Au wok

Ce projet est composé de deux parties. La première partie peut fonctionner sans la seconde (mais pas l'inverse). Lorsque vous commencerez la deuxième partie, faites une sauvegarde de la première à toute fin utile.

Partie 1 : Simulation multithreadée

Un restaurant vous demande de simuler son fonctionnement en vue d'améliorer la fluidité de son service. Si le but n'est pas ici d'avoir une simulation hautement réaliste, il sera toutefois nécessaire de correctement simuler les situations de concurrences entre clients, entre clients et employés, *etc.*

Votre programme principal, `Restaurant`, représente le restaurant, qui a un nombre de places limité à 25 (le nombre de client total peut dépasser ce nombre : lorsque le restaurant est plein, les clients qui souhaitent rentrer attendent que d'autres sortent.) En plus de cette capacité limitée, le restaurant devra contenir, initialiser et démarrer les éléments suivants :

Le buffet. Le buffet contient quatre compartiments, chacun contenant initialement 1kg de nourriture crue différente. Plus précisément, il y a :

- un compartiment de 1kg de poisson cru
- un compartiment de 1kg de viande cru
- un compartiment de 1kg de légumes crus
- un compartiment de 1kg de nouilles froides

Les clients et les employés peuvent accéder en même temps au buffet si ce n'est pas dans le même compartiment. En revanche, deux client ou employés ne pourront pas accéder en même temps au même compartiment. Si les compartiments ont une capacité de 1 kg (quel que soit leur contenu), chaque fois qu'un client se sert, la quantité de nourriture restante dans le compartiment est diminué du poids de la portion prise par ce client. Aussi, un client qui souhaite une quantité d'un ingrédient supérieur à ce qu'il reste dans le compartiment correspondant devra attendre qu'il soit réapprovisionné, par l'employé du buffet.

L'employé du buffet. L'employé est chargé de veiller à ce qu'il reste suffisamment de nourriture dans chacun des compartiments. Pour cela, il fait le tour des compartiments afin de vérifier les quantités restantes de chacun : si un compartiment a moins de 100g de nourriture restante, il le remplit (afin que sa nouvelle quantité soit de nouveau de 1kg). L'employé n'est jamais à court de nourriture. Il est important de veiller à ce que l'employé ne remplisse pas un compartiment alors qu'un client est en train de se servir (sauf bien sûr si ce client est en train d'attendre qu'il soit rempli.)

Le stand de cuisson. Le stand de cuisson est l'endroit ou les clients doivent passer après avoir rempli leur assiette au buffet, afin de faire cuire ce qu'ils y ont pris. Le stand de cuisson est le lieu où doivent se synchroniser le cuisinier et les clients, selon les règles qui sont décrites par la suite.

Le cuisinier. Le cuisinier est là pour faire cuire les plats des clients une fois que ceux-ci ont leur assiette remplie de nourriture crue prise au niveau du buffet. Le cuisinier doit se synchroniser avec les clients (au niveau du stand de cuisson) pour faire cuire leur plat un par un (sans forcément respecter l'ordre d'arrivée des clients à son stand). Le cuisinier :

1. attend un client ;
2. fait cuire sont plat (il ne peut faire cuire qu'un seul plat à la fois) ;
3. notifie le client quand c'est cuit, et recommence.

Des clients. Le nombre de clients total pour la simulation sera d'au moins 40. Chaque client, dans cette ordre :

1. entre dans le restaurant (attend qu'une place se libère si besoin) ;
2. se rend au buffet et prendre une portion d'un poids aléatoire entre 0 et 100g de chaque ingrédient ; Il n'est pas nécessaire de simuler la prise de couvert. Chaque prise de portion prendra entre 200 et 300 ms. Comme dit plus haut, le client devra attendre si la quantité restante de nourriture est inférieure à ce qu'il souhaite ;
3. fait la queue au stand de cuisson (il ne sera pas nécessaire de simuler une file d'attente), puis attend la fin de la cuisson de son plat ;
4. mange (en un temps aléatoire compris entre 1 et 2 secondes) ;
5. sort (c'est-à-dire libère sa place, pas besoin de modéliser le paiement).

Partie 2 : API Rest de suivi

Une fois que votre simulation vous semble satisfaisante, vous devrez développer une API Rest permettant de la contrôler. Plus précisément, en utilisant la librairie Restlet, vous devrez développer une API permettant de :

- lister les clients avec leur état (`GET /clients`)
- faire apparaitre un nouveau client et le démarrer (`POST /clients`)
- retourner l'état du buffet (`GET /buffet`)

Les clients auront successivement les états suivants :

1. WAITING_TO_ENTER - en attente pour rentrer
2. AT_THE_BUFFET - en train de se servir (ou d'attendre un remplissage)
3. WAITING_FOR_THE_COOK - en attente du cuisinier ou de la cuisson de son plat
4. EATING - en train de manger
5. OUT - sorti

L'état du buffet est simplement, pour chaque compartiment, la quantité de nourriture restante.

Conseils et instructions

Une fois votre partie 1 complète et sauvegardée, une façon de commencer la partie 2 est de repartir du code du TP3 et de remplacer la classe `InMemoryDatabase` par votre classe `Restaurant` et de mettre toutes vos classes de la partie 1 dans le package Backend. Votre `main` de la partie 1 ne sera plus utilisée.

Le TP doit être envoyé au plus tard le mardi 10 décembre 2024 (23h59, CET) à votre chargé de TP. L'objet de votre mail sera [TP-CSR Gr<X>] <Nom1> <Nom2> [1]. Ce courrier devra contenir (ou permettre de récupérer) :

1. Les sources Java commentées avec toutes les instructions et fichiers annexes éventuels nécessaires à l'exécution de votre programme ;
2. Un rapport court permettant de comprendre votre code et vos choix d'implémentations. En particulier, il sera indispensable de fournir une liste des problèmes de synchronisation que vous aurez identifiés et comment vous les aurez résolus.

1. Merci de respecter ce format, permettant de retrouver votre travail dans le flot de mails.