Programmation et Développements Orientés Objets 19357

Examen du Samedi 17 septembre 2005

Durée: 3 heures

Les documents sont autorisés.

Exercice 1 : typage java (5 points)

```
class Livre{
    String auteur, titre, editeur;
}
class Roman extends Livre{}
class Editeur{
    int compare(Livre l1, Livre l2, Livre l3){
        return 1;
    }
    int compare(Roman l1, Roman l2, Livre l3){
        return 2;
    }
    int compare(Livre l1, Livre l2, Roman l3){
        return 3;
    }
}
class PCNAM extends Editeur{
        int compare(Livre l1, Livre l2, Livre l3){
        return 4;
    }
}
class Typage{
    public static void main(String[] args){
        Livre lv1 = new Livre();
        Livre lv2 = new Roman();
        Roman rm1 = new Roman();
        Roman rm2 = new Roman();
        Roman rm3 = new Roman();
        Editeur ed = new PCNAM();
        // envois de message
        System.out.println(ed.compare(rm1,lv1,lv2));
        System.out.println(ed.compare(lv2,rm1,lv1));
```

```
        System.out.println(ed.compare(rm1,rm2,rm3));
        // conversions de type
        rm1 = (Roman) lv1;
        rm2 = (Roman) lv2;
        PCNAM pc = (PCNAM) ed;
        pc = (PCNAM) lv1;
    }
}
```

1. pour chacun des trois envois de messages de la méthode main, dites si cet envoi de message est correctement typé et si c'est le cas, donnez le type de la méthode utilisée par le typage à la compilation (il s'agit d'un type fonctionnel, avec une flèche).
2. pour chacun des envois de messages de la méthode main, donnez l'entier affiché par cet envoi de message à l'exécution. La question ne se pose que pour les envois de message corrects du point de vue du typage.
3. lesquelles des conversions de type explicites (cast) de la méthode main provoquent une erreur à la compilation ?
4. lesquelles des conversions de type explicites (cast) de la méthode main provoquent une erreur à l'exécution ?

Exercice 2 : programmation en Java 1.5 (7 points)

Une information peut être connue avec différents degrés de certitude : elle peut être sûre, incertaine ou fausse. On veut manipuler des données pour lesquelles on peut consulter et faire évoluer le degré de certitude.

Question 1

On propose l'interface suivante :

```
import java.util.Vector;
interface AvecStatut<S>{
    S getStatut();
    void setStatut(S s);
}
```

Ecrire une classe représentant le score d'un match de football (par exemple, Argentine-Brésil 3-1). On veut utiliser la notion de statut et l'interface `AvecStatut` pour modéliser le fait qu'un match est commencé, en cours ou terminé.

Question 2

On veut maintenant abstraire quelque peu le programme pour pouvoir donner le score de matchs dans différents sports : volley, handball, athlétisme. La façon d'exprimer un score est différente pour chacun de ces sports. Certains d'entre eux comportent plusieurs manches. Certains expriment les résultats en points, d'autre en buts, d'autre en temps. Ecrivez une classe `Score` qui implémente l'interface `AvecStatut`et qui soit générique en fonction de la représentation du score.

Question 3

Ecrivez une classe permettant de représenter un championnat dans lequels un certain nombre d'équipes se rencontrent pour des matchs, les résultats des matchs se traduisant en points. Tous les matchs de la saison sont enregistrés dès le début avec le statut *pas commencé*. Un calcul du classement actuel est possible à tout moment en ne prenant en compte que les résultat des matchs terminés.

Exercice 3 : Ocaml (8 points)

