CSE 331: Software Design & Implementation

Section 1

In this section, we will debug a very simple web application and log our debugging process.

Start by performing the setup steps described on page 2. Then, complete the following parts.

1. Fix Fibonacci

In this problem, we will fix the function `fib` in `src/routes.js` so that it correctly returns the n-th Fibonacci number, where n is a parameter supplied by the user as a query parameter.

(a) Run `npm run start` and navigate to `http://localhost:8080` and click the button `Calculate Fibonacci`. You will see that the message returns `undefined`.

Navigate to `https://comfy.cs.washington.edu/service/hw1-practice` to access the Debugging Log and add a new entry. In the new log entry, document this error and write down the start time of the debugging session. Then look through the console logs and see if you can find the bug. If you can, fix it.

(b) Navigate to `http://localhost:8080` and input a number in the textbot and click the button `Calculate Fibonacci`. Verify that the correct Fibonacci number is printed.

(c) Once you have fixed the bug, fill out the rest of log entry with your findings. Be sure to answer:

- What experiments did you run to fix the error?
- What was the error?
- What was the defect?

Once you have added a log entry, be sure to click 'Save Log' to save your log entry. Note: For the practice website, the log will not actually save since it is just practice.

2. Next and Previous Fibonacci

In this problem, we will implement the functions `nextFib` and `prevFib` in `src/routes.ts` which will calculate the next and previous Fibonacci numbers given the current Fibonacci number.

(a) Notice that you ARE NOT given the current Fibonacci number as a request query parameter to the functions `nextFib` and `prevFib`. Instead, the current Fibonacci number is stored in the state of the application in the variable `currentFib`. Use this variable to calculate the next and previous Fibonacci numbers and implement the functions `nextFib` and `prevFib` in `src/routes.ts`.

(b) Navigate to `http://localhost:8080` and click the buttons `Next Fibonacci` and `Previous Fibonacci`. Verify that the correct Fibonacci numbers are printed.

(c) Try pressing the `Previous Fibonacci` button until it becomes a negative number. If we are calculating a negative Fibonacci number, we should return `undefined`. Does your implementation handle this case correctly? If not, fix it.

(d) In the textbox, input the number 10 and click the button `Calculate Fibonacci`. Then, click the button `Next Fibonacci`. Verify that the correct Fibonacci number is printed. Then, click the button `Previous Fibonacci`. Verify that the correct Fibonacci number is printed. If your code does not work as expected, fix it.

(e) Once you have implemented the functions `nextFib` and `prevFib`, be sure to Journal any bugs you encounted (if any) and how you fixed them.

Coding Setup

We will assume that you have already installed Git Bash, NPM, and VSCode on your machine. Bash provides an environment where we can type commands to run programs. NPM (node package manager) reads and runs commands from a `package.json` file that describes the organization of our coding project. VSCode is an editor and is optional; you can replace it by any other editor of your choice.

Perform the following steps to get started with the code provided for this section:

1. Navigate to an appropriate directory on your machine in the terminal (using `cd`). Then, run the command

   ```
   git clone https://gitlab.cs.washington.edu/cse331-24au-materials/sec01
   ```

   This will copy the starter code into a new subdirectory of the current directory called `sec01`.

2. Change into the `sec01` directory (`cd sec01`). Then, run the command

   ```
   npm install --no-audit
   ```

   This will have `npm` download all of the dependencies referenced in the `package.json` file and store them locally (in the `node_modules` subdirectory) so that you can call them from your own code.

   Do not forget to include the `--no-audit` parameter. If you do forget, you will see some scary errors claiming that there are critical vulnerabilities in the code you downloaded that require fixing. This is not true. (If you want to confirm, you can run `npm audit --production` to see that there are no vulnerabilities in the code we are including in the app. `npm audit` is confused and thinks that some of the developer scripts are part of the production code, when they are not.)

3. Start the app by running the command `npm run start`, opening your browser, and navigating to `http://localhost:8080`. You should see a simple web page with a textbox and 3 buttons