TP Noté 2 – Carnet de rendez-vous

06/12/2022

*Durée : **2h00**. Documents autorisés. Faites une archive (tar.gz ou zip) ne contenant que les sources et déposez-le sur l'espace Moodle dans le devoir « TP Noté 06/12/22 ».*

1. Écrire une classe `horaire` représentant un horaire d'un rendez-vous. Cette classe devra mémoriser 3 attributs : l'heure (un entier positif), le jour du mois (un entier positif) et le mois (un entier positif). Munir cette classe des fonctionnalités suivantes :
   Constructeur prenant comme paramètres heure, jour et mois[1]. Pas de constructeur par défaut.
   Trois accesseurs appelés `heure`, `jour` et `mois` pour accéder aux attributs et un mutateur à l'heure.
   Une méthode `tostring` retournant une chaîne de la forme « 6/12 - 8h » (*jour***/***mois* **-** *heure***h**).
   Faire en sorte qu'on puisse comparer deux horaires avec les opérateurs ==, !=, < et <=[2].
   Un opérateur de sortie sur flux qui envoie sur le flux l'horaire sous la même forme que `tostring`.
2. Définir un type énuméré `unite` pouvant prendre trois valeurs possibles POO, `Archi`, DCRA.
   Écrire une fonction `unite_tostring` prenant comme argument une matière et retournant une chaîne contenant l'intitulé de la matière.
3. On veut maintenant représenter des rendez-vous. Pour chaque rendez-vous, on mémorisera l'horaire de début. Il y a différents types de rendez-vous : les créneaux d'enseignements, les rendez-vous médicaux, et les vacances.
   Dans le cas d'un créneau d'enseignement, on mémorisera la matière (sous la forme d'une `unite`), et pour un rendez-vous médical le nom du médecin (une chaîne).
   Un rendez-vous a une durée, elle est de 80 minutes pour un créneau d'enseignement, sauf pour les examens qui sont des créneaux d'enseignement particuliers de longueur variable (choisie à la construction). Un rendez-vous médical dure 60 mn, et les vacances ont une durée variable (choisie à la construction).
   Définir ces classes et fournir les accesseurs à l'horaire de début, à la durée, à la matière, et au médecin (évidemment certains de ces accesseurs ne sont pas accessibles dans toutes les classes).
4. Écrire des constructeurs pour chacune de ces classes.
5. Pour construire des instances, une vérification des valeurs devra être effectuée pour certains types de rendez-vous. Dans tous les cas, la durée ajoutée à l'horaire de début ne doit pas dépasser minuit[3]. Dans le cas d'un créneau d'un examen, l'heure de l'horaire devra être comprise entre 8 et 16. Dans le cas d'un rendez-vous vacances, la durée devra être supérieure ou égale à 60. Faire en sorte qu'il soit impossible de créer d'instances ne respectant pas ces conditions en fournissant une/des méthode(s) retournant une nouvelle instance, allouée dynamiquement, uniquement si les conditions sont respectées.
6. Écrire une méthode `decaler` prenant un entier signé et effectuant un déplacement d'un rendez-vous du nombre d'heures passé en argument[4]. La méthode retournera un booléen représentant si le décalage a été accepté. Un décalage qui ferait changer le jour est interdit[5]. Un décalage d'un enseignement ne peut être fait que de plus ou moins une heure. Un décalage de rendez-vous médical est interdit, alors qu'un décalage de vacances est toujours possible[6].
7. Écrire un opérateur de sortie sur flux de rendez-vous sortant la totalité des informations sous la forme : *type de rendez-vous – horaire de début – durée – informations supplémentaires : matière ou médecin*, par exemple « *Examen - 6/12 - 8h – 120mn - POO* », « *Enseignement – 5/12 – 12h – 80mn - POO* » ou « *Vacances – 8/12 – 8h – 240 mn* ».

1 Aucune vérification ne sera effectuée sur le jour, le mois ou l'heure.

2 Ces deux derniers opérateurs feront une comparaison classique représentant un horaire qui est avant l'autre (sur une même année). Par exemple le 20 Février à 15h est avant le 2 Mars à 18h.

3 En d'autres termes, un rendez-vous ne doit pas être « à cheval » sur 2 jours.

4 Par exemple, un décalage sur -1 rendez-vous de 15h le fera passer à 14h, et un décalage de 3 le fera passer à 18h.

5 En d'autres termes, on interdira un décalage qui conduira à une heure négative, ou une heure supérieure ou égale à 24.

6 À condition de respecter la propriété citée plus haut pour tous les types de rendez-vous : pas de changement de jour.

8. Définir une classe `carnet` composée d'un ensemble de rendez-vous. Munir cette classe d'un constructeur par défaut construisant un carnet vide, sans rendez-vous.

9. Écrire une classe `exception_carnet` sous-classe de `std::exception` représentant une erreur dans une opération sur un carnet (voir question suivante). Le constructeur de cette classe prendra comme argument une chaîne et la méthode `what` retournera cette chaîne.

10. Écrire dans `carnet` une méthode `ajouter` permettant de rajouter un rendez-vous. Si un rendez-vous avec le même horaire de départ existe, une `exception_carnet` sera levée avec le message du type « *Horaire 6/12 – 8h déjà utilisé* ».

11. Écrire une méthode `examens_apres` de `carnet` prenant comme argument un `horaire` et retournant tous les examens qui ont lieu après cet horaire.

12. Écrire une méthode `supprimer_matiere` prenant comme argument une `matiere` et supprimant d'un `carnet` tous les enseignements de cette matière.

13. Écrire dans carnet une méthode `saisie_medical` demandant à l'utilisateur de saisir un horaire et un nom de médecin, et ajoutant (par appel à `ajouter`) un rendez-vous médical correspondant à cet horaire et ce médecin. Si la méthode `ajouter` refuse de faire l'ajout, une message d'erreur sera affiché et l'utilisateur sera invité à saisir une autre **heure** (le jour, le mois et le nom du médecin ne seront pas demandés à nouveau). Cette demande sera répétée tant que la méthode `ajouter` refusera l'ajout.

14. Écrire une classe `qt_decaler` dont le constructeur prendra comme argument un rendez-vous. Cette classe affichera dans une fenêtre l'horaire et la durée (dans deux widgets différents). Elle disposera aussi d'un champ de texte éditable permettant à l'utilisateur de saisir un entier signé, et d'un bouton « *Décaler* ». Quand l'utilisateur cliquera sur le bouton, cela provoquera un appel à la méthode `decaler` sur le rendez-vous avec l'entier saisi par l'utilisateur. Si le décalage n'est pas possible un `QMessageBox` affichant « Décalage impossible » sera affiché, sinon, la fenêtre se fermera.

Barème (donné à titre indicatif)

q	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**main**
pts	2,5	0,5	3	0,5	2,5	2,5	1,5	0,5	1	2	2	2	2	3	0,5
														Total	**15**