Examen

18 janvier 2011

6 pages – Durée 3h – Tous documents autorisés – Responsable : E. Waller
Si un point de l'énoncé vous semble personnellement non clair, consultez immédiatement un enseignant.

Cette page de garde de l'examen est exactement celle qui a été fournie avant l'examen. Si vous l'avez déjà lue, il est inutile de la relire et vous pouvez passer directement à la page suivante.

Pour des raisons d'organisation des copies uniquement, les exercices sont organisés dans des parties. Chaque exercice dépend du précédent, et il est conseillé de les faire dans l'ordre (sauf le 4 qui est indépendant). Mais il est parfois possible de faire certaines parties d'un exercice même si on n'a pas fini tous les exercices précédents. Chaque exercice doit être rédigé séparément : il n'est pas possible de donner un seul code contenant plusieurs exercices.
Remarque : L'énoncé a été conçu pour qu'il soit possible de programmer les exercices dans l'ordre de façon à ce que chaque nouvel exercice ne fasse que rajouter du code sans modifier du code existant (sauf parfois dans la saisie), mais il n'est pas obligatoire de procéder ainsi.

Le barème est indicatif. Pour chaque exercice il correspond à la fois à la difficulté et/ou au temps nécessaire pour faire cet exercice. *Rappel :* Comme cela a été dit en cours, soit un exercice est correct et il comptabilisera la totalité des points, soit il n'est pas correct et il ne comptabilisera presque rien. Vous vous attacherez donc à vérifier soigneusement les programmes que vous écrirez (un programme par exercice), afin qu'ils fonctionnent tels quels, et à suivre les principes de qualité vus en cours et TD. Les ratures sur les copies ne seront pas pénalisées. Il est donc possible de ne pas utiliser de brouillon, tout faire directement sur la copie, puis si nécessaire raturer et recommencer. Ne recopiez pas inutilement du code (indiquez-le plutôt par exemple en numérotant les lignes puis en indiquant les numéros).

Important: Utilisez à chaque fois que c'est pertinent les concepts vus dans le module.

Organisation des copies *obligatoire*. Vous devez changer de copie à chaque partie. Les exercices doivent apparaître dans l'ordre sur la copie, *tous* les numéros doivent apparaître, y compris jusqu'à la fin de chaque partie, et *toutes* les parties doivent apparaître. Si vous ne faites pas un exercice tout de suite, vous pouvez laisser un espace et y revenir ensuite (si alors vous n'avez pas assez de place, indiquez que vous poursuivez l'exercice en fin de copie). Après un exercice, tirez un grand trait horizontal sur toute la largeur de la page, éventuellement sans règle, tournez la page, et soulignez le titre de l'exercice suivant sur la nouvelle page (ex : Exercice 2). Un point sera attribué si cette organisation est *rigoureusement* respectée.

On donne ci-dessous le cahier des charges, puis ensuite les exercices demandés. Toutes les informations nécessaires sont indiquées dans ce cahier des charges. Tous les traitements de ce cahier des charges seront demandés dans un exercice. Pour chaque exercice vous appellerez votre programme avec toujours tous les paramètres, comme indiqué à la fin du cahier des charges, même si parfois ils ne sont pas tous utilisés.

Cahier des charges : Bibliothèque : emprunts, livres, clients divers

Données On considère des clients, des livres et des emprunts. Un livre a un titre et un auteur.

Il y a deux sortes de clients. Des clients simples avec un nom, un âge (entier), une durée d'emprunt et un nombre maximum de livres par emprunt qui sont les mêmes pour tous, et le montant de leur inscription qui est la moitié de leur âge.

Des clients privilégiés, avec un nom, un âge, une durée d'emprunt qui est celle des clients simples mais augmentée d'une "rallonge", le nombre maximum de livres par emprunts ci-dessus, et le montant de leur inscription qui est la moitié de leur âge plus la rallonge (ils paient un peu plus pour pouvoir garder les livres un peu plus).

Un emprunt a un numéro, une date de début (un entier entre 1 et 365), un client (simple ou privilégié), l'ensemble des livres empruntés par le client ce jour-là, et une durée. Cette durée est la durée d'emprunt du client, et le nombre maximal de livres par emprunt est celui de la bibliothèque. On doit pouvoir accéder, à partir d'un emprunt, à son client et à ses livres, mais on n'a jamais besoin d'accéder aux emprunts à partir des clients, ni à partir des livres.

Restrictions On a les contraintes suivantes. Tout emprunt a un client et un ensemble de livres (au moins un). Il peut y avoir des clients n'apparaissant dans aucun emprunt, et de même pour les livres. Un client peut apparaître dans deux emprunts. Un livre n'est jamais commun à deux emprunts. On doit pouvoir accéder aux clients sans passer par les emprunts, et de même pour les livres.

