OFPPT

مكتب التكوين المهني وإنعاش الشغل

Office de la Formation Professionnelle
et de la Promotion du Travail

Examen de Passage

Session Juin 2007

Filière : TSDI

Epreuve : Pratique

Niveau : Technicien Spécialisé

Durée : 4 h 30

Barème : 40 Pts

Variante n° 6

Partie 1 : JAVA (16 pts)

1. Créer deux classes principales BankLocal et BankLocalServer qui vont agir respectivement comme client et comme serveur (en local). Seule BankLocal sera déclarée publique. **(2 Points)**
2. Créer une classe Account, représentant un compte bancaire, munie des champs String password, le mot de passe associé au compte et int balance le solde du compte. Cette classe ne contiendra qu'un constructeur et aucune autre méthode. Ce constructeur ne sera chargé que d'enregistrer le mot de passe et d'initialiser le solde à zéro.**(2 Points)**
3. L'ensemble des comptes bancaires sera stocké dans une table de hachage (Hashtable) réalisant l'association nom du titulaire d'un compte – objet de type Account (le compte, informatiquement parlant). Cette table sera le seul champ de BankLocalServer, nommé accounts. **(3 Points)**
4. Créer une exception BankingException qui hérite d'exception et qui ne définit qu'un seul constructeur (et aucune autre méthode) BankingException(String) qui appelle le constructeur de la classe mère. Cette exception servira pour des cas tels que "Solde insuffisant" ou "Mot de passe invalide". **(3 Points)**
5. Prévoir les opérations suivantes : **(4 Points)**

   – Ouverture de compte par la méthode
   public void openAccount(String name, String password) throws BankingException.

Le nom du titulaire name est donné en premier argument et son mot de passe password en deuxième argument. La méthode réalise la vérification d'un compte ayant déjà le même nom de titulaire, auquel cas elle lève une exception de type BankingException et dans le cas contraire crée le compte et l'enregistre dans la table accounts.
– Une méthode utilitaire de vérification d'existence de compte et de mot de passe valide : public Account verify(String name, String password) throws BankingException.
Si le compte de nom name n'existe pas (dans account) ou si le mot de passe password est différent de celui trouvé dans accounts comme correspondant à name, la méthode génère une exception de type BankingException.
Sinon, elle renvoie une référence sur l'objet de type Account correspondant.

- Fermeture de compte par la méthode
  public int closeAccount(String name, String password) throws BankingException.
  Elle effectue une vérification à l'aide de verify(...), retire le compte de accounts, met le solde à zéro et renvoie le montant disponible.
- Dépôt d'argent sur un compte par la méthode
  public void deposit(String name, String password,
  int money) throws BankingException.
  Elle effectue une vérification à l'aide de verify(...) et incrémente le solde du montant money déposé.
- Retrait d'argent sur un compte par la méthode
  public int withdraw(String name, String password,
  int amount) throws BankingException.
  Elle effectue une vérification à l'aide de verify(...), vérifie que le solde est suffisant pour le montant du retrait (dans le cas contraire, elle genère une exception de type BankingException), décrémente le solde du montant amount et renvoie le montant retiré.
- Obtention du solde d'un compte par la méthode
  public int getBalance(String name, String password) throws BankingException.
  Elle effectue une vérification à l'aide de verify(...), puis renvoie le solde du compte.

6. La méthode main(...) se trouvera dans BankLocal et proposera à l'utilisateur un menu textuel afin qu'il puisse gérer son compte. Les entrées seront effectuées avec des readLine() (sur une référence créée comme suit BufferedReader stdin = new BufferedReader(new InputStreamReader(System.in)) ;) et les sorties seront effectuées avec des System.out.println(...) **(2 Points)**

Partie 2 : Programmation événementielle (17 pts)

Le service après vente d'un grand magasin souhaite informatiser son système de gestion des interventions réalisées. Les réparations du SAV ne concernent que le matériel électronique (le plus sujet à des pannes).

On est alors en mesure d'enregistrer une intervention, identifiée par un numéro d'intervention, la date de l'intervention, la durée de l'intervention (en heures) ainsi que le type d'intervention (ex : magasin, déplacement à domicile, etc.) et le technicien (nom et prénom) concerné par l'intervention.

Questions :

1- Créer l'interface suivante :

(1 point)

NB : la zone de texte qui concerne le numéro d'intervention doit être verrouiller et incrémenter par 1 à chaque nouvelle intervention.

2- Créer la classe intervention et la collection qui va contenir toutes les interventions.
(2 points)

3- Ecrire une méthode de contrôle de saisie **(1 point)**

4- Ecrire les fonctions qui correspondent aux différents boutons de l'interface :
   a. Premier **(1 point)**
   b. Précédant **(1 point)**
   c. Suivant **(1 point)**
   d. Dernier **(1 point)**

e. Nouveau **(1 point)**
f. Enregistrer **(1.5 point)**
g. Supprimer **(1.5 point)**

5- Ajouter une liste affichant l'ensemble des interventions triées par type d'intervention **(2 points)**
6- Sachant qu'une heure d'intervention est facturée à 165 dhs. Ecrire un programme permettant de calculer le montant des interventions entre deux dates saisies par l'utilisateur et afficher le résultat dans une zone de texte verrouillée. **(3 Points)**

Partie 3 : Sql Server (7 pts)

Soit la structure de la base de données suivante :

ANIMAL (NoAnimal, NomAnimal, NoEmplacement, CodeEspèce)
EMPLACEMENT (NoEmplacement, TypeEmplacement)
GARDIEN (NoGardien, NomGardien, NoChef, Salaire)
AFFECTATION (NoEmplacement, NoGardien)
ESPECE (CodeEspèce, NomEspèce, Classe)

Les animaux sont parqués dans des emplacements. Ces emplacements sont affectés à un ou plusieurs gardiens. Certains gardiens dirigent d'autres gardiens. NoChef est le numéro du chef d'un gardien. Cette information a la valeur null si le gardien n'a pas de chef.

A – Créer la base de données **Zoo** en respectant le **MLD** ci-dessus **(2 pts)**

B – les requêtes SQL (5 pts) :

1 -Afficher les noms d'espèces et classes des animaux résidant aux emplacements de type bassin ou aquarium. **(1 pt)**

2 - Le zoo décide de supprimer les différentes cages à oiseaux (typeEmplacement = 'CAGE_A_OISEAUX'). Les volatiles de ce type d'emplacements seront indistinctement relogés à l'emplacement numéro 7, car c'est celui de la grande volière. **(1 pt)**
Ne pas oublier de supprimer les affectations à ces cages.

3-afficher les noms des gardiens qui s'occupent d'un emplacement dont leur chef ne s'occupe pas. **(1 pt)**

4- Afficher tous les numéros d'emplacements avec, pour chaque emplacement, le nombre de gardiens affectés à cet emplacement. **(1pt)**

5- Afficher les noms des espèces dont s'occupe le gardien nommé Dupond. On dira qu'un gardien s'occupe d'une espèce si ce gardien est affecté à un emplacement occupé par un animal de cette espèce. **(1 pt)**