Programmation asynchrone Checkpoint en JavaScript

Dans ce point de contrôle, nous verrons les sujets de l'itération avec async/await, de l'attente d'un appel, de la gestion des erreurs avec async/await, du chaînage async/await, de l'attente de requêtes concurrentes et de l'attente d'appels parallèles

Tâche 01 :

Itération avec Async/Await : écrivez une fonction asynchrone **iterateWithAsyncAwait** qui prend un tableau de valeurs et enregistre chaque valeur avec un délai de 1 seconde entre les journaux.

Solution :

```
const values = [1, 2, 3, 4, 5];

iterateWithAsyncAwait(values)
   .then(() => {
      console.log('Toutes les valeurs ont été traitées.');
   })
   .catch(error => {
      console.error('Une erreur s\'est produite :', error);
   });
```

Tâche 02 :

En attente d'un appel : créez une fonction asynchrone **waitCall** qui simule la récupération de données à partir d'une API. Utilisez **wait** pour attendre la réponse de l'API, puis enregistrez les données.

Solution :

```
waitCall()
   .then(response => {
      console.log('Traitement des données récupérées :', response);
   })
   .catch(error => {
      console.error('Une erreur s\'est produite :', error);
   });
```

Tâche 05 :

En attente d'appels parallèles : écrivez une fonction **parallelCalls** qui prend un tableau d'URL et récupère les données de chaque URL simultanément à l'aide **de Promise.all()** . Enregistrez les réponses une fois que toutes les demandes sont terminées.

Solution :

```
async function parallelCalls(urls) {
   try {
      // Créer un tableau de promesses pour chaque appel d'API
      const promises = urls.map(url => fetch(url));
```

```
        // Attendre que toutes les promesses se résolvent avec succès
        const responses = await Promise.all(promises);

        // Attendre et extraire les données JSON de chaque réponse
        const dataPromises = responses.map(response => response.json());
        const dataList = await Promise.all(dataPromises);

        // Enregistrer les réponses une fois que toutes les demandes sont terminées
        console.log('Réponses récupérées :', dataList);
    } catch (error) {
        console.error('Une erreur s\'est produite lors des appels parallèles :', error);
    }
}
```