IUT Belfort Montbéliard
Département Informatique
DUT S1 – Année 2012/2013

Programmation Orientée Objet
Devoir N°2

Durée : 1 semaine.
Noté sur 10 (50 % de la note du partiel N°2).
Évaluation en salle-machines pendant vos séances de TP, la semaine du 10.12.12

Ce qui est évalué dans ce devoir :

i) Votre capacité à écrire un programme JAVA syntaxiquement correct et qui fonctionne sans erreurs. Plus précisément, il s'agit d'évaluer vos capacités :
   -- à définir et utiliser une classe, un constructeur, des méthodes, des attributs/champs (variables d'instance) ;
   -- à effectuer des choix corrects pour les types d'attributs, des arguments/paramètres des méthodes et du type de retour de méthodes ;
   -- à indenter, commenter correctement votre code et générer la documentation de vos classes ;
   -- à donner une implémentation conforme à une classe et à définir et prendre en compte correctement la visibilité des attributs/champs et méthodes (public ou private) ...

ii) Votre capacité à implémenter des classes liées par une relation d'héritage en JAVA. C'est à dire :
   -- déterminer quelles méthodes doivent être redéfinies et quelles méthodes peuvent être héritées directement (sans redéfinition) ;
   -- à utiliser l'implémentation d'une méthode de la super classe lors de la redéfinition d'une méthode.

Sujet : On souhaite concevoir une application de gestion d'un Club d'adhérents. Un club est caractérisé par un nom et une liste d'adhérents qui sont des personnes étudiantes, salariées ou ni l'un ni l'autre.

1- en utilisant les classes *Personne*, *Etudiant* et *Salarie* fournies en annexe (vues publiques à écrire/implanter), écrire une classe *Club* qui rend les services suivants (certaines parties sont volontairement masquées) :

public class **Club** extends java.lang.Object

Constructor Summary

Club()	constructeur vide : ne fait rien
Club(Club c)	constructeur par copie
Club(String unNomClub, ...[] desAdherents)	initialise le *Club* courant

Method Summary

...[]	**getAdherents**() retourne les tableau des adhérents du *Club* courant
String	**getNomClub**() retourne le nom du *Club* courant
void	**init**() initialise interactivement le *Club* courant
String	**toString**() retourne la chaîne de caractères représentant le *Club* courant

a) écrire les variables d'instances.

b) écrire le constructeur vide et les méthodes d'accès

c) écrire le constructeur *Club(String unNomClub, ...[]desAdherents)* sachant que le tableau *desAdherents* contient les adhérents du *Club* que l'on est en train d'initialiser (ne pas faire de copie de ce tableau).

d) écrire le constructeur par copie. Remarque : il faut récupérer le tableau des adhérents du *Club c*, en faire une copie, ainsi qu'une copie de chacune des instances qu'il réfère. Afin de comprendre le mécanisme de copie :

   i. dessiner le tableau d'adhérents de *c* (avec les références et les instances) et le tableau d'adhérents de *this* que l'on veut obtenir
   ii. en déduire les instructions nécessaires (vous pouvez utiliser l'opérateur **instanceof**)

e) écrire la méthode *toString()* qui retourne une chaîne contenant le nom du *Club* courant et les information disponibles sur chaque adhérent

f) écrire la méthode *init( )* qui fonctionnera selon le mode suivant :

   i. demander le nombre d'adhérents à l'utilisateur
   ii. pour chacun d'eux : demander s'il est salarié, étudiant, ou ni l'un ni l'autre, créer et initialiser une instance en conséquence

2 - écrire et tester la classe *Club* (dans une classe *TestClub* créer et afficher un *Club c1* initialisée interactivement et un *Club c2* copie de celui référencé par *c1*).

Annexes

public class **Personne**

Constructor Summary

Personne()	constructeur vide (ne fait rien)
Personne(Personne p)	constructeur par copie
Personne(java.lang.String n, int a)	initialise la *Personne* courante avec nom, age

Method Summary

boolean	**equals**(Object o) retourne *true* si la *Personne* o a les mêmes caractéristiques que la *Personne* courante
int	**getAge**() retourne l'age de la *Personne* courante
String	**GetNom**() retourne le nom de la *Personne* courante
void	**init**() initialise interactivement la *Personne* courante
String	**toString**() retourne la chaîne de caractères représentant la *Personne* courante

public class **Salarie** extends Personne

Constructor Summary

Salarie() constructeur vide
Salarie(Salarie s) constructeur par copie
Salarie(String unNom, int unAge, String unNumeroSecu, String unEmployeur) initialise le *Salarie* courant

Method Summary

boolean	**equals**(Object o) retourne *true* si le *Salarie* référencé par *o* les mêmes caractéristiques que le *Salarie* courant
String	**getEmployeur**() retourne l'employeur du *Salarie* courant
String	**getNumeroSecu**() retourne le numéro de sécurité sociale du *Salarie* courant
void	**init**() initialise interactivement le *Salarie* courant
String	**toString**() retourne la chaîne de caractères représentant le *Salarie* courant

Methods inherited from class Personne
getAge, getNom

public class **Etudiant** extends Personne

Constructor Summary

Etudiant() constructeur vide
Etudiant(Etudiant e) constructeur par copie
Etudiant(String unNom, int unAge, String unNumeroEtudiant, String uneFac) initialise l'*Etudiant* courant

Method Summary

boolean	**equals**(Object o) retourne *true* si l'*Etudiant* référencé par *o* a les mêmes caractéristiques que l'*Etudiant* courant
String	**getFaculte**() retourne la faculté où étudie l'*Etudiant* courant
String	**getNumeroEtudiant**() retourne le numéro d'étudiant de l'*Etudiant* courant
void	**init**() initialise interactivement l'*Etudiant* courante
String	**toString**() retourne la chaîne de caractères représentant l'*Etudiant* courant

Methods inherited from class Personne
getAge, getNom