CONTRÔLE FINAL
MASTER II - SRI

Questions de cours (5 pts)

1. A quoi sert l'encapsulation dans les systèmes de gestion de bases de données orientées objets ? (1pt)
2. Quelle est la différence entre classe et type ? (1pt)
3. Quelques SGBDO acceptent la contrainte inverse. Définissez cette contrainte. (1pt)
4. L'héritage multiple est la cause d'un conflit dans les SGBDO. Quelles solutions proposent les SGBDO pour résoudre ce conflit ? (1pt)
5. Donnez le résultat de cette requête OQL : (1pt)

```
GROUP x IN (    SELECT e FROM e IN Enseignants
                WHERE FOR ALL c IN e.cours-assurés : c.cycle=3
           )
BY ( statut : x.statut )
WITH ( nbx : COUNT(partition) ) ;.
```

Requêtes OQL (5 pts)

Dans un système GPS, une classe appelée « Distance » est définie comme suit :

```
Class Distance {
    Dedut: String; //Ville début
    Fin: String; //Ville fin
    Km: Integer; //Distance entre les deux villes en Km.
}
```

Donnez les requêtes OQL qui répondent aux questions suivantes :

1. L'ensemble des villes de fin possibles. (1pt)
2. Les distances qui ont la ville de début « Batna ». (1pt)
3. Les villes de fin dans les distances qui ont la ville début Batna et une distance de 500 Km au plus. (1pt)
4. L'ensemble des villes fin, de toutes les distances, qui ne sont pas des villes débuts pour d'autres distances. (2pts)

Modélisation (10 pts)

Un système de gestion d'un tramway conserve les informations suivantes concernant les stations, les trains et les voies.

- Un train est caractérisé par un identifiant, un nom, une capacité et un état (en service ou hors service).
- Chaque station est caractérisée par un identifiant, un nom, une localisation (adresse composée elle-même du nom de la rue, du code postal et du nom de la ville) et un état (en service ou hors service), identifiant_train (0 si aucun train dans la station).
- Une ligne est caractérisée par un identifiant, une « station début », une « station fin », une liste de stations sur cette ligne et une liste de trains circulant sur cette ligne.

1. Ecrivez en java les classes nécessaires pour modéliser la gestion du tramway. La classe principale sera nommée « **GEST_TRAM** ».
2. Dans le constructeur de la classe **GEST_TRAM** créez 10 trains et 2 lignes de 5 stations chacune, tout en précisant les stations début et fin de chaque ligne ainsi que la liste des stations et la liste des trains circulant sur cette ligne.
   Les trains auront des noms différents, de capacité de 100 passagers et leurs états seront « en service ».
   Les stations aussi auront des noms différents, des localisations quelconques et leurs états seront « en service ».
   Tout sera stocké dans une base de données orientée objet.
3. Ecrivez la méthode **getListeTrains()** qui retourne la liste de trains circulants sur une ligne.
4. Ecrivez la méthode **getListeStations()** qui retourne la liste des stations sur une ligne.
5. Pour garder les traces de déplacement des trains, écrivez les méthodes **arriveeTrain()** et **sortieTrain()** qui traitent l'arrivée et la sortie d'un train à une station et met à jour la base de données.
6. Supposant que la base de données est remplie. Ecrivez la méthode **main()** de la classe **GEST_TRAM** qui affiche les noms des trains en service circulant sur la ligne n°1, affiche les noms des stations en services sur la ligne n°2, fait entrer le train n°5 à la station n°8 et fait sortir le train n°1 de la station n°4.

Bonne chance…

NB: Le corrigé type vous le trouverez sur le site :
http://www.larbiguezouli.com

CORRECTION DU CONTRÔLE FINAL
MASTER II - SRI

Questions de cours (5 pts)

1. A quoi sert l'encapsulation dans les systèmes de gestion de bases de données orientées objets ? (1pt)
   L'encapsulation permet de cacher l'implantation d'un objet (valeurs) et l'utilisateur ne connait rien sur les constituantes internes des données contenues dans cet objet.
2. Quelle est la différence entre classe et type ? (1pt)
   Une classe d'objets permet de regrouper des objets possédant les mêmes caractéristiques.
   Un type permet de décrire la structure de données.
3. Quelques SGBDO acceptent la contrainte inverse. Définissez cette contrainte. (1pt)
   L'objet composite dépend de ses objets composants.
4. L'héritage multiple est la cause d'un conflit dans les SGBDO. Quelles solutions proposent les SGBDO pour résoudre ce conflit ? (1pt)
   - **soit ils refusent la définition d'un tel héritage;**
   - **soit ils demandent au concepteur de choisir la sur-classe «dominante» dont la sous-classe héritera;**
   - **soit ils appliquent une règle (par exemple la première classe citée dans la clause Is-a).**
5. Donnez le résultat de cette requête OQL : (1pt)

```
GROUP x IN (     SELECT e FROM e IN Enseignants
                 WHERE FOR ALL c IN e.cours-assurés : c.cycle=3
           )
BY ( statut : x.statut )
WITH ( nbx : COUNT(partition) );
```

