Liars Lie

Liars Lie is a game where a client queries a set of agents about an integer value. A configurable subset of agents tell the truth and will reveal the true value when asked. On the other hand, liars lie in Liars Lie and each liar will respond with an arbitrary (random) value, but always the same arbitrary value, when asked. The challenge is to determine the "true" value `v` by asking the agents their individual values. We call this `v` the *network value* below.

Your task is to implement our favorite game such that we can play it with independent agents, being queried by a client. The agents and the client should communicate according to a protocol of your design with an arbitrary number of messages and types but the following constraints:

- Liars must lie convincingly, liars cannot reveal that they are liars, they must be found out.
- Each liar must lie consistently and respond to the same message request with the same response throughout the game.

The project should compile an executable `liarslie`, which will be used to play the game from the command line. When started, it should accept in a loop the commands from the standard input, and print the results to the standard output.

We specify two modes of play, "standard mode" and "expert mode". You should implement **both** modes.

Standard mode

`start --value v --max-value max --num-agents number --liar-ratio ratio`

Should launch a number of independent agents, with `number * (1-ratio)` honest agents always responding with the specified integer value `v`, and (`number * ratio`) liar agents responding x with `x != v` and `1 <= x <= max`. This command will start the game and, when ready, produce the agents.config file, which will contain sufficient information to identify and communicate with all agents reliably, and print "ready" on the terminal. We expect the agents to be instantiated in one of two possible forms:

1. Each agent is in a separate thread or process.
2. Agents communicate over the network, for example using TCP.

Additional requirements:

- Each agent should only know its own identity and value; it may learn values or identities of other agents only by following the protocol of your design. In particular, only the client is allowed to access the `agents.config` file.
- The `agents.config` file should contain only the agent identities, but not their values; it should not be possible to determine the network value from this file.

`play`

May be invoked multiple times. Upon each invocation, the client should read the agents.config file (which is to be treated as coming from an external service), connect to the agents using the protocol of your design, play a round of the game, and print the network value v.

Hint: agents **DO NOT** need to communicate to other agents for this mode of game play to be successful.

`stop`

This command will stop all agents listed in the file agents.config, remove the information about stopped agents from the file, and exit from the executable.

Expert mode

We want Liars Lie to continue to work in a situation where the client can only communicate directly with a subset of agents. We also want to be able to reconfigure the network on the fly. Here are the new commands to implement for expert mode:

`extend --value v --max-value max --num-agents number --liar-ratio ratio`

This command should check for the existence of agents.config, and, if present, *extend* the network by launching the specified agents and appending information about them into agents.config.

`playexpert --num-agents number --liar-ratio ratio`

In liars lie expert mode we can determine the network value by directly querying only a small number of agents. Following the same assumptions as for play command, this command should determine the network value by directly querying at most the given number of agents. The liar ratio gives the user assumption on the ratio of liars among agents. Remember, liars cannot modify their own value, but could tamper with other types of messages. Make sure the client can deal with this.

Hint: Agents **DO** need to communicate to other agents for this mode of game play to be successful.

`kill --id id`

We model network failures by allowing some agents to be killed. This command should kill the agent with id, but should leave `agents.config` untouched.