Pour simplifier on suppose ce qui suit. Tous les noms de client et titres de livres sont différents. Le nombre maximal de clients, de livres et d'emprunts est fixé à la saisie (sur la ligne de commande) Enfin, on ne fera aucune vérification (dates, etc.). (Remarque : on distingue bien le nombre maximal de livres par emprunt, et le nombre maximal d'emprunts.)

Traitements considérés On veut pouvoir effectuer les traitements suivants.

1. Afficher toutes les informations de l'application : d'abord l'ensemble des livres, puis l'ensemble des clients, puis enfin l'ensemble des emprunts ; lors de l'affichage d'un emprunt, on n'affichera de son client que son nom, et de ses livres que leur titre.
2. Afficher le numéro de l'emprunt de durée la plus longue.
3. Afficher simplement, depuis le *main* et *sans fonctions*, tous les titres de livre apparaissant dans un emprunt.

Concernant la saisie, on procède comme suit (voir aussi l'appel Java ci-dessous). Pour simplifier, on fera l'hypothèse, mais *uniquement pour la saisie*, que chaque emprunt a exactement deux livres. *Attention :* Le reste de l'application doit pouvoir gérer un nombre quelconque de livres par emprunt. Les informations sont saisies sur la ligne de commande, avec les conventions suivantes.

D'abord un client simple (qui n'apparaît dans aucun emprunt), puis un livre (qui n'apparaît dans aucun emprunt), puis chaque emprunt avec son client et ses livres, puis le nombre maximal de livres par emprunt et la durée de base d'un emprunt, puis le nombre maximal de livres de la bibliothèque, le nombre maximal de clients, et le nombre maximal d'emprunts. Juste avant chaque client, un *0* si c'est un client simple, un *1* si c'est un client privilégié. Les informations non pertinentes sont remplacées par *-1*.

Exemple On considère l'exemple suivant.
La cliente *Lulu*, *23* ans, n'apparaît dans aucun emprunt.
Le livre *RueDesPetitesDaurades*, de *Fellag*, n'apparaît dans aucun emprunt.
L'emprunt numéro *11* regroupe *Riton*, *24* ans, et les livres *AssimilBrésilien* de *AyrtonSenna*, et *DesMusclesEnDixLeçons* de *Schwarzenegger*.
L'emprunt numéro *12* regroupe *Rita*, cliente privilégiée, *22* ans, rallonge de *15*, et les livres *PaysDeNeige* de *Kawabata*, et *LesPlanchesCourbes* de *Bonnefoy*.
Le nombre maximal de livres par emprunt est *4*, et la durée de base d'un emprunt est *21* jours.
Le nombre maximal de livres de la bibliothèque est *5000*, le nombre maximal de clients *200*, et le nombre maximal d'emprunts *1000*.

Attention : Quelle que soit la question, vous appellerez votre programme de la manière suivante *obligatoirement* (*attention à l'ordre : obligatoire*). Cet appel correspond à l'exemple ci-dessus. Mais bien sûr votre programme devra fonctionner quelles que soient les valeurs lues sur la ligne de commande. (Pour faciliter la compréhension, les paramètres de la ligne de commande ont été donnés sur plusieurs lignes, et alignés.)

```
java Examen Lulu 23 RueDesPetitesDaurades Fellag
11 0 Riton 24 -1 AssimilBrésilien AyrtonSenna DesMusclesEnDixLeçons Schwarzenegger
12 1 Rita 22 15  PaysDeNeige      Kawabata    LesPlanchesCourbes    Bonnefoy
4 21 5000 200 1000
```

Partie 1 (7 points)

Attention : rédigez sur une nouvelle copie et placez les exercices dans l'ordre.

Rappel : Dans tous les exercices de l'épreuve, vous devez utiliser l'appel Java complet donné dans le cahier des charges.

1. (4 points) Dans cet exercice, on considère uniquement des clients simples avec toutes leurs informations. Pour simplifier, on considère tous les clients passées sur la ligne de commande comme des clients simples (on ignore leurs autres spécificités), et on ignore les livres et les emprunts.

   Vous rangerez les clients simples dans un tableau (appelé *lesClients*).
   Saisissez au clavier les informations considérées.
   Programmez le traitement 1 avec uniquement les informations considérées.

   Pour des raisons de lisibilité des copies, donnez le *main* complet. Pour chaque nouvelle fonction, vous justifierez en une ligne pourquoi elle est statique ou dynamique.
   Il est interdit de créer le tableau en dehors d'une fonction. Rappel : Ne recopiez pas inutilement du code de l'exercice précédent.

   Voici par exemple ce qu'affiche le programme final des enseignants pour cet exercice pour l'appel ci-dessus.