Un fournisserur d'accès sur internet propose deux types de contrats à ses clients : la classe `hDebit` modélise le contrat d'accès internet avec haut débit, alors que la classe `plusTel` "propose", en plus du haut débit, le téléphone illimité (hors numéros spéciaux). La classe `plusTel` possède une méthode `appels(n)` qui permet d'ajouter un nombre `n` d'appels vers des numéros spéciaux faits par un abonné, et une méthode `appelsHorsContrat()` calculant le côut supplémentaire pour ces appels.

```
class hDebit debit =
object(self)
    val deb = debit
    val prixBase = 24
    method prix() = prixBase
    method affiche()=  print_string "Debit  = "; print_int deb;
        print_newline(); print_string("Prix Mensuel =  ");
        print_int(self#prix()); print_newline()
end

class plusTel debit =
object(self)
   inherit hDebit debit as super
   val mutable nbAppels = 0
   method prix() = prixBase + 6
   method total() = self#prix() +  self#appelsHorsContrat()
   method appels(n) = nbAppels <- nbAppels+n
   method appelsHorsContrat() = 2*nbAppels
   method affiche() = super#affiche();
         print_string ("Cout appels hors contrat = ");
         print_int (self#appelsHorsContrat()); print_newline();
         print_string"Total = "; print_int (self#total());
         print_newline()
end
```

Partie I

On suppose définis :

```
# let h = new hDebit 10;;
# let p = new plusTel 20;;
# p#appels(5);;
```

1. **(1 point)** En considérant le code suivant :

```
# h#prix();;
# h#affiche();;
# p#prix();;
```

   (a) Donnez les réponses affichées par le top-level Ocaml.

   (b) Les appels `h#prix()` et `p#prix()` donnent-ils le même résultat ? **Justifiez**.

2. Considérez le code :

```
# p#total();;
# p#affiche();;
```

   (a) (**1 point**) Donnez les réponses affichées par le top-level.

   (b) (**0.5 point**) Chacune parmi `hDebit` et `plusTel` possède sa propre méthode `prix`. Laquelle des deux est exécutée lors de l'appel `p#total()` ? **Justifiez votre réponse**.

   (c) (**1.5 points**) Au cours de l'exécution de `p#affiche()`, la méthode `prix` est exécutée 2 fois. Pouvez-vous expliquer à quel moment se fait chaque appel, et dans chaque cas, quelle est la méthode `prix` exécutée ?

3. (**1.5 points**) Supposons que l'on élimine dans la classe `hDebit` toute référence à `self`. Concrètement, on ne changera que la méthode `affiche` comme indiqué par les soulignés `^^^^`, et on laissera le reste comme avant. On appelle `hDebitBis` cette nouvelle classe.

```
class hDebitBis debit =
object
    ....
    method affiche()=   print_string "Debit  = ";
        print_int deb;
        print_newline(); print_string("Prix Mensuel =  ");
        print_int(prixBase);   print_newline()
                  ^^^^^^^^
end
```

   On définit également une classe `plusTelBis`, identique à `plusTel` sauf qu'elle hérite maintenant de `hDebitBis`. En supposant que `h1` et `p1` sont définis par :

```
# let h1 = new hDebitBis 10;;
# let p1 = new plusTelBis 20;;
# p1#appels(5);;
```

   (a) Quelles sont les réponses du top-level pour les phrases suivantes ?

```
# h1#prix();;
# h1#affiche();;
# p1#prix();;
# p1#affiche();;
```

   (b) Les réponses pour `p1#affiche()` (ici), et pour `p#affiche()` (pour les anciennes versions des deux classes) sont-elles différentes ? Pourquoi ?

Partie II

Notre fournisseur d'accès souhaite maintenant modéliser un abonné internet générique, pouvant souscrire l'un parmi les deux sortes de contrat qu'il propose. La classe `['a]ãbonneInternet` est paramétrée par une variable de type `'a`, qui représente le type de contrat souscrit.

```
class ['a] abonnementInternet (nom, contrat) =
object
  constraint 'a = #hDebit
  val nom = nom
  val contrat = (contrat : 'a)
  method facture() =  print_string ("Nom: "^nom);
             print_newline(); contrat#affiche()
end
```

1. (**1.5 point)** On suppose définis :

   ```
   # let simple = new abonnementInternet ("Mr. Simple ", h);;
   # let enplus = new abonnementInternet ("Mr. Enplus ", p);;
   ```

   Donnez et *justifiez* le type donné par le top-level Ocaml pour ces objets. Quelle rapport entre ce typage et la contrainte `constraint 'a = #hDebit` dans le code de `abonneInternet` ?

2. (1 point) Quelle est le type de la variable d'instance `contrat` pour chacun des objets `simple` et `enplus` ? S'agit-il du même type ?