Les chaînes de caractères en C.

Patrick Poulingeas.

- En C, il n'y a pas à proprement parler de type 'chaîne de caractères', mais une convention de représentation qui fait que l'on interprète une suite de caractères comme une chaîne. Pour le C, une chaîne est une suite de caractères consécutifs en mémoire se terminant par le caractère '\0' (c'est-à-dire l'octet 0).

- Une façon d'introduire une variable chaîne de caractères est la déclaration suivante :
  char chaine[10] ;
  On dispose alors d'une variable de nom 'chaine' dans laquelle on pourra stocker au plus 9 caractères (En effet, il ne faut surtout pas oublier le '\0' marquant la fin de la chaîne !).

- Littéraux chaînes de caractères :
  Ils sont entourés de guillemets.
  Exemple d'utilisation :
  char chaine[10] = « salut » ;
  On peut se contenter d'écrire :
  char chaine[] = « salut » ;
  Le compilateur réservera un espace de 6 octets (1 char = 1 octet) pour stocker la chaîne (6, et non pas 5 ! Pensez au '\0' de fin – que le compilateur n'oublie pas.)

- Erreurs à ne pas commettre :

  - Une variable de type caractère n'est pas une chaîne, aussi il est très dangereux de fournir un caractère en argument à une fonction qui attend une chaîne.

  - Impossible de faire des comparaisons avec l'opérateur = =
    if (chaine == « salut »)
    donne un résultat imprévisible, quel que soit le contenu de la variable 'chaine'.

  - Impossible de faire des affectations entre chaînes :
    Exemple :
    char chaine1[] = "salut";
    char chaine2[6];

    chaine2 = chaine1;
    Message d'erreur du compilateur :
    error: ISO C++ forbids assignment of arrays

  - Impossible de faire des concaténations avec des chaînes ou des caractères à l'aide de l'opérateur + (comme en Pascal) :
    Exemple :
    char chaine[] = "salut";
    chaine = chaine + "la";

Message d'erreur du compilateur :
error: invalid operands of types `char[6]' and `const char[3]' to binary `operator+'

- Pour travailler avec les chaînes (faire des comparaisons, des copies, des concaténations, etc.), il faut utiliser des fonctions spéciales. Ces fonctions sont placées dans le fichier string.h de la bibliothèque standard du C. Comme on écrit des programmes C++, on fera appel à cstring et non pas à string.h.

Pour disposer des fonctions de manipulation des chaînes, on ajoutera donc en début de programme :

#include <cstring>

- Quelques fonctions de cstring :

  - Calcul de la longueur d'une chaîne :
    strlen(chaine)
    Exemple :
    cout << strlen(« Matrix II ») ;
    affiche la valeur 9.
    Remarque :
    Pour tester si une chaîne est vide, on peut tester si sa longueur est égale à 0.

  - Comparaison entre deux chaînes :
    strcmp(chaine1,chaine2)
    Cette fonction renvoie :
    - 0 si chaine1 = chaine 2,
    - une valeur < 0 si chaine1 < chaine2 (pour l'ordre ASCII),
    - une valeur > 0 si chaine1 > chaine2 (pour l'ordre ASCII).

  - Concaténation de deux chaînes :
    strcat(chaine1,chaine2)
    Cette fonction effectue ce qu'algorithmiquement on écrirait :
    chaine1 ← chaine1 + chaine2

    Attention :
    - La taille du tableau chaine1 doit être assez grande pour pouvoir contenir le résultat de la concaténation.
    - chaine2 ne peut pas être un caractère (Nous en avons déjà parlé).

Pour concaténer un caractère à une chaîne, on fabrique une chaîne contenant le caractère. Par exemple :

```
char caractere = 'e';
char chaine[256] = « amical » ;
char caractere_a_concatener[2] ;

caractere_a_concatener[0] = caractere ;
caractere_a_concatener[1] = '\0' ;

strcat(chaine, caractere_a_concatener) ;

cout << chaine ;        // Affiche : amicale
```

Alors que le programme erroné suivant :

```
char caractere = 'e';
char chaine[256] =  "amical";

strcat(chaine, caractere) ;
```

entraîne le message d'erreur du compilateur :
error: invalid conversion from `char' to `const char*'

- Copie d'une chaîne dans une autre :
  strcpy(chaine1,chaine2)
  Cette fonction copie le contenu de chaine2 dans chaine1 (Notez l'ordre !).
  C'est l'équivalent en C de l'écriture algorithmique suivante :
  chaine1 ← chaine2
  c'est-à-dire de l'affectation algorithmique.

Attention là aussi à la taille de chaine1 qui doit être assez grande pour pouvoir contenir chaine2.