Concurrency Exercise 1

You are supplied with the code for a class `DirectoryLister`. A user can interactively input a file path to `DirectoryLister` and, if that path is a directory, the program will create a recursive listing of its filesystem subtree and store them in a `Map`. The user can interactively query the map to discover which listings have been created in this run of the program, and can query the contents of a particular listing.

To see the problem with the program, experiment with a small and then a large directory tree. Because the program creates the listings sequentially, input is blocked while large listings are being created.

To fix this problem, refactor the code marked "TODO" into a separate method (this just makes testing more convenient, because it gives you the choice of submitting directories non-interactively), and turn each listing request into a task to submit to a thread pool, so that the program can execute more than one listing request simultaneously. You will want to store the resulting `Future`s in a `Map` like the one in the existing code (only thread-safe). Try to make sure that the thread pool shuts down in an orderly way.

Experiment with providing the user with some of the following functions:

- Ability to see all listing jobs that have begun, and whether they have completed
- Ability to query completed listing jobs, as at present
- Ability to cancel long-running listing jobs
- Ability to shut down the thread pool immediately (this one is tricky)
- Ability to return each listing as it is completed (using `ExecutionCompletionService`)