CHAPITRE 3 : LES CHAÎNES DE CARACTÈRES

I. Introduction :

Une chaîne de caractères est une suite de caractères quelconques. Ce type d'objets est très utilisé en informatique. Les caractères lus sont ceux définis dans le code ASCII.

En C, il n'existe pas de type de variables particulier pour représenter une chaîne de caractères, mais il est possible d'utiliser un tableau de caractères. C'est une convention et il existe de nombreuses fonctions standard permettant de manipuler les tableaux de caractères comme des chaînes de caractères, à la seule condition que le dernier caractère de ces chaînes soit le caractère de code ASCII 0 (qui peut s'écrire '\0').

II. Définition :

Une chaîne de caractère est donc représentée en C par un tableau de caractères, et doit toujours se terminer par le caractère '\0'. Ce tableau de caractères doit être assez grand pour contenir toute les caractères de la chaîne plus un emplacement pour le caractère de fin.

III. Initialisation de tableau de caractères :

Il est possible d'initialiser directement un tableau de caractères avec une chaîne de caractères constante :

char T[6] = "OK" ;

Cette déclaration réserve en mémoire un emplacement de 6 octets, contenant :

'O'	'K'	'\0'	?	?	?

Les fonctions **printf** et **scanf** peuvent être utilisées pour lire est écrire (Afficher) des chaînes de caractères avec le code format **%s**.

Attention : En C, le nom d'un tableau permet de représenter l'adresse de ce tableau (C'est à dire l'adresse du premier caractère de ce tableau). Lorsque l'on utilise la

fonction *scanf* pour lire une chaîne de caractères avec le code **%s**, il ne faut donc pas mettre l'opérateur **&** devant le nom du tableau de caractères passé en argument.

```
#include <stdio.h>

main()
  {
      char nom[32] ;

      printf( "Donnez votre nom : ") ;
      scanf("%s", nom) ;          /* ou scanf( "%s", &nom[0] ) ; */

      printf( " votre nom est : %s\n", nom ) ;
  }
```

Remarques :

- Dans l'exemple précédent, si l'utilisateur entre plus de 31 caractères le programme ne marche plus. (le tableau n'aurait pas assez de cases pour contenir toutes les lettres de la chaînes plus '\0').
- La fonction **scanf** ne permet de lire que des mots ne contenant pas d'espace ou de sauts de lignes (elle saute les premiers blancs et sauts de lignes)

Il existe de nombreuses fonctions standards pour manipuler des chaînes de caractères. La plupart de ces fonctions peuvent être utilisées en incluant la bibliothèque **string.h**.

IV. Lecture et écriture spécifiques : les fonctions gets et puts

Ces fonctions sont des versions spécifiques de lecture et d'écriture d'une chaîne de caractères.

IV-1. La fonction gets :

Syntaxe : **gets (** *chaîne* **) ;**

Bibliothèque : **stdio.h**

La fonction **gets** permet de lire une chaîne de caractères : les espaces blancs sont lus comme tout autre caractère, et la saisie ne s'arrête que lorsqu'un caractère de fin de ligne est rencontré.

Cette fonction permet donc de lire la prochaine ligne entrée au clavier, et de mettre son contenu dans la variable qui lui est fournie en paramètre.

IV-2. La fonction puts :

Syntaxe : **puts (** *chaîne* **) ;**

Bibliothèque: **stdio.h**

La fonction **puts** permet d'écrire à l'écran le contenu de la chaîne de caractères fournie en paramètre, en ajoutant un caractère de retour à la ligne à la fin.

```
#include <stdio.h>

main()
   {
        char nom[32] ;

        printf( "Donnez votre nom : ") ;
        gets(nom) ;            /* ou gets( &nom[0] ) ;*/

        printf( " votre nom est ") ;
        puts(nom) ;
   }
```

Affichage :
Donnez votre nom : ALAOUI CHERIF
votre nom est ALAOUI CHERIF

V. Obtenir la longueur d'une chaîne : la fonction strlen

Cette fonction permet de déterminer la longueur d'une chaîne de caractères.

