We need you to efficiently merge a directory of sorted text files to a single, sorted output file.

Each file in the directory contains lines of text. Other than occasional empty lines, the lines within each text file are in lexicographic order. Your job is to write a program that takes in the name of the directory and an output file, and produces an output text file that:

- Contains no empty lines
- Contains, in lexicographic order, each unique non-empty line contained by any file in the directory
- Contains no duplicate lines

Your solution should also *verify* that the input criteria are true. If any of the input files' lines are *not* sorted in lexicographic order, excluding empty lines, your solution should throw an appropriate error.

For example, for the command:

```
./merge input_dir output_file
```

Where inputdir contains the following files and contents:

```
a.txt:
      familiarity breeds contempt

      slow and steady wins the race
      the pen is mightier

      the truth will set you free

b.txt:
      no peace for the wicked
      no peace for the wicked
      to err is human

c.txt:
      no peace for the wicked


      the truth will set you free
```

The output file created should contain:

```
      familiarity breeds contempt
```

```
no peace for the wicked
slow and steady wins the race
the pen is mightier
the truth will set you free
to err is human
```

Please think carefully about the performance characteristics of your solution. **Think about both computational complexity and space complexity.** Try to minimize both. Your solution should work efficiently even if the directory contains a very large number of files. It should work even if the files in the directory are very large. **The number one way candidates fail this exercise is by building a RAM-hungry solution that does not work properly with large inputs.** So, again: please minimize both computational and space complexity of your solution.

You may use any tools you like, but please produce a solution we will be able to easily run on a single machine in a typical UNIX environment. Please send:

- All source code and any configs or other files necessary to build and run your program.
- Instructions for building and running your program.
- A quick analysis of the computational complexity and space complexity of your program.

Candidates typically spend somewhere between 1 and 3 hours on this exercise, but it isn't strictly timed and we don't enforce a hard deadline. We note how long it takes to get a solution back, and if it's 8 hours we may have some questions, but in general we prefer quality over speed.

Thanks!