

# IN3S02 : Sujet du TD C++

Groupe ESIEE, Denis BUREAU, novembre 2004.

**Attention !** Le sujet peut être modifié jusqu'à la veille du TD.

## 1 Les objectifs

Écrire de premiers programmes C++ en utilisant les notions les plus importantes (passages de paramètres, chaînes de caractères, pointeurs, allocation dynamique, etc...).

## 2 Notions vues dans les 2 cours C/C++

- Commentaires, identificateurs, affectation
- Affichage, saisie
- Types de données primitifs, déclarations de variables et de constantes
- Opérateurs, expressions, fonctions mathématiques, conversions
- Tests, boucles
- Fonctions, procédures, ordre des déclarations
- Passage de paramètres
- Tableaux, chaînes de caractères
- Définition de nouveaux types, structures
- Pointeurs, passage par adresse en C
- Équivalence tableaux  $\leftrightarrow$  pointeurs, arithmétique des pointeurs
- Allocation dynamique de types simples, de tableaux, et de structures
- Notions de maillon et de liste chaînée

## 3 Premier programme

Écrire un programme complet qui affiche simplement **Bonjour !**.

On s'attachera à ce que la syntaxe soit 100% correcte.

## 4 Passage de paramètres

1. Écrire une fonction **hms2s** qui prend en entrée 3 entiers représentant un nombre d'heures, de minutes, et de secondes, et qui retourne le nombre total de secondes que cela représente. Il n'y a pas de limitation sur la valeur de chacun des 3 nombres, mais on supposera que le nombre total de secondes ne dépasse pas la capacité maximale du type **int**.
2. Écrire une procédure **s2jhms** qui prend en entrée un nombre de secondes et qui fournit en sortie le nombre de jours, d'heures, de minutes, et de secondes que cela représente.
3. Écrire un programme principal qui saisit 3 entiers représentant un nombre d'heures, de minutes, et de secondes, sans limitation particulière, qui les convertit en un nombre de jours, d'heures, de minutes, et de secondes, et qui affiche le résultat. Ce programme appellera bien évidemment les 2 sous-programmes définis précédemment.  
Exemple d'exécution : 70 70 70  $\rightarrow$  2j 23h 11mn 10s

## 5 Chaînes de caractères

1. Définir une constante `MAX` par exemple égale à 10.
2. Définir un type `Chaine` = tableau de `MAX` caractères.
3. Écrire une procédure `affiche` qui prend une chaîne de caractères en entrée et qui affiche la chaîne entre parenthèses.
4. Écrire un programme principal qui saisit 2 chaînes de caractères et qui boucle 8 fois en affichant alternativement chacune des 2 chaînes. Pour cela, il est possible d'utiliser un compteur dont la valeur vaut alternativement 1 ou 2; suivant cette valeur, le programme recopiera l'une des 2 chaînes de caractères dans une troisième, et appellera `affiche` sur cette dernière.

**Remarque :** Si `t` est un tableau de caractères, `cin >> t;` permet de saisir d'un seul coup un mot dans `t`. Par contre, il n'est pas possible de saisir ainsi des phrases avec des espaces. Dans ce cas, il faut utiliser : `cin.getline( t, taille_de_t );`.

## 6 Pointeurs et chaînes de caractères

Modifier le programme précédent pour :

1. qu'il déclare un type `PtrChar` = pointeur de caractère,
2. que le paramètre d'`affiche` soit un `PtrChar`,
3. que le programme principal ne recopie plus une chaîne de caractères à chaque tour de boucle, mais se contente de recopier un pointeur.

**Remarque :** il y a en tout seulement 6 lignes à modifier.

## 7 Pointeurs et allocation dynamique

1. Définir une constante `INCREMENT` par exemple égale à 2.
2. Définir un type `PtrDouble` = pointeur de `double`.
3. Écrire un programme principal qui saisit une taille initiale, qui alloue un tableau de `double` de cette taille, puis qui rentre dans la boucle suivante : tant que le nombre saisi est positif, le stocker dans le tableau (après avoir stocké un premier nombre négatif, la boucle s'arrête) tout en appelant `agrandit` à chaque fois que nécessaire.
4. Écrire une procédure `agrandit` qui a comme paramètres d'entrée/sortie un pointeur de `double` (pointant sur un tableau de `double`) et un entier (représentant la taille du tableau), et qui agrandit le tableau de `INCREMENT` cases, sans bien sûr perdre les valeurs qui s'y trouvaient.