BTI325 Test 1

[30 marks]

# Instructions: Complete all questions in the spaces provided. This quiz is worth 10% of your final mark and you will have exactly 30 minutes to complete it.

# Question 1 (10 Marks):

For each multiple-choice question, identify the correct answer for the given question. There is only **one** correct answer provided for each question:

|  |  |
| --- | --- |
| 1. The main code editor used in this course is: 2. Brackets 3. Sublime Text 4. Eclipse 5. Visual Studio Code | 1. The JavaScript runtime environment used in this course is: 2. Runtime.js 3. Execute.js 4. Node.js 5. Process.js |
| 1. The following command will execute server.js using node: 2. execute server.js --node 3. npm server.js 4. server.js run --node 5. node server.js | 1. "JavaScript" is a "dialect" of: 2. C++ 3. TypeScript 4. ActionScript 5. ECMAScript |
| 1. The HTTP Protocol states that all HTTP Requests & Responses are composed of: 2. Data, Location, Protocol and Headers 3. Start Line, Headers, Blank Line and Body 4. Headers and Body 5. Data, Protocol and Location | 1. To expose variables and functions from a module, the following is used: 2. module.exports 3. export.module 4. this.module 5. module.output |
| 1. An example of an HTTP header is: 2. Content-Type: text/html 3. Transfer-Encoding => Chunked 4. [Content-Encoding]: qzip 5. Connection = "keep-alive" | 1. Publicly available modules (ie: "moduleName") can be installed using the following command: 2. node install moduleName 3. npm save moduleName 4. npm install moduleName --save 5. node -npm moduleName --save |
| 1. The following command is used to generate a package.json file in the current directory: 2. npm generate 3. npm init 4. node package.json --create 5. git package.json | 1. To automatically install the dependencies listed in our package.json file, we use the command: 2. npm fetch 3. npm getAll 4. npm install 5. npm install \* |

# Question 2 (8 Marks):

Write a module called "counter" that exposes the following **3 functions** and has a variable named **privateCounter** which is a non-negative integer and initialized with 0. (**NOTE**: You may assume that you're writing your module inside a separate file "counter.js")

The module’s three functions:

## increase(num)

This function adds the value of "num" to the variable **privateCounter.**

## decrease(num)

This function decreases the variable **privateCounter** with the value of "num". If negative number occurs, the **privateCounter s**hould be set to 0.

## value()

This function will return the value of **privateCounter.**

## Usage Example (assuming "counter" references your module):

counter. increase (1);

counter. decrease (2);

console.log(counter. value ()); *// outputs 0*

counter. increase (5);

counter. decrease (2);

console.log(counter. value ()); *// outputs 3*

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

# Question 3 (12 Marks):

Given the following code for server.js - fill in the blanks to ensure that it functions correctly when executed.

**var** express = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ("express");

**var** app = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;

**var** path = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ("path");

**var** HTTP\_PORT = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ || 8080;

*// call this function after the http server starts listening for requests*

**function** onHttpStart() {

console.log("Express http server listening on: " + \_\_\_\_\_\_\_\_\_\_\_);

}

*// setup a 'route' to listen on the default url path (http://localhost)*

\_\_\_\_\_\_\_\_\_\_\_ ( "\_\_\_\_\_\_\_" , **function**(req,res){

res.\_\_\_\_\_\_\_\_\_\_\_\_\_ ("Hello World<br /><a href='/about'>Go to the about page</a>");

});

*// setup another route to listen on /about*

\_\_\_\_\_\_\_\_\_\_\_\_ ( "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_", **function**(req,res){

res.\_\_\_\_\_\_\_\_\_\_\_\_\_ (path.join(\_\_dirname + "/views/about.html"));

});

*// setup http server to listen on HTTP\_PORT*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(HTTP\_PORT, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_);