**Java**

Quelle est la différence entre une référence Java et une valeur ?

Comment fonctionne Java ?

Quelle est la différence entre "==" et "equals()" pour comparer des objets Java ?

Qu'est-ce que l'héritage en Java ?

Comment fonctionne la surcharge de méthode en Java ?

Qu'est-ce que la généricité en Java et comment l'utiliser ?

Comment fonctionne le ramasse-miettes (Garbage Collector) en Java ?

Qu'est-ce que l'interface en Java et comment l'utiliser ?

Comment gérer les exceptions en Java ?

Qu'est-ce que la programmation orientée objet (POO) en Java et comment l'utiliser ?

**Javascript**

Comment fonctionne la portée des variables en JavaScript ?

Quelle est la différence entre une fonction classique et une fonction fléchée en JavaScript ?

Comment fonctionne le concept de closures en JavaScript ?

Comment gérer les erreurs en JavaScript ?

Qu'est-ce que le "hoisting" en JavaScript ?

Comment fonctionne l'héritage des objets en JavaScript ?

Qu'est-ce que la programmation asynchrone en JavaScript et comment l'utiliser ?

Comment utiliser les classes en JavaScript ?

Comment fonctionne la gestion de mémoire en JavaScript ?

Comment utiliser les modules en JavaScript ?

Comment fonctionnent les callbacks en JavaScript et comment les utiliser ?

Comment fonctionne l'asynchronisme en JavaScript et comment gérer les promesses ?

Qu'est-ce que le "spread operator" en JavaScript et comment l'utiliser ?

Comment fonctionnent les closures en JavaScript et comment les utiliser de manière efficace ?

Comment utiliser les fonctions de rappel (callback) dans les événements de DOM en JavaScript ?

**Spring Boot:**

**React.js:**