Corrigés exercices chapitre 1 :

Exercice 1.2

- non, un référent pointe vers un objet unique
- oui, un même objet peut être accédé par plusieurs référents
- oui, l'attribut d'un objet peut être le référent vers un autre. C'est une manière pour ces deux objets d'établir une communication.
- La classe est le modèle de l'objet. Elle en décrit sa structure et ses activités.
- L'objet change constamment d'état comme le résultat de l'exécution successive de ses méthodes. Seules les méthodes de l'objet peuvent modifier son état.
- Cela signifie que la méthode f(x) va s'appliquer sur l'objet « a ».
- La méthode f(x) doit être déclarée dans la classe qui décrit l'objet a et dont l'objet a est une instance.
- Un envoi de message se produit entre un objet a et un objet b quand l'objet a déclenche l'exécution d'une méthode sur l'objet b.
- Le premier objet déclenche une méthode propre au deuxième qui a pour fonction de modifier son état.

Exercice 1.3 :

- humain, sportif, puis ensemble, en dessous de sportif, footballeur et skieur, en-dessous de skieur, spécialiste du slalom et en dessous de footballeur, avant-centre.
- Instrument de musique, puis ensemble en dessous d'instrument de musique, instrument à corde, instrument à vent, voix, en dessous d'instrument à corde, violon (la voix pourrait être également considérée comme un instrument à corde), puis en dessous d'instrument à vent, trompette et saxophone.

Corrigés exercices chapitre 2 :

Exercice 2.2

Seule les trois premières sont possibles. La dernière modifie la valeur du « retour » et sera considérée une erreur à la compilation. Au moment de l'exécution, on ne sera pas laquelle de deux versions, il faudra exécuter. La nature du retour ne fait pas partie de la signature de la méthode.

Exercice 2.3

Pour le code « A.java », le constructeur de la classe A ne possède aucun retour. Il faut donc supprimer le « void » qui le précède. Son mode d'appel est unique. Notez qu'en Java, le code ainsi écrit définit une nouvelle méthode qui pourrait exister en plus du constructeur. En C#, c'est impossible on ne peut créer une méthode avec un « retour » spécifique et qui porte le même nom que le constructeur.

Exercice 2.6

Pour le code « PrincipalTest.java », un constructeur a été défini dans la classe Test, le constructeur par défaut n'est plus actif, un objet ne peut être construit qu'en utilisant ce nouveau constructeur. Si l'on veut utiliser un constructeur par défaut (ne recevant aucun argument, il faut le redéfinir explicitement).

Pour le code « PrincipalTest.cs », la valeur « 6.5 » est un réel et les seuls deux constructeurs ne sont prévus que pour recevoir en argument des entiers.

Pour le code « PrincipalTest.cpp », là encore, aucun constructeur par défaut n'a été prévu pour satisfaire la construction du troisième objet.

Exercice 2.7

Pour le code « PrincipalTest.java », un attribut non statique ne peut être référencé dans une méthode déclarée statique. Il faut se rappeler que les méthodes statiques peuvent s'appeler à partir de leur classe et donc ne doivent pouvoir traiter que des attributs de classe.

Pour le code « PrincipalTest.cs », il est impossible en C# d'appeler une méthode statique à partir d'un objet, à la différence de Java, on ne peut toujours l'appeler qu'à partir de la classe.

Pour le code « PrincipalTest.cpp », il est indispensable de rajouter en dessous de la déclaration de la classe, une initialisation de l'attribut statique comme : « int Test::c=5 ».