TP 7

1 Container Tableau, Itérateur et Inserteur

Finir le type abstrait `Tableau<T,AGRANDISSEMENT>` avec la classe d'`Iterator` associée. Définissez une classe Iterateur d'insertion à la fin du type Tableau (bien relire les slides du cours 6 relatifs à ce sujet avant de se lancer).

Manque t-il des éléments à vos classes pour pouvoir utiliser votre propre classe de `Container Tableau` en lien avec les `Algorithms` de la `STL` ?

2 Déplacement pour la hiérarchie de classe `Expression`

Reprendre votre hiérarchie de classe `Expression` permettant la gestion d'expressions binaires polymorphes et mettre une trace de passage dans chacun de vos constructeurs, destructeurs et opérateurs d'affectation. Que constatez vous lors de l'exécution du programme principal ? Complétez votre hiérarchie de classe de manière à optimiser le cas où vous construisez des Expressions à partir de temporaires, en utilisant une stratégie de déplacement. Vous aurez notamment besoin de munir votre hiérarchie de classe d'une méthode qui réalisera le clone d'un temporaire. Attention de bien faire appel à `std::move` partout où cela sera nécessaire. En quoi vos ajouts modifient t-ils le comportement du programme principal ?

3 Utilisation de la STL

Finir cet exercice si vous ne l'avez pas fait au TP précédent.

4 De C++ à Java ou Vice Versa (3)

Modifiez la classe $Image$ des TPs précédents en la *templatant* par le type de ses pixels. Dans cette version, il ne sera donc plus nécessaire de disposer d'une hiérarchie de classe $Pixel$ polymorphe. Il suffira à l'utilisation de *templater* directement la classe $Image$ par des $PixelNiveauGris$ ou des $PixelCouleurs$. Que constatez-vous en terme d'efficacité ?