Programmation web en js

TD n°1 : fonctions

Exercice 1 : fonctions basiques

1. écrire une fonction qui retourne le minimum de 2 nombres, et une fonction qui retourne le max de 2 nombres.
2. écrire une fonction qui affiche (en utilisant console.log) les nombres de 1 à 100 ; de plus, pour les nombres divisibles par 3, affiche "woueee", pour les nombres divisibles par 5, affiche "yoooo", et pour le nombre 73, affiche "Biinnngooo",
3. écrire une fonction power() qui reçoit 2 nombres *n* et *x ,* et retourne x^n ; le calcul est fait en utilisant une boucle,
4. écrire la même fonction qui retourne x^n , mais utilise un appel *récursif* ; (rappel : $x^n = x * x^{n-1}$)

Exercice 2 : fonctions avancées : closures et fonctions retournées

1. écrire une fonction creerMultiplicateur() qui reçoit un paramètre entier *n* et retourne une fonction qui multiplie son paramètre x par *n,*
2. écrire une fonction creerSequence() qui reçoit 2 paramètres : une valeur initiale *init* et une valeur d'incrément *step*, et retourne une fonction qui délivre à chaque appel les valeurs successives de la séquence démarrant à *init* et incrémentées de *step*.
3. écrire une fonction qui permet de parcourir la suite de fibonacci. La fonction reçoit 2 arguments qui sont les 2 valeurs initiales de la suite, et retourne une fonction qui, à chaque appel, délivre les valeurs successives de la suite. (rappel : la suite de fibonacci est la suite $u_n = u_{n-1} + u_{n-2}$),
4. modifier la fonction creerMultiplicateur() pour que, dans le cas où elle reçoit 2 paramètres *n* et *x*, elle retourne *n*x*, et si elle reçoit 1 seul paramètre, elle retourne la fonction qui multiplie son paramètre par *n.*
5. modifier sur le même principe la fonction power() : si elle reçoit 2 paramètres n et x, elle retourne x^n, et si elle reçoit 1 seul paramètre n, elle retourne une fonction qui met à la puissance n. Ecrire la fonction de manière récursive.
6. écrire une fonction formatter() qui construit une fonction de formatage de messages en ajoutant un n° de message incrémenté à chaque appel. On l'utilisera ainsi :

```
var format = formatter(10) ;   // on numérote les messages à partir de 10
format( 'le ciel est blanc') ; // → retourne "10 : le ciel est blanc"
format('la neige est bleue') ; // → retourne "11: la neige est bleue"
```

7. écrire une fonction write() qui écrit le message qu'elle reçoit sur la console. écrire une fonction writeAlert() qui écrit le message qu'elle reçoit en utilisant alert(),
8. écrire une fonction logger(), qui reçoit en paramètre une fonction de formatage et une fonction d'écriture de messages, et retourne une fonction de log qui reçoit un message en paramètre, le formate et l'écrit avec les fonctions passées en paramètres.