Groupe 4 :

Omar DIOP

Maimouna DIOUF

Baye Mbaye Biteye

Donald William Kodjo EVLO

Mohamed MBAYE

Kadidiatou Sow

Kardiatou THIAM

Adama SADJI

Ghislain Nodjiwam Djimrabaye

Exercice

Soit la définition de classe suivante:

```
class A {
public void f (int n, float x){ ..... }
public void f (float x1, float x2) { ..... }
public void f (float x, int n) { ..... }
}
```

Avec les déclarations suivantes:

```
A a;
short p;
int n1, n2;
float x;
```

La classe A est-elle correcte ? La corriger sinon.

Parmi les instructions suivantes, quelles sont celles qui sont correctes et, dans ce cas, quelles sont les méthodes appelées et les éventuelles conversions mises en jeu?

```
a.f(n1, x) ;
a.f(x, n1) ;
a.f(p, x) ;
a.f(n1, n2)
```

Reponse

1—Oui la classe est correcte car elle respecte les règles de surcharge des méthodes permises par java c'est-à-dire les signatures.

2- **a.f(n1,x)** : Correcte

Justificatif :

- On constate que l'appel de la méthode a.f(n1,x) prend en argument un entier n1 et un float x
- Au niveau de la classe A nous avons la première méthode **public void f(int n,float x){}** qui a les mêmes types de paramètres c'est-à-dire un entier et un float donc la première méthode est exécutée

a.f(x, n1) : Correcte

Justificatif :

- On constate que l'appel de la méthode a.f(x,n1) prend en argument un float x et un entier n1
- Au niveau de la classe A nous avons une première méthode **public void f(int n,float x){}** qui prend en argument un entier n et un float x, mais qui ne correspond pas à la signature de la méthode invoquée donc elle ne sera pas exécutée alors on passe à la méthode suivante.
- La méthode suivante est **public void f (float x1, float x2) { }** prend en argument deux float x1 et x2, mais qui ne correspond pas à la signature de la méthode invoquée donc elle ne sera pas exécutée alors on passe à la méthode suivante.
- La méthode **public void f (float x, int n) {}** prend en argument un float x et un entier n qui correspond à la signature de la méthode invoquée de ce fait elle va s'exécuter.

a.f(p, x) : Correcte

Justificatif

- On constate que l'appel à la méthode a.f(x,n1) prend en argument un short p et un entier n1
- Au niveau de la classe A nous avons une première méthode **public void f(int n,float x){}** qui prend en argument un entier n et un float x et est proche de la signature de la méthode invoquée alors on passe à la méthode suivante.
- La méthode suivante est **public void f (float x1, float x2) { }** prend en argument deux float x1 et x2, mais qui ne correspond pas à la signature de la méthode invoquée donc elle ne sera pas exécutée alors on passe à la méthode suivante.
- La méthode **public void f (float x, int n) {}** prend en argument un float x et un entier n qui correspond à la signature de la méthode invoquée de ce fait elle va s'exécuter.
- Parmi toutes les méthodes seule la méthode **public void f(int n,float x){}** est proche de la signature de la méthode, le premier argument est un entier et le deuxième est float de ce fait une conversion implicite va se produire on va convertir le short de la méthode invoquée en entier et de ce fait la première méthode est exécutée

a.f(n1, n2); : Incorrecte

Justificatif

- On constate que l'appel à la méthode a.f(n1,n2) prend en argument deux entiers n1 et n2
- L'appel à la fonction a.f(n1,n2) sur la classe A ne correspond directement à aucune des méthodes car aucune de ses méthodes ne prend en argument deux entiers de ce fait une conversion implicite ce fera ses les 3 méthodes :
- Au niveau de la première méthode **public void f(int n,float x){}** une conversion va se produire au niveau du deuxième argument n2 de la méthode invoqué en float
- Au niveau de la deuxième méthode **public void f (float x1, float x2) {}** deux conversion vont se produire au premier argument n1 de la méthode invoquée en float et le deuxième argument n2 de la méthode invoqué en float
- Au niveau de la troisième méthode **public void f (float x, int n) {}** une conversion va se produire au niveau du premier argument n1 de la méthode invoqué en float
- On remarque que sur la première et troisième méthode il y'a qu'une seule conversion par contre la deuxième y'a deux conversions donc ils sont plus efficaces.
- Conséquence : Problème d'ambiguïté entre les deux méthodes **public void f(int n,float x){} et public void f (float x, int n) {}**.
- La compilation va retourner une erreur pour cause d'ambigüité.