Threads + asynchronous programming

# Stack

1. Create a class that represents the stack (it will have methods for adding and removing values and for detecting whether the stack is empty) - use the List <T> collection to implement it.
2. Run 5 threads that will randomly add and remove data from the stack, in a loop (to infinity).
3. Verify that the stack is working correctly - if so, you have a bug somewhere :D
4. Modify the stack so that it is "thread safe" - so that there are no errors when working with the stack within multiple threads at the same time (it is necessary to add locking).

# Synchronization

1. Start one thread, which will use the already prepared stack - add one randomly generated value into it, every 100ms.
2. Run another 5 threads, where each thread will do the following indefinitely:
   1. If the stack is empty, it will wait for a value to appear in it.
   2. If the stack is not empty, get and then remove the value from the stack.
   3. Print the value along with the ID of the current thread to the console.
   4. Sleep the thread for a random time between 40 and 1000ms.
3. Try to implement Monitor class for waiting for the data.

# Asynchronous programming

1. Create a new console application and modify Main to be asynchronous.
   1. Create a new asynchronous method called "Experiment". Then implement this logic:
   2. Print “Start“, to the console.
   3. Pause task for 1000ms.
   4. Write some data into the file using class **StreamWriter** – asynchronously.
   5. Pause task for 1000ms.
   6. Print ”End”, to the console.
2. Call method Experiment from the Main method.
3. In the class Experiment, use class Stopwatch to measure time required for writing data to the file. Then return this time as a return value of this function.