easier to comprehend. A class called “FileHandler” has been created, which manages the setup of the LogDirectory as well as that of the log files themselves. This includes headers of the files, formatting, naming, as well as writing in the files. Furthermore, error handling has been introduced to prevent the program from crashing, especially on IO/locked files. Also, the FileHandler allows for the LogDirectory to be changed which automatically handles the allocation of log files to the new directory.

Several bugs were fixed such as the modification of the “_lines” collection during the execution of the foreach. A simple ToList() invocation on the collection creates a non-referenceable copy which cannot be modified during the execution of the loop and the exception is avoided.

Another notable bug is failing to create a new log when midnight passes. This was due to the incorrect way of checking the difference between the current time and the startup time of the logger.

Additional changes include reduced nesting of if statements, removing redundant code, using string interpolation, closing StreamWriters to remove file locks.

Aside from the requested logs, two additional ones have been created to show the bug related to determining whether midnight has passed. In one of them, the results from using the two different approaches have been juxtaposed for the interested reader.

NUnit has been added as the test framework of choice. Test cases have been created for the major demands. For more details, check the code file.

I would be happy to answer your questions related to the code during the interview.