Syntaxe : n=**strlen (** *chaîne***) ;**

Bibliothèque: **string.h**

Cette fonction retourne la longueur de la chaîne passée en argument, c'est à dire le nombre de caractères contenus dans le tableau de caractères, jusqu'au premier caractère de fin de chaîne rencontré. ***Le caractère de fin de chaîne n'est pas compté.***

VI. Comparaison de chaînes : la fonction strcmp

Syntaxe : n=**strcmp (** *chaîne1, chaîne2* **) ;**

Bibliothèque: **string.h**

Cette fonction retourne une valeur entière positive si chaîne1 est supérieure à chaîne2 suivant l'ordre alphabétique (idem dictionnaire), la valeur zéro si elles sont identiques et une valeur négative sinon.

```
#include <stdio.h>
#include <string.h>

main()
  {
       char txt1[32] ;
       char txt2[32] ;
       int n ;

       printf( "Donnez vos deux  mots: ") ;
       scanf( "%s, txt1) ;
       scanf( "%s, txt2) ;

       n = strcmp( txt1, txt2) ;
       if      ( n <  0) printf( "Dans l'ordre alphabétique" );
       else if  ( n == 0 ) printf( "Identiques" );
       else if  ( n >  0 ) printf( "Pas dans l'ordre alphabétique" );
  }
```

VII. Recopie de chaînes : la fonction strcpy

Syntaxe : **strcpy (** *chaîne1, chaîne2* **) ;**

Bibliothèque: **string.h**

Cette fonction permet de copier le contenu de chaîne2 dans la variable chaîne1. Cette fonction recopie tous les caractères qu'elle rencontre jusqu'à recopier un caractère '\0' (Si chaîne2 n'est pas correctement terminée par ce caractère cela peut durer longtemps et poser de gros problèmes).

VIII. Concaténer deux chaînes de caractères

Syntaxe : **strcat (** *chaîne1, chaîne2* **) ;**

Bibliothèque: **string.h**

IX. Autres fonctions :

Il existe des dizaines d'autres fonctions concernant les chaînes de caractères. Vous les découvrirez à l'occasion grâce à l'aide en ligne.

Annexe

Table ASCII

(American Standard Code for Information Interchange)

Partie 1 :

Dec	Char	Dec	Char	Dec	Char	Dec	Char
0	Null	32	Space	64	@	96	`
1	Start of heading	33	!	65	A	97	a
2	Start of text	34	"	66	B	98	b
3	End of text	35	#	67	C	99	c
4	End of transmit	36	$	68	D	100	d
5	Enquiry	37	%	69	E	101	e
6	Acknowledge	38	&	70	F	102	f
7	Audible bell	39	'	71	G	103	g
8	Backspace	40	(	72	H	104	h
9	Horizontal tab	41	)	73	I	105	i
10	Line feed	42	*	74	J	106	j
11	Vertical tab	43	+	75	K	107	k
12	Form feed	44	,	76	L	108	l
13	Carriage return	45	-	77	M	109	m
14	Shift out	46	.	78	N	110	n
15	Shift in	47	/	79	O	111	o
16	Data link escape	48	0	80	P	112	p
17	Device control 1	49	1	81	Q	113	q
18	Device control 2	50	2	82	R	114	r
19	Device control 3	51	3	83	S	115	s
20	Device control 4	52	4	84	T	116	t
21	Neg. acknowledge	53	5	85	U	117	u
22	Synchronous idle	54	6	86	V	118	v
23	End trans. block	55	7	87	W	119	w
24	Cancel	56	8	88	X	120	x
25	End of medium	57	9	89	Y	121	y
26	Substitution	58	:	90	Z	122	z
27	Escape	59	;	91	[	123	{
28	File separator	60	<	92	\	124	|
29	Group separator	61	=	93	]	125	}
30	Record separator	62	>	94	^	126	~
31	Unit separator	63	?	95	_	127	□

Partie 2 :

Dec	Char	Dec	Char	Dec	Char	Dec	Char
128	Ç	160	á	192	└	224	α
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Γ
131	â	163	ú	195	├	227	π
132	ä	164	ñ	196	─	228	Σ
133	à	165	Ñ	197	┼	229	σ
134	å	166	ª	198	╞	230	µ
135	ç	167	º	199	╟	231	τ
136	ê	168	¿	200	╚	232	Φ
137	ë	169	⌐	201	╔	233	Θ
138	è	170	¬	202	╩	234	Ω
139	ï	171	½	203	╦	235	δ
140	î	172	¼	204	╠	236	∞
141	ì	173	¡	205	═	237	ø
142	Ä	174	«	206	╬	238	ε
143	Å	175	»	207	╧	239	∩
144	É	176	░	208	╨	240	≡
145	æ	177	▒	209	╤	241	±
146	Æ	178	▓	210	╥	242	≥
147	ô	179	│	211	╙	243	≤
148	ö	180	┤	212	╘	244	⌠
149	ò	181	╡	213	╒	245	⌡
150	û	182	╢	214	╓	246	÷
151	ù	183	╖	215	╫	247	≈
152	ÿ	184	╕	216	╪	248	°
153	Ö	185	╣	217	┘	249	∙
154	Ü	186	║	218	┌	250	·
155	¢	187	╗	219	█	251	√
156	£	188	╝	220	▄	252	ⁿ
157	¥	189	╜	221	▌	253	²
158	₧	190	╛	222	▐	254	■
159	ƒ	191	┐	223	▀	255	