Task 1:
-simplify syntax of FabricCAClient.js file and put in ES6+ style
-refactor without changing any functionality
->= 2 types of coding style improvements
(+) Create patch on fabric-sdk-node repository on Gerrit (software), or answer as a report

https://www.makeuseof.com/tag/es6-javascript-programmers-need-know/
ES6+ style:
-ECMA script is the standardized name for Javascript
-ES6 refers to version 6 of the ECMA script programming language
-was subsequently renamed to ECMAScript 2015
-after ES6 in 2015, there was a new ESX released each year - up to ES9 now, but nothing major
-still not fully supported on all browsers, but you can use a transpiler to convert to ES5, which is
Added Functionalities:
-Constants
-Block-Scoped Variables and Functions (let)
-Arrow Functions (var a = x => x+1)
-Default Function Parameters (var a = (a, b=1) => a + b)
-Rest Function Parameters (var a = function(x, y, z, … args) , ex. then can use args.length in function body)
-String Templating (can make templates which are functions that resolve to strings within ` `
-Object Properties (instead of old {x:x, y:y}, you can declare the variables and say var a = {x, y}, and manipulation and creations of object properties much easier. Can also define object functions within the object declaration!)
-Formal Class Definition Syntax (syntactic sugar)
-Destructuring - binding using pattern matching, with support for matching arrays and objects, fail soft, producing undefined values when not found
-Iterators
-For .. of
-Generators
-Unicode/regex support
-Modules
-Map, Set, WeakMap, WeakSet
-Proxies
-Symbols
-Able to subclass built-in types
-more API's for Math, number, string, array objects
-binary and octal literals
-Promises - for asynchronous programming
-Reflect API - like the inverse of the Proxy API
-Tail calls - calls in tail-position guaranteed to not grow the stack unboundedly

https://medium.com/@madasamy/javascript-brief-history-and-ecmascript-es6-es7-es8-features-673973394df4
ES7 Additional Features
-Array.prototype.includes()
-exponentiation operator **

ES8 Additional Features
Major features:
-Async Functions -
-Shared Memory and Atomics -
Minor features:
-Object.values/Object.entries
-String Padding
-Object.getOwnPropertyDescriptors()
-Trailing commas in function parameter lists and calls

ES9 Additional Features
Major features:
-asynchronous iteration
-rest/spread properties
Regular expression features:
-regexp named capture groups
-regexp unicode property escapes
-regexp lookbehind assertions
-s (dotAll) flag for regular expressions
Other features:
-Promise.prototype.finally()
-Template literal revision

.js file (556 lines total)
-line 32 - old style of class definition
-line 121 - has a promise declaration
-”const self = this” seen multiple times, i.e. line 165, 201, 412 - but are they in same class def? - yes! - in a method, “this” refers to the owner object, and in a function, “this” refers to the global object.
-line 308 const empty array looks weird
-line 419 the old way of doing an object declaration
-line 505 empty {} const looks weird

(1) Changed all const self = this and moved it to one spot on line 26.
(2) Updated object property declarations on two instances of constant “requestOptions”

~~~~~~
~

Task 2:
Run fabric-sdk-node full test on my development machine - look at readme file to know how to do so
Determine which tests are end-to-end tests, if it does not fail. If it does fail, write about why you think it did fail

https://medium.com/@giltayar/testing-your-frontend-code-part-iii-e2e-testing-e9261b56475
-(*) There are not many E2E tests
-It is basically automating how a user might interact with the actual software
-They are slow tests
-they are flaky tests, meaning they may fail
-should only be between 1 to 10 of these

https://github.com/hyperledger/fabric-sdk-node/blob/release-1.4/CONTRIBUTING.md
-Functional tests are held at the repository level under teh ~/test/integration directory and target a Fabric network that has been created based upon information within the ~test/fixtures directory. The functional tests are currently written in Tape, with the intention of teting the Fabric-SDK-Node packages from a user perspective against a live Hyperledger Fabric Network

(*) Gulp was not installed on the vm that I was logged into, but I believe the same functionality can be achieved by simply running make unit-test. However, this did crash midway through after getting an error on a test called “chain 3-node Raft Cluser”

Unit test logs I collected may be seen in file Q2.

Out of 101 tests, only one test showed an error, and this was called “TestEtcdraft”. A summary of the error as displayed was as follows: [Fail] Chain 3-node Raft cluster when reconfiguring raft cluster reconfiguration [It] does not deadlock if leader steps down while config block is in-flight. I originally thought the time that it took to run this test might have suggested it was E2E, but it seems from the logs that it actually times-out after 10 seconds.

~~~~~~
~

Task 3:

How to build a command-line nodejs script:
https://www.sitepoint.com/javascript-command-line-interface-cli-node-js/

Answer contained here:
https://github.com/CheyenneAtapour/hyperledger/blob/master/q3.js

~~~~~~
~

Task 4:

Reasons to go open source:
-Transparency
-Having customers understand what is happening behind the scenes
-building trust with the community, which includes C and B customers/clients (businesses and regular customers)
-Allows other people to contribute and feel helpful
-helps draw upon the skills of the public, which are diverse
-get all kinds of troubleshooting from different people, with different development environments from all over the world
-promotes a culture where other customers help other customers from all over the world, in a mutual altruistic fashion

Why diversity and collaboration is required for the success of an open source project:
-Diversity is best for highly productive teams, according to a google study
-You do not get all the benefits of going open source if you do not allow open collaboration from everyone, which means diversity
    -Futhermore:
    -you do not appear transparent because you are not giving others the chance to contribute
    -you are not opening up the projects to the full expertise of the public
    -You are disincluding some groups of people, which can hurt your reputation and make your company seem biased towards certain groups of people
    -You do not give everyone a fair chance to feel they are helpful
    -different perspectives can offer different values that some may not consider, especially if they are not from a tech background, such as issues of ethics, and fairness - i.e. machine learning algorithms which are not trained on proper data, such as the one which classified african americans as monkeys

Why open source projects managed by non-profit foundations are popular on earth
-non-profit and open source speaks a lot about the company
-this promotes a culture of fairness and genuinely wanting to help mankind with software solutions, and allowing people to benefit from it and contribute from it
-People like free software, which is something they can achieve with open source from non-profits
~~~~~~

-People also want to support non-profits and feel they are helping the world, so they want to work for such companies, which also helps those companies grow, as well as the people grow. Thus, it is a sustainable model of business since there is altruism and reciprocation

My knowledge of the hyperledger community:
I first heard about hyperledger at the largest University-led blockchain and fintech conference led in London at Imperial College.
I understand that hyperledger was named in Forbes top 50 blockchain companies, and many enterprise software applications utilize hyperledger technologies