

Evaluation 2 - partie pratique

La salle de spectacle

Travail à accomplir (temps à disposition : 120 minutes)

Vous travaillez pour l'entreprise *EMFSpect SA* basée à Fribourg. Cette société gère une salle de spectacle, les activités qui s'y déroulent et s'occupe de la vente des billets pour les différents événements.

La vente des billets se fait uniquement auprès de la caisse de la salle et est gérée par une application Java.

Ce programme permet de voir l'état d'occupation de la salle, d'afficher les tarifs appliqués pour les 3 catégories de billets disponibles et de commander un billet. La vente des billets se fait uniquement à l'unité (1 commande = 1 billet).

On vous demande de réaliser un programme Java permettant la gestion de la billetterie de cette salle.



Attention : pour les besoins de l'exercice, la salle de spectacle contiendra une unique rangée de 10 sièges.

Deux tableaux permettront de gérer les catégories/tarifs des sièges et l'occupation de la salle. Dans le premier tableau, qui est une constante, on trouvera la définition des différentes catégories de sièges. Dans la configuration actuelle, 1 paire de sièges de catégorie 3 sera disposée à chaque bout de la rangée. On trouvera ensuite 1 paire de sièges de catégorie 2, plus au centre, de chaque côté de la rangée et enfin, 2 sièges de catégorie 1 au centre de la rangée.

Dans le second tableau, on trouvera l'occupation des sièges. Il s'agit d'un tableau de valeurs booléennes où un **false** représente un siège libre et un **true** représente un siège occupé.

Les deux tableaux ont la même taille. Pour un même index, on trouvera la catégorie de la place dans le premier tableau et son occupation (**true/false**) dans le second.

Exemple :

Tableaux des catégories (valeurs constantes)

3	3	2	2	1	1	2	2	3	3
---	---	---	---	---	---	---	---	---	---

Tableaux d'occupation de la salle

O	O	O	X	O	X	X	X	X	X
---	---	---	---	---	---	---	---	---	---

Dans cet exemple, les sièges 0, 1, 2 et 4 sont occupés tandis que les autres sont disponibles. Les sièges occupés ont respectivement les catégories 3, 3, 2 et 1. Sur la console du programme, on affichera un « O » pour une place occupée et un « X » pour une place libre.

Fonctionnement : au démarrage, le programme demande à l'utilisateur de saisir une commande ; il s'agit d'une valeur numérique entre 0 et 3 et exécutera un traitement en particulier. Voici à quoi correspondent les commandes :

- 0 → Quitter le programme.
- 1 → Commander un billet. L'utilisateur devra préciser la catégorie désirée.
- 2 → Afficher les tarifs des différentes catégories de billets.
- 3 → Affichez l'état d'occupation de la salle.

Etapes à réaliser

1. Créez un nouveau projet Java dans NetBeans que vous nommerez **403_E2_NOM** (exemple 403_E2_GALLEY).
2. Créez les constantes **PRIX_CATEGORIE_UN**, **PRIX_CATEGORIE_DEUX** et **PRIX_CATEGORIE_TROIS** qui contiendront des entiers avec, respectivement, les valeurs suivantes : 25, 18 et 12.
3. Créez le tableau constant **CATEGORIE_SALLE** qui contiendra des entiers avec les valeurs suivantes : 3, 3, 2, 2, 1, 1, 2, 2, 3 et 3.
4. Créez une méthode nommée **afficherTarifs**. Cette méthode sera uniquement responsable d'afficher les différents tarifs appliqués dans la salle de spectacle (voir exemple d'affichage à la console).
5. Créez une méthode nommée **afficherSalle**. Cette méthode prendra en paramètre un tableau de booléens nommé **occupationSalle** qui représentera le fait que les différents sièges soient occupés ou non et qui ne retournera rien.
 - 5.1. Affichez le texte « Occupation de la salle ».
 - 5.2. Sur la ligne suivante, affichez le numéro des catégories contenu dans la constante **CATEGORIE_SALLE**. N'oubliez pas le retour à la ligne lorsque tous les numéros de catégories seront affichés !
 - 5.3. Sur la ligne suivante, affichez le fait qu'une place soit occupée ou non. La méthode affichera un X pour les places libres et un O pour les places occupées (voir exemple d'affichage à la console).
6. Créez une méthode **trouverPlace**. Cette méthode prendra en paramètre un entier nommé **categorie** qui représentera la catégorie du billet souhaitée par l'utilisateur et un tableau de booléens nommé **occupationSalle** qui représentera l'occupation actuelle de la salle de spectacle. Le but de cette méthode est de retourner la première position libre dans le tableau **occupationSalle** (cellule dont la valeur est *false*) pour la catégorie souhaitée. Pour rappel : le même index peut désigner un emplacement dans la salle (contenu dans le tableau de la variable **occupationSalle**) ET sa catégorie (contenu dans le tableau de la constante **CATEGORIE_SALLE**). Si aucune place n'est trouvée, la méthode retournera -1.
7. Créez une méthode **commanderBillet**. Cette méthode prendra en paramètre un tableau de booléens nommé **occupationSalle** qui représentera l'occupation des sièges (*false* = siège libre, *true* = siège occupé). Le but de cette méthode est de retourner un tableau de booléens représentant le nouvel état d'occupation de la salle une fois qu'un billet sera commandé. Voici les étapes afin de réaliser cela :
 - 7.1. Affichez sur la console le texte « Quelle catégorie voulez-vous : ».
 - 7.2. Lisez la valeur numérique que l'utilisateur va entrer au clavier sur la console et stockez cette valeur dans la variable **categorie**. Les catégories que l'utilisateur peut entrer sont 1, 2 ou 3. Pour rappel, voici le code nécessaire pour lire une valeur depuis la console :

```
// N'oubliez pas d'importer la classe Scanner avec l'instruction « import
// java.util.Scanner » au début du fichier, juste au-dessous du nom de package!
Scanner scanner = new Scanner(System.in);
int input = scanner.nextInt();
```

- 7.3. Si la valeur de la variable **categorie** est correcte, alors :
 - 7.3.1. Appelez la méthode **trouverPlace** et stockez le résultat dans la variable **positionPlace**.
 - 7.3.2. Si la valeur de la variable **positionPlace** est différente de -1, alors :
 - 7.3.2.1. Définissez la place à la position **positionPlace** dans le tableau **occupationSalle** comme étant occupée (avec la valeur *true*).