   Cette requête donne en résultat un ensemble de structures:
   SET (STRUCT(statut:STRING, nbx:INT))
   qui définissent pour chaque enseignant (parmi les enseignants qui ne donnent que des cours de 3ème cycle et ceux qui ne donnent aucun cours) son statut et le nombre d'enseignants de chaque groupe.

Requêtes OQL (5 pts)

Dans un système GPS une classe appelée « Distance » est définie comme suit :

```
Class Distance {
    Dedut : String ; //Ville début
    Fin : String ; //Ville fin
    Km : Integer ; //Distance entre les deux villes en Km.
}
```

Donnez les requêtes OQL qui répondent aux questions suivantes :

1. L'ensemble des villes de fin possibles. (1pt)

```
SELECT d.Fin
FROM d IN Distance
```

2. Les distances qui ont la ville de début « Batna » (1pt)

```
SELECT d
```

```
FROM d IN Distance
WHERE d.Debut="Batna"
```

3. Les villes de fin dans les distances qui ont la ville début Batna et une distance de 500 Km au plus. (1pt)

```
SELECT d.Fin
FROM d IN Distance
WHERE d.Debut="Batna" AND d.Km<=500
```

4. L'ensemble des villes fin, de toutes les distances, qui ne sont pas des villes débuts pour d'autres distances. (2pts)

```
SELECT d.Fin
FROM d IN Distance
WHERE d.Fin NOT IN (SELECT x.Debut FROM x IN Distance)
```

Modélisation (10 pts)

Un système de gestion d'un tramway conserve les informations suivantes concernant les stations, les trains et les voies.

- Chaque station est caractérisée par un identifiant, un nom, une localisation (adresse composée elle-même du nom de la rue, du code postal et du nom de la ville) et un état (en service ou hors service), identifiant_train (0 si aucun train dans la station).
- Un train est caractérisé par un identifiant, un nom, une capacité et un état (en service ou hors service).
- Une ligne est caractérisée par un identifiant, une « station début », une « station fin », une liste de stations sur cette ligne et une liste de trains circulant sur cette ligne.

1. Ecrivez en java les classes nécessaires pour modéliser la gestion du tramway. La classe principale sera nommée « **GEST_TRAM** ». (2pts)
2. Dans le constructeur de la classe **GEST_TRAM** créez 10 trains et 2 lignes de 5 stations chacune, tout en précisant les stations début et fin de chaque ligne ainsi que la liste des stations et la liste des trains circulant sur cette ligne.
   Les trains auront des noms différents, de capacité de 100 passagers et leurs états seront « en service ».
   Les stations aussi auront des noms différents, des localisations quelconques et leurs états seront « en service ».
   Tout sera stocké dans une base de données orientée objet. (2pts)
3. Ecrivez la méthode **getListeTrains()** qui retourne la liste de trains circulants sur une ligne. (1pt)
4. Ecrivez la méthode **getListeStations()** qui retourne la liste des stations sur une ligne. (1pt)
5. Pour garder les traces de déplacement des trains, écrivez les méthodes **arriveeTrain()** et **sortieTrain()** qui traitent l'arrivée et la sortie d'un train à une station et met à jour la base de données. (2pts)
6. Supposant que la base de données est remplie. Ecrivez la méthode **main()** de la classe **GEST_TRAM** qui affiche les noms des trains en service circulant sur la ligne n°1, affiche les noms des stations en services sur la ligne n°2, fait entrer le train n°5 à la station n°8 et fait sortir le train n°1 de la station n°4. (2pts)

Le code source de la partie modélisation peut être téléchargé à partir du lien suivant :
http://www.larbiguezouli.com/Fichiers/Enseignement/MasterSRI/bdoo/Controle_Final_Correction_2011_2012_Modelisation.zip