Pre-Interview Questions

Instructions for Completion

Please submit answers as a github project with instructions in a readme on how to build and run. Preference for language choice is one of Node.js, Java or Scala and can assume the appropriate standard runtime and build tools are available for each of those languages.

Q1. Employee Associates

An employee of a company works with multiple people in the organisation. Each employee has a single manager but works on multiple projects in the organisation.

Q2.1 Create a data model to represent this structure.

Example:
John works for Lisa and works on Project A and Project B
Jack also works for Lisa and is the works on Project A
James works for Leonard and works on Project A and Project C
Lucy works for Leonard and works on Project B and Project C
Sam works for Leonard and works on Project B
Lisa and Leonard work for Simon

Q2.2 Create a function that for a given organisation and employee will return all their project team mates (employees that work on a common project).

Example: James is teammates with John, Jack and Lucy

Q2.3 Create a function that for a given organisation and two employees will return their first common manager.

Example: John and Jack's common manager is Lisa
John and Sam's common manager is Simon

Q2.4 Create a function that for a given organisation and project will provide a guess on the closest people manager responsible for the project based on the number of direct and indirect reports working on it.

Example: Project A is likely the responsibility of Lisa as the majority of team members work for Lisa

Q2. URL Pinger

Assume there is a helper object ‘db’ already defined in your environment with the following functions:

```
 /*
 * Saves an arbitrary object in a NoSQL database
 */
db.save(obj: Object)

 /*
 * Find all objects whose attibute @attr matches the regular
 * expression @what
 */
db.find(attr: String, what: RegExp)
```

Q2.1 - Define an object with the minimal set of properties to save to the DB to represent the response to a web page request. Eg: if we had to define the minimal properties in a carObject they could look like this:

```
 carObject:
      brand: String,
      color: String,
      engineSize: Int,
      drive: '2wd' | '4wd' | 'awd'
```

Q2.2 - Write a script/class that given 4 urls, it will send a request to each one of them and store the result in the database.

Q2.3 - Now, if a given url is in the test.com domain, also print the results to stdout.

Eg:

- http://site1.com/path - save to db only
- http://site2.com/path - save to db only
- http://test.com/path - save to db and print

Q2.4 - Write a function that will find all urls in the database that tried to set a cookie when requested.

Eg: back to the carsObject example, if we had to find all cars that are not “awd” we could do:

```
function findAllNonAWDCars = {
      return db.find('drive',/^(?!awd).*$/);
}
```