COMP 3123 | Full stack Development – I

Test – 1 (6%)

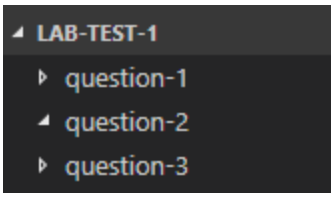
Lecture Time: 06:00 PM to 08:00 PM

Instructions for a lab test submission involving JavaScript (ES6) and Node.js. Here's a quick checklist and guide for completing your lab test and submission:

Developer Note:

When working on your questions, please create separate folder for your work. This way you won't be putting all your code in the same file, which can pollute the global namespace. In short, it will prevent you from overwriting your own work and causing your code to compile incorrectly.

Organize your folder structure in this way (Use it as a reference)



1. Folder Structure and Code Organization

- **Create a folder for your work** to ensure you don't pollute the global namespace.
    - Example: ***studentid_comp3123_labtest1/***
- Create separate files/modules inside the folder if needed to avoid overwriting code.

2. Documentation Resources

- **JavaScript ES6 Documentation**: MDN JavaScript Docs
- **Node.js Documentation**: Node.js Docs

3. Git Repository Setup

- Create a **private Git repository** on GitHub.
    - Repository name: studentid_comp3123_labtest1 (replace studentid with your actual ID).
    - Push your code to this repo throughout the development process.
    - **Make the repository public** after the submission deadline **(Oct 3rd, 8:00 PM)**.

4. Submission Requirements

- **ZIP of the project**: Ensure you include all files in the ZIP (source code, readme, etc.).
- **GitHub repository link**: Ensure the repo is public when you submit the link.
- **Screenshots**: Take screenshots of the output from your code and include them in the submission as a separate docx/pdf file.

5. Due Date

- **Deadline**: Thursday, 03th October 2024, 08:00 PM.
- **Late submissions**: ZERO percent will be awarded for late submissions, so ensure everything is submitted on time!

Question 1: ES6 Features

- Create a script with a function named ***lowerCaseWords*** that takes a mixed array as input.

The function will do the following.

  - return a promise that is resolved or rejected
  - filter the non-strings and lower case the remaining words

- Input

```
const mixedArray = ['PIZZA', 10, true, 25, false, 'Wings']
```

- Output

```
[ 'pizza', 'wings' ]
```

Question 2: Promises

- Given the script file ***callbacks.js***, write a script that does the following:
  - Create a method ***resolvedPromise*** that is similar ***to delayedSuccess*** and resolves a message after a timeout of 500ms.
  - Create a method ***rejectedPromise*** that is similar to ***delayedException*** and rejects an error message after a timeout of 500ms.
  - Call both promises separately and handle the resolved and reject results and then output to the console

callbacks.js

```
const delayedSuccess = () => {
    setTimeout(() => {
        let success = {'message': 'delayed success!'}
        console.log(success);
      }, 500)
}

const delayedException = () => {
    setTimeout(() => {
        try {
            throw new Error('error: delayed exception!');
          } catch (e) {
            console.error(e);
          }
      }, 500)
}

delayedSuccess()
delayedException()
```

- Output

```
{ message: 'delayed success!' }
{ error: 'delayed exception!' }
```

Question 3: File Module

- Create a script that will do the following:

1. **Remove Log files**
    - remove all the files from the Logs directory, if exists
    - output the file names to delete
    - remove the Logs directory

2. **Create Log files**
    - create a Logs directory, if it does not exist
    - change the current process to the new Logs directory
    - create 10 log files and write some text into the file
    - output the files names to console

- **Hint:** use the fs module and path module, and the process current working directory to build directory path. It is acceptable, to have a remove.js script and separate add.js script.

> Output