# TD 4 - Forme Canonique de Coplien

Programmation Orientée Objet

## Objectif

- Comprendre le principe de la forme canonique de Complien

## 1 La Classe chaîne

On vous demande de créer une nouvelle classe selon la forme canonique de Coplien. Celle-ci est une chaîne simple dotée de 3 attributs :

- un tableau de caractères (`char *`) contenant la chaîne proprement dite.
- Un entier dénotant la capacité de la chaîne, c'est à dire le nombre de caractères qu'elle peut physiquement contenir à un instant donné. Il s'agit donc de la capacité physique du tableau de caractères.
- Un entier dénotant la longueur de la chaîne, c'est à dire le nombre de caractères significatifs qu'elle contient. Il s'agit de l'information renvoyée par l'application de la fonction `strlen` sur le tableau de caractères.

Le constructeur par défaut créera une chaîne de longueur nulle mais de capacité 10 caractères. Le constructeur par recopie devra positionner correctement la capacité et la longueur mais également allouer la mémoire du tableau de caractères et recopier le modèle. L'opérateur d'affectation, après nettoyage de la mémoire, devra allouer la mémoire puis recopier la chaîne du modèle. Attention, il ne faut surtout pas utiliser `strdup` car ce dernier utiliser `malloc`, incompatible avec `new` et `delete`.

## 2 La Classe chaîne intelligente

Modifiez la classe précédente de manière à prendre en compte les points suivants. On souhaite rendre ces chaînes "intelligentes" en leur permettant de ne stocker qu'une fois les chaînes identiques. L'idée est de faire en sorte que si une chaîne `ch1` contient "coucou", alors l'affectation `ch2 = ch1` provoque la situation illustré par la figure FIG. 1 en mémoire :

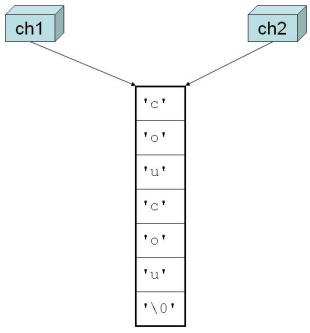


Fig. 1 – Exemple de chaîne intelligente

Ainsi la destruction de ch2 ne provoque pas la libération de la zone mémoire contenant les caractères, tandis

que la destruction de ch1 (en considérant qu'elle est la " dernière " à contenir ces caractères) provoque la libération de la zone

Ce changement de comportement va impliquer un changement de structure de la classe. Modifiez la structure en conséquence, et donc aussi toutes les méthodes associées.