



1 Présentation

Dans ce sujet, nous présentons quelques éléments d'un logiciel pour gérer des réunions. Les noms des classes, attributs et méthodes vous sont indiqués en police de caractères `TrueType`. Les signatures des méthodes ne vous sont pas toujours données, dans ce cas, vous devez les déterminer.

Une `SalleReunion` accueillera des `Reunions`. Nous considérons ici deux catégories de réunions (qui seront des sous-classes de la classe `Reunion`) : la `ReunionPresentiel` dont tous les participants sont sur place et la `VideoConference` dont une partie seulement des participants sont en présentiel, et les autres participants sont sur des sites distants.

2 Représentation des réunions

On ne créera jamais d'instance directe de la classe `Reunion`, seulement des instances de ses sous-classes.

Toutes les réunions (`Reunion`) sont décrites par :

- un sujet (`sujet`),
- une date au format "AAAAMMJJ" (`date`). Par exemple la chaîne "20191218" représente le 18 décembre 2019,
- une heure de début et une heure de fin, qui sont des nombres entiers entre 8 et 20 (pour représenter des réunions se tenant entre 8h et 20h).

Une réunion en présentiel (`ReunionPresentiel`) est une sorte de réunion avec la particularité suivante :

- un nombre de participants (`nbParticipants`).

Une vidéo-conférence (`VideoConference`) est une sorte de réunion avec les particularités suivantes :

- un nombre de participants présents (`nbParticipantsPresentes`),
- un nombre de participants sur les sites distants (`nbParticipantsDistants`),
- un nombre de sites distants connectés (`nbSitesDistants`).

Question 1. Ecrivez en Java les entêtes et attributs des classes représentant les réunions. Pour la suite, vous supposerez qu'un constructeur sans paramètre et les accesseurs de type `get` et `set` existent pour tous les attributs de toutes ces classes. Vous ne les écrivez pas et vous supposez que les accesseurs en modification (`setXXX`) contrôlent les valeurs.

Question 2. Ecrivez en Java, dans les classes `Reunion` et `VideoConference` uniquement, un constructeur avec paramètres permettant d'initialiser les attributs propres aux instances. Indiquez bien dans quelle classe vous écrivez quel constructeur.

Question 3. Ecrivez en Java, dans la classe `Reunion`, une méthode `duree`, qui retourne la durée d'une réunion. Vous l'obtiendrez en soustrayant l'heure de début à l'heure de fin. Par exemple, si la réunion commence à 10h et termine à 13h, la durée est 3h.

Question 4. Ecrivez en Java, dans les classes `Reunion` et `VideoConference` uniquement, une méthode `nbPlacesNecessaires` qui retourne le nombre de places nécessaires. Ce nombre de places nécessaires n'est pas défini pour les réunions en général. Pour les vidéoconférences, il s'agit du nombre de participants présents.

Question 5. Ecrivez en Java, pour les classes `Reunion` et `VideoConference` uniquement, une méthode `toString`, retournant une chaîne de caractères représentant l'objet. Indiquez bien dans quelle classe vous écrivez quelle méthode.

- Pour toutes les réunions, cette chaîne contient : la date (gardez le format AAAAMMJJ), l'heure de début, l'heure de fin, la durée, le sujet, et le nombre de places nécessaires.
- Pour les vidéo-conférences, cette chaîne contient comme ci-dessus la date (gardez le format AAAAMMJJ), l'heure de début, l'heure de fin, la durée, le sujet, le nombre de places nécessaires, et on ajoute en plus le nombre de sites distants.

3 Représentation des salles de réunion

Une salle de réunion (`SalleReunion`) a une adresse, un numéro, une capacité (nombre de personnes qu'elle peut accueillir), et une liste de réunions déjà accueillies, en cours ou à venir (réunions du passé, du présent, du futur) .

Question 6. Ecrivez en Java l'entête et les attributs de la classe `SalleReunion` en les initialisant.

Question 7. Ecrivez en Java, dans la classe `SalleReunion`, une méthode `ajout` permettant de lui ajouter une réunion sans effectuer de vérification.

Question 8. Ecrivez en Java, dans la classe `SalleReunion`, une méthode `dureeOccupation` retournant la somme des durées d'occupation des réunions organisées (passé, présent, futur) dans la salle.

Question 9. Ecrivez en Java, dans la classe `SalleReunion`, une méthode `recherche` retournant la liste des réunions organisées (passé, présent, futur) dans cette salle et traitant d'un sujet passé en paramètre à la méthode.

4 Approfondissement (sur 4 points maximum du barème final)

Question 10. Ecrivez en Java, dans la classe `Reunion`, une méthode `static boolean intervalleCorrect(int hd, int hf)` qui retourne vrai si l'heure de fin est bien postérieure à l'heure de début, et si `hd` et `hf` sont bien compris entre 8 et 20.

Question 11. Ecrivez en Java, dans la classe `Reunion`, une méthode `static boolean NeSeChevauchentPas(int hd1, int hf1, int hd2, int hf2)` qui retourne vrai si les deux intervalles ne se chevauchent pas. Pour cela, vous vérifierez que `[hd1,hf1]` est entièrement avant `[hd2,hf2]` (ce qui se produit lorsque `hf1 <= hd2`) ou l'inverse (ce qui se produit lorsque `hf2 <= hd1`).

Question 12. Ecrivez en Java, dans la classe `SalleReunion`, une méthode `boolean salleLibre(String date, int hd, int hf)` retournant vrai si la salle est (ou était) libre à la `date`, entre `hd` et `hf`. Il en existe une si aucune des réunions organisées dans la salle ce jour-là n'a des horaires qui chevauchent l'intervalle `[hd, hf]`.

Question 13. Ré-écrivez en Java, dans la classe `SalleReunion`, la méthode `ajout` permettant de lui ajouter une réunion, en ajoutant les conditions selon lesquelles : (1) la capacité de la salle est suffisante, (2) la salle est libre à ce moment-là.