|  |  |
| --- | --- |
| Sprint 9, Assignment 5.7 Please also update the doc name with correct numbers. | |
| Assignment type: JS Interactive | |
| Assignment name  Can remain the same as the assignment topic name, or…  can be created depending on the task in the active form, e.g. “Create your own X” | “Factorials” |
| BDG Description\*  What is the task and why is it important?  In this part, it’s encouraged to think about storytelling and future job-specific context e.g. “You’ve been asked to help out X with Y. They want Z on their website, yet aren’t too sure on how to achieve it”.  Drawing on practical examples and adding context can increase a student's motivation and increase long term learning according to Instructional Design principles, because this helps to relate some familiar or existing knowledge to new bits of information.  This will appear in the course as text before a button, leading to the interactive platform assignment. | Take a break from your holiday coding and dig deeper on recursion: create a factorial function from scratch! |
| The Assignment  A short specific description of the assignment and tasks using bullet points that the student will need to do. | * Create a factorial function |

|  |  |  |  |
| --- | --- | --- | --- |
| Steps  Step-by-step instructions on what the student should do. | | |  |
| Step # | Step  Write each small step of the task | At least 1x hint(s)  Write some text (not necessarily, but can also be a part of code if relevant) which would hint the student to figure out the correct step forward. | The correct output should be…  (if relevant to the task) |
| 1 | Declare a function that a) takes in an argument, and returns the result of b) multiplying that argument by itself depricated by 1 |  | function factorialize(num) {          return (num \* (num - 1));        }        console.log(factorialize(5)); |
| 2 | Make this recursive by feeding the depreciated argument back into the function |  | function factorialize(num) {          return (num \* factorialize(num - 1));        }        console.log(factorialize(5)); |
| 3 | Make your function stop \*if\* it returns a number equal to 0 | Hint: you likely encountered an error, which is caused by an infinite loop. Using a conditional to check for 0 will allow you to avoid producing a negative infinite loop ... and otherwise have it continue to perform the recursive process | function factorialize(num) {          if (num == 0) {                return 1;          } else {                return (num \* factorialize(num - 1));            }        }        console.log(factorialize(3)); |
| 4 |  |  |  |
| 5 |  |  |  |
| 6 |  |  |  |
| ... |  |  |  |