COMP 4958: Lab 4

Submit a zip file named `lab4.zip` containing the file `arithmetic.ex` and the folder named `card` for part 1. (Note: Do not submit the `_build` directory that is under `card`.) Your programs must compile without errors or warnings. Maximum score: 14

1. Implement a supervised card worker together with a card store. Use the `--sup` flag when creating the `card` project with `mix`.

   The card worker (in module `Card.Worker`) is a registered server and its interface is basically the same as the card server specified in lab 2 (i.e., no pid parameter to `new`, `shuffle`, `count` and `deal`), except that `start` is replaced by `start_link`.

   The card store (in module `Card.Store`) provides functions `get` and `put` to read and write the state of a card worker to a file. Name the file `cards.db`. Note that when the card worker is restarted by its supervisor, it should retain its previous state.

   To facilitate the testing of re-starts, the card worker

   - must print a message whenever it is being started or re-started;
   - must "crash" when `deal` is called with an argument that is not an integer. (Make sure that there is no handler for `deal` when it is called with a non-integer argument.)

2. The `Arithmetic.Server` from part 2 of the previous lab creates a pool of workers to handle requests. We want to improve `Arithmetic.Server` so that when any of its workers dies unexpectedly, the server creates a new worker to replace it. Effectively, `Arithmetic.Server` becomes the supervisor of its workers.

   Make changes so that when a worker dies, the server is "notified" and starts a replacement worker process. To facilitate testing,

   - do not handle the error that occurs when a worker is asked to calculate the square or square root of a non-number, so that we can crash the worker by asking for the square of, for example, `:hello`;
   - print the PID of the replacement process when it is created.

   Note that the PIDs of all the workers are printed when the server starts up, as specified in the previous lab.

   Put your code in a file named `arithmetic.ex`. Use only what we have talked about in class (exit signals, links, etc.) to implememt this. (Do not use monitors or polling via `Process.alive?`)