Exercices JavaScript

Calculer la somme de deux nombres:

- **Exercice:** Demander à l'utilisateur de saisir deux nombres et afficher leur somme.

Tableau et boucle:

- **Exercice:** Créer un tableau de fruits et afficher chaque fruit dans une liste à puces.

Fonction pour vérifier si un nombre est pair:

- **Exercice:** Écrire une fonction qui prend un nombre en entrée et retourne true si le nombre est pair, false sinon.

Objet et méthode:

- **Exercice:** Créer un objet représentant une personne avec les propriétés nom, âge et une méthode pour afficher ses informations.

Asynchrone avec Promise:

- **Exercice:** Simuler une requête API avec une Promise qui résout après 2 secondes et affiche un message de succès.

Manipulation du DOM : Création d'éléments dynamiques

- **Exercice:** Créer un formulaire permettant à l'utilisateur de saisir un texte. À chaque saisie, ajouter le texte saisi dans une liste non ordonnée à la page.

Fonction récursive : Calcul de la factorielle

- **Exercice:** Écrire une fonction récursive pour calculer la factorielle d'un nombre.

Création d'un objet "Livre"

- **Exercice:** Créer un constructeur pour créer des objets "Livre" avec les propriétés titre, auteur et année de publication. Ajouter une méthode pour afficher les informations du livre.

Asynchrone avec async/await : Requête API

- **Exercice:** Utiliser async/await pour faire une requête à une API (par exemple, l'API de l'heure actuelle) et afficher le résultat dans un élément HTML.

Programmation orientée objet : Héritage

- **Exercice:** Créer une classe Animal avec les propriétés nom et age. Créer ensuite des classes Chat et Chien héritant de Animal et ajoutant des propriétés spécifiques (par exemple, race pour le chien).

Créer une fonction qui mémorise ses appels

- **Exercice:** Créer une fonction qui, à chaque appel, ajoute 1 à une valeur interne et retourne cette valeur.

Créer un compteur incrémental utilisant des objets et des closures (version ES6+)

- **Exercice:** Créer un compteur incrémental utilisant des objets et des closures .

EXEMLPES D'UTILISATION DES CLASSES EN JAVASCRIPT

Exemple avec une classe :

```
class Counter {
  constructor() {
    this.count = 0;
  }

  increment() {
    return ++this.count;
  }
}

const counter = new Counter();
console.log(counter.increment());
```

Exemple avec une classe Personne

```
class Personne {
  constructor(nom, age) {
    this.nom = nom;
    this.age = age;
  }

  saluer() {
    console.log(`Bonjour, je m'appelle ${this.nom} et j'ai ${this.age}
 ans.`);
  }
}

// Créer une instance de la classe
const personne1 = new Personne("Alice", 30);
personne1.saluer(); // Affiche : Bonjour, je m'appelle Alice et j'ai 30 ans.
```

Héritage et polymorphisme avec des classes Animal et Chat

```
class Animal {
  constructor(nom) {
    this.nom = nom;
  }
  faireDuBruit() {
    console.log("L'animal fait du bruit.");
  }
}
class Chat extends Animal {
  faireDuBruit() {
    console.log("Miaou !");
  }
}
const chat = new Chat("Félix");
chat.faireDuBruit(); // Affiche : Miaou !
```

Créer une classe CompteBancaire avec des getters et setters

```
class CompteBancaire {
  constructor(titulaire, solde) {
    this.titulaire = titulaire;
    this.solde = solde;
  }

  get solde() {
    return this._solde;
  }

  set solde(nouveauSolde) {
    if (nouveauSolde < 0) {
      console.error("Le solde ne peut pas être négatif.");
    } else {
      this._solde = nouveauSolde;
    }
  }
  retirer(montant) {
    if (montant <= this.solde) {
      this.solde -= montant;
      console.log(`Retrait de ${montant}€ effectué. Nouveau solde : ${this.solde}€`);
    } else {
      console.error("Solde insuffisant.");
    }
  }
}
const compte = new CompteBancaire("Bob", 1000);
compte.retirer(200); // Retrait de 200€ effectué. Nouveau solde : 800€
compte.solde = -50; // Affiche un message d'erreur car le solde ne peut pas être négatif
```

Créer une classe CompteBancaire avec des getters et setters

```
class Rectangle {
  constructor(longueur, largeur) {
    this.longueur = longueur;
    this.largeur = largeur;
  }

  calculerSurface() {
    return this.longueur * this.largeur; [1]

  }
}
// Création d'un rectangle
const monRectangle = new Rectangle(5, 3);

// Calcul de la surface
const surface = monRectangle.calculerSurface();

// Affichage du résultat
console.log("La surface du rectangle est de", surface, "unités carrées.");
```

Exemple d'utilisation plus avancée:

```
class Rectangle {
  // ... (même code que précédemment)

  // Méthode pour afficher les dimensions

  afficherDimensions() {
    console.log(`Longueur : ${this.longueur}, Largeur : ${this.largeur}`);
  }
}

const rectangle1 = new Rectangle(10, 7);
rectangle1.afficherDimensions(); // Affiche : Longueur : 10, Largeur : 7
console.log("Surface :", rectangle1.calculerSurface());
```