Class Notes 10/9/2017

Zazil Toma M/W 7/2017 Cohort

* Constructors should begin with a capital letter as a coding standard
* Constructors are considered classes in other languages but js isn’t a class-based language.
* Prototype – can write a function outside the constructor by using this. i.e. Programmer.prototype.printStats function() {…
* When we have a constructor, you’re building up these things in memory. So we’re creating the function every time if it’s inside. If we use prototype outside the constructor, it isn’t being stored in memory. Can make a big difference with larger scripts.
* Keep properties and constructors outside the constructor whenever possible.
* Doesn’t matter where your new objects are vs the functions because they’ll have access to them because of prototype.
* You can load start scripts in the package.json file in “scripts”. i.e. “scripts”: {

“start”: “node programmerWithPrompt.js - - host= localhost”,

Etc… } Then in file you include db = process.argv

* “main” – defines the module that your application represents
* To create a number of objects with a constructor, you can make an if stmt to check number of count i.e. see programmersLoop-recursionArray.js. if it’s less than the count then it’ll go through the function, add the object to an array, add to the count and then call itself. Then it prints all in the else.
* Without recursion – each question is asked in the inquirer function at the same time for however many loops.
* Did not finish player activity (see my activity.js file) and watched solution. Learned that because we are calling the createPlayer function again before exiting the .then function that adds the player to the array, it is building up in memory. At this scale it’s ok, but could be problematic. It’s called a call stack. “Call back hell” – because we are waiting for an operation to complete (because js is asynchronous), we cannot go to the next operation.
* Movie activity – can export only parts of a module so don’t have to expose all of the file. When requiring a file, we are only getting what is being exported from the file.
* MVC – models, views and controllers: can modify objects using the models. Controllers manipulate the models and the view is like the html where you see it. So manipulating the data the same way every time and the data is displayed on the views based on the way the controller manipulated the models. Makes it all maintainable.
* Models represent the nouns and controllers call the functions that mess with the nouns. Views literally have everything that display the app.
* There is a difference between module.exports and exports.