1. Create a class library (NetworkTools) with the followings:
	1. PingStats class: string address + int packets + int totalBytes + int totalMs. Equals() + GetHashCode() is needed too.
	2. HostPinger class: the constructor expects an address, and the class should contain a *PingStats TryOnce(int numTries)* method, that calls the „*ping <CÍM> -n <DARABSZÁM>*” command in a new process, and it analyses the output to create the returned *PingStats* instance.
	3. AsyncHostPinger class: the constructor expects an address, and it launches a new Task that will (once in every 5 seconds) start two pings towards the given host, and put the result in a static (shared) list. The list can only contain maximum 100 elements. The class should have a *void Cancel()* method that stops the Task, and a *public static* *List<PingStats> GetListCopy()* method that will return with a shallow copy of the list. Be careful with thread synchronization (multiple AsyncHostPinger will use the same list, and the List<T> is not thread safe).
	4. MultiHostPinger class: the constructor creates an empty *List<AsyncHostPinger>* instance. The class should have a *void AddHost(string address)* method that will add a new *AsyncHostPinger* instance to the list; and also a *void Cancel()* method that will stop all the *AsyncHostPinger* instances.
	5. ListProcessor class: it should have a
	*List<PingStats> GetSummarizedStats(List<PingStats> originalList)* method, that summarizes the input list according to addresses (e.g. if the list contains six entries for the hup.hu address, then the output should contain only one PingStats instance, with the sum packet count/bytes/milliseconds of the six entries in it).
2. Create a class Library (NetworkToolsTests), with a ListProcessorTester class. This class will test the GetSummarizedStats method against the following rules:
	1. Null parameter is not allowed.
	2. For a list input with zero elements, a list with zero elements is the output. For a list input with one element, a list with one element is the output.
	3. If the input list contains the same host multiple times, a list with one element is the output, and that one element should have the host address that is the original list input.
	4. If the input list contains elements with all different host addresses, then the same elements should be in the reply.
	5. Check if the method works well using at least three different test cases coming from a TestCaseSource.
3. Create a console application
	1. First it should create a ListProcessor instance and a MultiHostPinger instance, and add the addresses hup.hu, bbc.co.uk, amazon.com.
	2. In a while loop that is waiting for a keypress (Console.KeyAvailable), in every 5 seconds write out the summarized stats to the console
	*listProcessorInstance.GetSummarizedStats(AsyncHostPinger.GetListCopy())*
	3. When the loop is broken, the MultiHostPinger should be stopped by calling Cancel().