Thème 11 - TME

Exercice

Lors de chaque TME, vous devrez créer un répertoire pour chaque exercice, y mettre les fichiers s'y rapportant et soumettre ce répertoire à la fin du TME.

Exercice 15 – Le réveillon au restaurant

Nous nous intéressons à l'organisation du réveillon dans un restaurant. Le restaurant dispose de `10` tables pouvant chacune accueillir 2 personnes. Les tables peuvent être déplacées de manière à accueillir des groupes. Par exemple, un groupe de 5 personnes occupera 3 tables.

Les clients sont organisés en groupes mais se comportent de manière indépendante : le thread principal crée des instances de la classe `GroupeClients` et la création d'un groupe induit la création de l'ensemble des clients qui le composent, qui sont tous exécutés par des threads indépendants.

Question 1

Proposez une implémentation de la classe `TestRestaurant` exécutée par le thread principal.

Question 2

Proposez une implémentation de la classe `Restaurant`. Le restaurant est caractérisé par son nombre de tables, et la classe propose une méthode `reserver` qui renvoie un numéro de réservation ou **`null`** lorsqu'il n'y a pas suffisamment de tables disponibles pour accueillir le groupe.

Nous nous intéressons maintenant à la classe `GroupeClients`. Nous souhaitons associer un identifiant à chaque groupe de manière à pouvoir tracer l'exécution. De plus, la construction d'un groupe de clients doit créer et lancer les threads correspondant à chacun des clients. Nous souhaitons conserver une référence sur chacun des threads créés.

Question 3

Donnez les variables d'instance et le constructeur de la classe `GroupeClients`.

Les clients d'un groupe sont tous identiques, en particulier il n'y a pas un leader chargé de la réservation. Tous vont donc appeler une méthode `reserver` de la classe `GroupeClients` dont on définit le comportement comme suit :

si le numéro de table du groupe n'est pas affecté, la méthode appelle la méthode `reserver` de la classe `Restaurant`. Si celle-ci renvoie **`null`**, il n'y a pas suffisamment de place pour accueillir le groupe, la tentative de réservation est un échec. Le client modifie alors l'*interrupted status* de l'ensemble des clients de son groupe pour leur signaler cet échec. Un client dont l'*interrupted status* est positionné doit donc immédiatement interrompre sa tentative de réservation.

Si tout se passe bien, la méthode affecte un numéro de table au groupe.

Question 4

Proposez une implémentation de la méthode `reserver` de la classe `GroupeClients`.

Nous nous intéressons maintenant au comportement d'un client. Celui-ci essaie de faire une réservation pour son groupe. Si la réservation échoue, il se termine. Par contre, si le groupe obtient une table, le client va prendre un temps aléatoire pour se rendre au restaurant. Lorsque le dernier membre du groupe arrive à table, il affiche un message signalant que le groupe est au complet et prêt à passer la commande.

Question 5

Modifiez l'implémentation de la classe `GroupeClients` en conséquence et proposez une implémentation de la classe `Client`.