Javascript-Nesting if else statements

Objective

In this exercise, you will learn how to use nested `if/else` statements with multiple conditions in JavaScript. This technique allows you to evaluate more complex conditions and take action based on the results of these evaluations.

Instructions

You are provided with a blank `index.js` file. Your task is to fill in the file by completing the following steps.

Here's what you need to do:

1. **Declare Variables for Comparison**

Declare three variables: `num1`, `num2`, and `num3`.
- **num1**: Declare a variable `num1` using the `var` keyword and assign it the value `10`.
- **num2**: Declare a variable `num2` using the `var` keyword and assign it the value `20`.
- **num3**: Declare a variable `num3` using the `var` keyword and assign it the value `30`.

These variables will be used for comparison and to demonstrate nested conditional statements.

2. **Using Nested `if/else` with Multiple Conditions**

You will use `if/else` statements to compare the variables `num1`, `num2`, and `num3`.

- **First Condition**: Use an `if` statement to check if `num1` is less than `num2` and `num2` is less than `num3`. If true, log "num1 is less than num2, and num2 is less than num3" to the console. If false, check the next condition.

- **Second Condition**: Inside the `else`, use a nested `if/else` to check whether `num1` is greater than `num2`, or `num2` is equal to `num3`. If true, log "num1 is greater than num2, or num2 is equal to num3". If false, log "Neither condition is true".

3. **Nested `if/else` with Another Condition**

Now, use another nested `if/else` statement to check if `num1` is equal to `10`.

- **First Check**: If `num1` is `10`, then check if `num2` is `20`. If both conditions are true, log "num1 is 10 and num2 is 20" to the console. If `num2` is not `20`, log "num1 is 10 but num2 is not 20".

- **Second Check**: If `num1` is not `10`, log "num1 is not 10".

Mandatory Assessment Guidelines:

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**
2. **To open the command terminal the test takers, need to go to**

   Application menu (Three horizontal lines at left top) -> Terminal ->New Terminal.
3. **This editor Auto Saves the code.**
4. **If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**
5. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**
6. **This is a web-based application, to run the application on a browser, use the internal browser in the workspace. Click on the second last option on the left panel of IDE, you can find Browser Preview, where you can launch the application.**

   Note: The application will not run in the local browser
7. **You can follow series of command to setup Angular environment once you are in your project-name folder:**
   a. **npm install -> Will install all dependencies -> takes 10 to 15 min.**
   b. **node src/index.js -> To compile and run the index.js file.**
   c. **node src/test/custom-grader.js -> to run all test cases. It is mandatory to run this command before submission of workspace -> takes 5 to 6 min.**

8. **Once you are done with development and ready with submission, you may navigate to the previous tab and submit the workspace. It is mandatory to click on "Submit Assessment" after you are done with code.**
9. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**