7.3.2.2. A l'aide d'un *switch*, affichez le texte « Votre place est réservée et coûte » avec ensuite le prix du billet puis « CHF » pour indiquer le prix en francs en fonction de la valeur de la variable `categorie` (voir exemple d'affichage à la console).

7.3.3. Sinon, affichez sur la console le texte « Aucune place disponible ».

7.4. Sinon, affichez sur la console le texte « Cette catégorie n'existe pas ».

7.5. Retourner le tableau `occupationSalle`.

8. Dans la méthode *main*, implémentez les actions suivantes :

8.1. Déclarez un tableau de booléens nommé `occupationSalle` et initialisez-le avec la taille du tableau contenu dans la constante `CATEGORIE_SALLE`.

8.2. Déclarez la variable entière `commande` et initialisez-la avec la valeur -1.

8.3. Tant que la valeur de la variable `commande` n'est pas égale à 0, effectuez le traitement suivant :

8.3.1. Affichez une ligne séparatrice contenant des tirets « ---- ... » (Environ 50 tirets).

8.3.2. Affichez une ligne contenant les commandes disponibles dans l'application « 1 = Commander un billet, 2 = Afficher les tarifs, 3 = Afficher l'état de la salle, 0 = Quitter ».

8.3.3. Lisez la valeur numérique que l'utilisateur va entrer au clavier sur la console et stockez cette valeur dans la variable `commande`. Les commandes que l'utilisateur peut saisir sont 0, 1, 2 ou 3. Pour rappel, voici le code nécessaire pour lire une valeur depuis la console :

```
// N'oubliez pas d'importer la classe Scanner avec l'instruction « import
// java.util.Scanner » au début du fichier, juste au-dessous du nom de package!
Scanner scanner = new Scanner(System.in);
int input = scanner.nextInt();
```

8.3.4. A l'aide d'un *switch*, déterminez le traitement à réaliser en fonction de la variable `commande` :

8.3.4.1. Si la variable est égale à 0, afficher le texte « Au revoir ».

8.3.4.2. Si la variable est égale à 1, appelez la méthode `commanderBillet` avec, en paramètre, la variable `occupationSalle` précédemment créée puis stockez le résultat de l'appel dans cette même variable.

8.3.4.3. Si la variable est égale à 2, appelez la méthode `afficherTarifs`.

8.3.4.4. Si la variable est égale à 3, appelez la méthode `afficherSalle`.

8.3.4.5. Dans le cas où la commande n'est pas connue, affichez le texte « Commande inconnue ».

Exemple de résultat à la console :

```
-----
1 = Commander un billet, 2 = Afficher les tarifs, 3 = Afficher l'état de la salle, 0 = Quitter
Quelle opération voulez-vous faire : 2
Catégorie 1 : 25 CHF
Catégorie 2 : 18 CHF
Catégorie 3 : 12 CHF
-----
1 = Commander un billet, 2 = Afficher les tarifs, 3 = Afficher l'état de la salle, 0 = Quitter
Quelle opération voulez-vous faire : 3
Occupation de la salle :
3 3 2 2 1 1 2 2 3 3
X X X X X X X X X
-----
1 = Commander un billet, 2 = Afficher les tarifs, 3 = Afficher l'état de la salle, 0 = Quitter
Quelle opération voulez-vous faire : 1
Quelle catégorie voulez-vous : 1
Votre place est réservée et coûte 25 CHF
-----
1 = Commander un billet, 2 = Afficher les tarifs, 3 = Afficher l'état de la salle, 0 = Quitter
Quelle opération voulez-vous faire : 1
Quelle catégorie voulez-vous : 1
Votre place est réservée et coûte 25 CHF
-----
1 = Commander un billet, 2 = Afficher les tarifs, 3 = Afficher l'état de la salle, 0 = Quitter
Quelle opération voulez-vous faire : 1
Quelle catégorie voulez-vous : 3
Votre place est réservée et coûte 12 CHF
-----
1 = Commander un billet, 2 = Afficher les tarifs, 3 = Afficher l'état de la salle, 0 = Quitter
Quelle opération voulez-vous faire : 3
Occupation de la salle :
3 3 2 2 1 1 2 2 3 3
O X X X O X X X X
-----
1 = Commander un billet, 2 = Afficher les tarifs, 3 = Afficher l'état de la salle, 0 = Quitter
Quelle opération voulez-vous faire : 1
Quelle catégorie voulez-vous : 1
Aucune place disponible
-----
1 = Commander un billet, 2 = Afficher les tarifs, 3 = Afficher l'état de la salle, 0 = Quitter
Quelle opération voulez-vous faire : 0
Au revoir
```

Restitution

Lorsque vous avez terminé, faites-signé au professeur pour lui remettre votre travail.

N'oubliez pas de formater votre code !

Bon spectacle !

