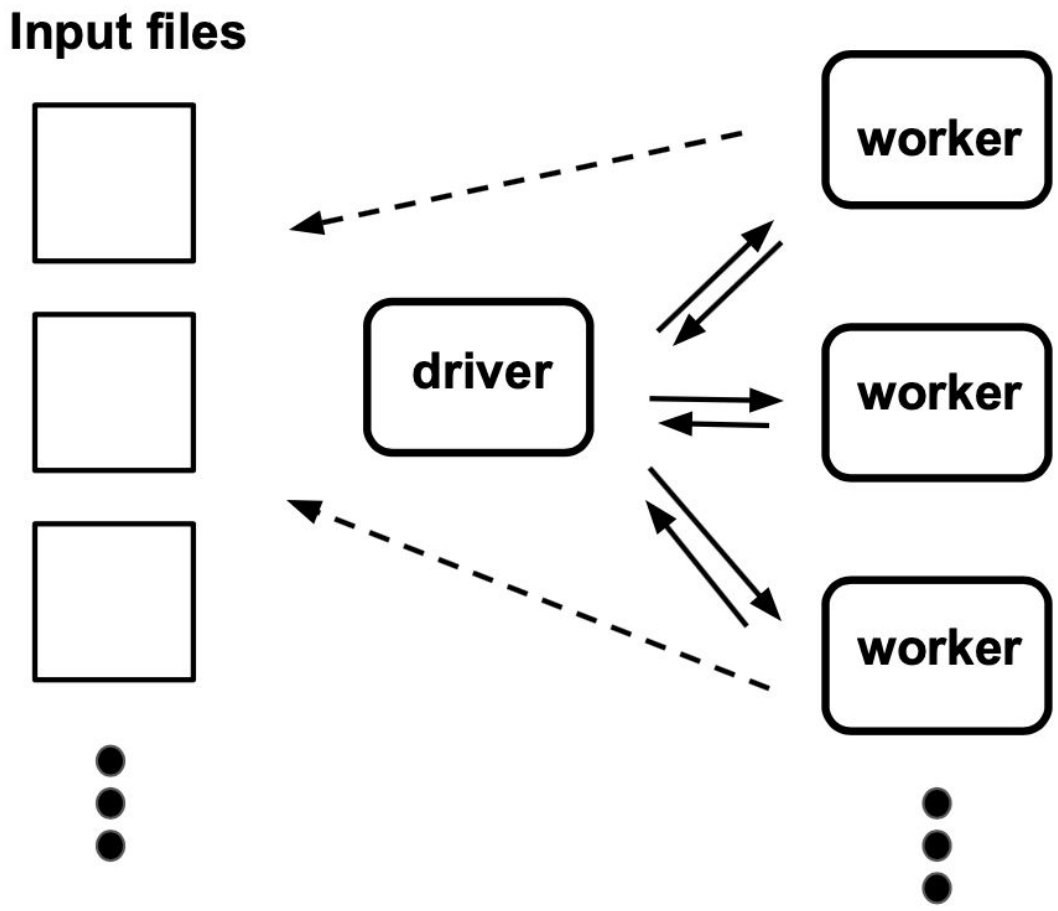
Create a distributed map reduce program to do wordcount with gRPC (https://grpc.io) on your local machine, when you finish, you can start driver program, and then open a few more terminal windows, and start more worker program, when it finish, you should see output files containing the frequency of each work in the input files

Please use gRPC and only standard library from the language of your choice, input files are in the inputs folder
Please include a README briefly explaining your code when you're done, also, some tests would be nice

on a high level, your code should function as follows:



Hints: (consider this as a thought process of this problem):

- Driver code and worker code are separated and should communicate through gRPC, the two are started separately.
- Worker should wait for driver to start, and ask driver for a task, each worker will process one task at a time
- Driver need to decide what task to give to a worker
- There are two types of task, map task and reduce task, each task should have a task id, and we should be about to config the number of map & reduce tasks, as N and M, map task id can be from 0 to N - 1, and so does reduce task id.
- For simplicity, you can directly put map and reduce function in worker, and call either one of them depends on the task given, and don't spend too much time to generalize it to other map

reduce task, only focus on wordcount is fine, the focus of this assignment is not about mapreduce, but distributed

- gRPC should only transfer metadata about the task (do not transfer actually text data through RPC), worker can directly access file system
- Map task take the input files, and separate text into single words, and put each word into a “bucket” (an intermediate file), bucket should be decided with (first letter of the word) % M, when it finish, it should produce intermediate file, e.g. mr-<map task id>-<bucket id>, in these files, one word per line. Do not aggregate in the map task
- Reduce task take all the mr-<map task id>-<bucket id> files with (bucket id == reduce task id), and count the frequency of each word, output the final file, e.g. out-<reduce task id>, output should have the format of followings:

  Hello 232
  World 653
  ...
- Driver should be able to decide if all the tasks are finished, if so, it will terminate, and worker should also be able to know when driver exit, and worker will exit as well

As for final result, please include a result directory structure as follows (N = 6, M = 4):

```
(base) ➜  mr_dist tree files
files
├── intermediate
│   ├── mr-0-0
│   ├── mr-0-1
│   ├── mr-0-2
│   ├── mr-0-3
│   ├── mr-1-0
│   ├── mr-1-1
│   ├── mr-1-2
│   ├── mr-1-3
│   ├── mr-2-0
│   ├── mr-2-1
│   ├── mr-2-2
│   ├── mr-2-3
│   ├── mr-3-0
│   ├── mr-3-1
│   ├── mr-3-2
│   ├── mr-3-3
│   ├── mr-4-0
│   ├── mr-4-1
│   ├── mr-4-2
│   ├── mr-4-3
│   ├── mr-5-0
│   ├── mr-5-1
│   ├── mr-5-2
│   └── mr-5-3
└── out
    ├── out-0
    ├── out-1
    ├── out-2
    └── out-3
```