# Chapitre 6 : Les chaînes de caractères

**Objectifs**

- *Connaître la convention de représentation des chaînes de caractères en C*
- *Comprendre les entrées -sorties de chaînes*
- *Manipuler les fonctions de traitement et de conversion de chaînes*

**NB :** Il n'existe pas de type spécial *chaîne* ou *string* en C. Une chaîne de caractères est traitée comme un *tableau à une dimension de caractères.*

## 1. Déclaration

La déclaration d'une chaîne de caractère se fait par l'une des méthodes suivantes :

- ✓ **char NomChaine [Longueur];**
- ✓ **char * NomChaine ;**

Exemple :

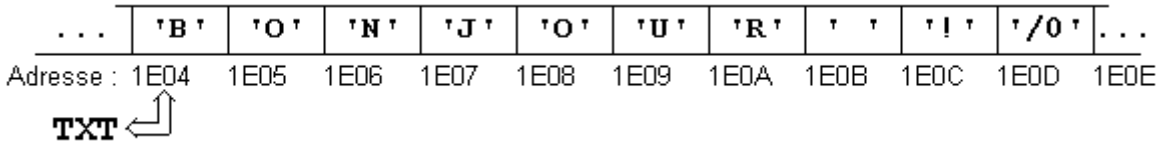
```
char NOM [20] ;       /*déclarartion sans initialisation */
char Prenom [] = « ALI » ;  /* déclaration avec initialisation */
```

## 2. Mémorisation

Le nom d'une chaîne est le représentant de l'adresse du premier caractère de la chaîne. Pour mémoriser une variable qui doit être capable de contenir un texte de N caractères, nous avons besoin de N+1 octets en mémoire:

Exemple :

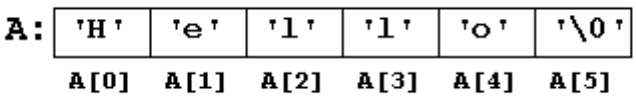
```
char TXT [10] = « BONJOUR !» ;
```




## 3. Accès aux éléments

Exemple :

```
char TXT [6] = « Hello» ;
```




## 4. Utilisation des chaînes de caractères

Les bibliothèques de fonctions de C contiennent une série de fonctions spéciales pour le traitement de chaînes de caractères.

Exemples 1: les fonctions de stdio.h

```
char TXT [6] = « Hello» ;
char CH[5] ;
printf(« %s »,TXT) ; /* affiche hello sans retour à la ligne*/
printf(«%s »,CH) ;  /* CH contient l'adresse du premier caractère de la châne */
puts(TXT) ;          /* affiche Hello avec un retour à la ligne */
puts(« bonjour ») ; /*affiche bonjour avec un retour à la ligne */
gets(CH) ;           /* lit une ligne de caractères de stdin
```

Exemples 2: les fonctions de string.h

```
char CH1 [] = « hello » ;
char CH2 [] = « bonjour » ;
char * chaine = « bonsoir » ;
int k=2 ;
char c='a' ;
strlen (CH1) ;    /* fournit la longueur de CH1 */
strcpy (CH1, CH2) ;  /* copie CH1 dans CH2 */
strcat (CH1, CH2) ;  /*ajoute CH2 à la fin de CH1 */
strcmp (CH1, CH2) ;  /* compare CH1 et CH2 lexico graphiquement */
….
```

strcmp renvoie un nombre:

- positif si la chaîne1 est supérieure à la chaine2 (au sens de l'ordre lexicographique)
- négatif si la chaîne1 est inférieure à la chaîne2
- nul si les chaînes sont identiques.

Remarque:
Comme le nom d'une chaîne de caractères représente une adresse fixe en mémoire, on ne peut pas 'affecter' une autre chaîne au nom d'un tableau. Il faut bien copier la chaîne caractère par caractère ou utiliser la fonction **strcpy** respectivement **strncpy**:

```
strcpy(CH1, "bonsoir");
```

Exemples 3: les fonctions de stlib.h

La bibliothèque <stdlib> contient des déclarations de fonctions pour la conversion de nombres en chaînes de caractères et vice-versa.

```
char CH [] = « 125 » ;
atoi (CH) ;   /* retourne la valeur numérique par CH comme int */
atol (CH) ;  /* retourne la valeur numérique représenté par CH comme long */
atof (CH) ; /* retourne la valeur numérique représenté par CH comme double */
```

**Règles générales pour la conversion:**

- ✓ Les espaces au début d'une chaîne sont ignorés
- ✓ Pour une chaîne non convertible, les fonctions retournent zéro

Exemples 3: les fonctions de ctype.h

Les fonctions de <ctype> servent à classifier et à convertir des caractères. Les symboles nationaux (é, è, ä, ü, ß, ç, ...) ne sont pas considérés. Les fonctions de <ctype> sont indépendantes du code de caractères de la machine et favorisent la portabilité des

programmes. Dans la suite, c représente une valeur du type **int** qui peut être représentée comme caractère.

***isupper (c)*** *; /* retourne une valeur différente de 0 si c est une majiscule */*
***islower (c)*** *; /*retourne une valeur différente de 0 si c est une miniscule */*
***isspace (c)*** *; /* retourne une valeur différente de 0 si c est un signe d'espacement */*

Les fonctions de **conversion** suivantes fournissent une valeur du type **int** qui peut être représentée comme caractère; la valeur originale de **c** reste inchangée:

**tolower (c)** ; /* retourne c converti en miniscule si c est une majuscule */

**toupper (c)** ; /* retourne c converti en majuscule si c est une minuscule */

## 5. Tableaux de chaîne de caractères

Souvent, il est nécessaire de mémoriser une suite de mots ou de phrases dans des variables. Il est alors pratique de créer un tableau de chaînes de caractères, ce qui allégera les déclarations des variables et simplifiera l'accès aux différents mots (ou phrases).

- ✓ Declaration : elle se fait par l'une des deux méthodes

```
char JOUR [7] [9] ;
char * mois [12] ;
```

- ✓ Initialisation : lors de la déclaration.

```
char JOUR [7] [9] = {"Lundi", "Mardi","Mercredi","Jeudi","Vendredi","Samedi", "Dimanche"};
char * MOIS[12] = {"jan","fev","mars","avr", "may", "juin","juil","aout", "sep", "oct","nov", "dec"} ;
```

- ✓ Accès aux différents composantes :

```
printf("%s",JOUR[2]) ;  /* affiche Mercredi */
printf("%s",MOIS[0]) ;  /* affiche jan */
```

- ✓ Accès aux caractères : (similaire au cas des tableaux)

```
for (i=0 ;i<7 ;i++)
      printf("%c",JOUR[1][0] ;
```

- ✓ Affectation : pas d'affectation directe (genre JOUR [2] = "vendredi" ;). Utiliser strcpy

```
strcpy(JOUR[4], "Friday") ;
```