   ```
   nb max livres par emprunt : 4
   Lulu, 23 ans, 11.5 euros, 21 jours
   Riton, 24 ans, 12.0 euros, 21 jours
   Rita, 22 ans, 11.0 euros, 21 jours
   ```

2. *Rappel : tirez un grand trait horizontal sur toute la largeur de la page, éventuellement sans règle, sautez deux lignes, et soulignez le titre de l'exercice. Faites de même pour tous les exercices.*

   (3 points) On reprend l'exercice 1, qu'on adapte. Dans cet exercice on considère, en plus des clients simples, uniquement les clients privilégiés.

   Est-il possible de ranger toutes les clients dans un unique tableau ? Si oui faites-le, sinon faites comme vous voulez. Dans tous les cas justifiez (deux lignes).

   Saisissez au clavier les informations considérées.
   Programmez le traitement 1 avec uniquement les informations considérées.

   Pour des raisons de lisibilité des copies, donnez le *main* complet. Pour chaque nouvelle fonction, vous justifierez en une ligne pourquoi elle est statique ou dynamique.

   Expliquez précisément le fonctionnement de tous les appels de fonction dans la fonction affichant tous les clients (5 à 6 lignes max).

   Voici par exemple ce qu'affiche le programme final des enseignants pour cet exercice pour l'appel ci-dessus.

   ```
   nb max livres par emprunt : 4
   Lulu, 23 ans, 11.5 euros, 21 jours
   Riton, 24 ans, 12.0 euros, 21 jours
   Rita, 22 ans, 26.0 euros, 36 jours
   , rallonge 15
   ```

Partie 2 (11 points)

Attention : rédigez sur une nouvelle copie et placez les exercices dans l'ordre.

Dans cette partie, on considère la totalité de la situation du cahier des charges.

3. On reprend l'exercice 2, qu'on adapte. Dans cet exercice on considère, en plus des clients des exercices précédents, les livres et les emprunts.

   Ecrivez un programme complet répondant au cahier des charges. Ecrivez en particulier les parties de code suivantes (dans l'ordre que vous voulez). Pour chaque nouvelle fonction, vous justifierez en une ligne pourquoi elle est statique ou dynamique.

   (a) (3 points) Structure des classes, constructeurs, fonctions de base, etc.

   (b) Pour des raisons de lisibilité des copies, donnez le *main* complet (sans le traitement 3).

   (c) (1 point) Traitement 1.

   (d) (3 points) Traitement 2. Expliquez précisément le fonctionnement de tous les appels de fonction en jeu (5 à 6 lignes).

   (e) (1 point) Saisie.

   (f) (3 points) Traitement 3.

Partie 3 (2 points)

Attention : rédigez sur une nouvelle copie et placez les exercices dans l'ordre.

Cette partie ne porte pas sur la situation du début de l'énoncé.

4. (2 points) On considère l'exécution du programme suivant (il compile sans erreur).

   (a) Donnez la configuration de la mémoire immédiatement après l'exécution de la ligne contenant le commentaire : `// ici`. Vous utiliserez un croquis détaillé et l'algorithme vu en cours et TD. En particulier, on allouera obligatoirement les cases mémoire dans l'ordre croissant à partir de 101. Comme en cours et en TD, si une case contient successivement plusieurs valeurs, on les écrira de gauche à droite dans la case en les barrant d'un seul trait au fur et à mesure. De même, on ne réutilisera pas les zones de fonctions.

   (b) Donnez le dernier état de la mémoire avant l'arrêt du programme.

   (c) Poursuivez le déroulement en utilisant une autre couleur, et dites ce qu'affiche ce programme.

```
// MemoireExaUn2010.java
class Machin {
    static int d;
    int valeur;
}
class Bidule {
    int valeur;
    Machin obj;
    void g (Bidule b) {
        this.obj = b.obj;
    }
    static void f (Bidule b, Bidule c) {
        b.obj = new Machin();
        Machin.d += b.valeur;
        c = b;
        b.obj.valeur++;
    }
}
class BiduleSpecial
    extends Bidule {
    int autreValeur;
    void g (Bidule b) {
        this.obj.valeur += b.obj.valeur;
    }
}
class Memoire {
    public static void main (String[] args ) {
        Bidule[] t = new Bidule[3];
        t[1] = new Bidule();
        int n = 2;
        t[n] = t[n-1];
        Bidule.f(t[n-1], t[n-2]);
        System.out.println(Machin.d+" "+n+" "+
                           t[n-1].valeur+" "+t[n-1].obj.valeur+
                           t[n].valeur+" "+t[n].obj.valeur); // ici
        n--;
        t[n] = new BiduleSpecial();
        t[n].g(t[n-1]);
        System.out.println(Machin.d+" "+n+" "+
                           t[n-1].valeur+" "+t[n-1].obj.valeur+
                           t[n].valeur+" "+t[n].obj.valeur);
    }
}
```