campus.owlab 2017 || Function – param – return II

1 / Écrire une fonction **deleteNodes**,

deleteNodes prend un paramètre list.
list est une nodeList contenant des éléments HTML à supprimer.
Retourner true si tous les éléments on été supprimés, false sinon.

2 / Écrire une fonction **isOdd** prenant un paramètre n

Si n n'est pas un nombre, lancer une nouvelle Erreur avec l'instruction throw.
Sinon retourner true si n est impair, false sinon.

3 / Écrire une fonction **sum**.

sum prend un array a de nombres en paramètre.
Si a contient une valeur non numérique, lancer une nouvelle Erreur avec l'instruction throw. Sinon retourner la somme de tous les éléments de a.

4 / Écrire une fonction **getNodesMetrics**, prenant un paramètre nl

nl est une nodeList contenant les éléments HTML à analyser. Retourner un tableau d'objets contenant, pour chaque élément de nl, sa hauteur et sa largeur en pixels.

5 / Écrire une fonction **usingCallback** prenant un paramètre clbk.

clbk est une fonction callback.
Exécutez clbk et passez lui un message en paramètre.
Afficher le message dans le corps de clbk.
Attention: clbk n'est pas déclaré dans le corps de usingCallback.

6 / Écrire une fonction **usingCallback2**, prenant deux params clbk et url.

usingCallback2 reprend le principe de l'exercice précédent:
clbk est une fonction callback.
url est une chaîne de caractère (ex: https://opendata.paris.fr/api/records/1.0/search/?dataset=liste_des_prenoms_2004_a_2012&rows=1000&facet=prenoms&facet=sexe&facet=annee)

Dans le corps de usingCallback2, un appel AJAX sur url.
Envoyer le résultat sous forme de tableau d'objets JS en paramètre de clbk.
Afficher chaque objet dans la console dans le corps du callback.

7/ Écrire une fonction **Hotel**, prenant un paramètre infos.

infos est un objet littéral comprenant les propriétés suivantes :

- nom (string)
- nombreEtoile (number)
- ville (string)
- aPiscine (boolean)
- nbChambres (number)
- nbChambresReservees (number)
- urlSite (string)

Ne pas utiliser return pour cette fonction.
Hotel est un constructeur. Initialiser toutes les propriétés d'infos sur son propre mot-clé this. Lever une Erreur avec l'instruction throw si un paramètres n'a pas le bon type.
Tester une affectation avec l'instruction var h = new Hotel({ }). Que retourne Hotel?

8 / Écrire une fonction Hotel.**reserverChambre**, prenant un paramètre n.

n est un number indiquant le nombre de chambres à réserver.
reserverChambre est associée au prototype d'**Hotel**.
Mettre à jour le nombre de chambre réservées.
Retourner le nombre de chambre disponibles
Vérifier avec deux instances d'Hotel, en effectuant des logs avant et après l'appel à reserverChambre.

9/ Rédiger la documentation des fonctions en consultant le site usejsdoc.org.
Générer un fichier HTML avec l'utilitaire de ligne de commande fourni par jsdoc.