Travaux Dirigés 5 : Chaînes de caractères

Notes de cours :

- **Description:** Une chaîne de caractère est une suite de caractères (ASCII) terminée par le caractère de code ASCII nul '\0'.
- **Déclaration :** var <identifiant> : chaine(<taille maximale>) ;
  où <taille maximale> est une constante entière positive égale au nombre maximum de caractères ('\0' compris).
- **Constantes :**Une constante chaîne de caractères est mise entre cotes simples. Exemple :
  'je suis une chaine constante'
- **Actions de base :** On peut lire ou ecrire une variable chaîne comme on peut lui affecter une autre.
  Exemple : Algorithme manipChaine ;

```
var         maChaine1 :chaine(20);
            maChaine2 :chaine(10);
debut
    lire(maChaine1) ; {mettra '\0' automatiquement}
    maChaine2 ← maChaine1 ; {idem}
    ecrire(maChaine2) ;
fin.
```

  - Si la chaîne lue ou affectée contient plus de caractères que la taille maximale de la chaîne réceptrice, cette dernière contiendra les <taille maximale>-1 premiers caractères puis '\0' uniquement ; i.e. les caractères en plus sont ignorés.
  - On peut affecter un caractère a une variable chaine et le résultat est une chaine composée du caractère affecté suivi du caractere '\0'.
- **Dégroupage :** On peut accéder à un élément (qui est de type caractère) de la chaîne de caractères par son indice mis entre () ou [ ].
  Exemple :
  var maChaine : chaine(10) ;
  debut maChaine ← 'bonjour' ; Fin.
  Après l'affectation, maChaine[1] contient le caractère 'b' et maChaine[8] contient '\0' ;
- **Opérateurs :**
  - On peut comparer deux chaînes de caractères par les opérateurs <, >,<=,>=,=,<> et le résultat booléen dépendra de l'ordre lexicographique déterminé par le code ASCII de leurs caractères.
  - L'opérateur + appliqué sur deux chaînes (une chaîne et un caractère ou deux caractères) permet de les concaténer (sans incidence sur ces chaînes).
    Exemple : maChaine1 ← maChaine2+maChaine3 ;
- **Fonctions prédéfinies :**
  - longueur(maChaine) : donne la longueur courante de maChaine sans compter '\0'.
  - ord(<caractere>) : donne le code ASCII du caractère <caractere>.
  - car(numeroCode) : donne le caractère ayant le code ASCII numeroCode.
- Tant que <vous lisez la série> faire EAQP ← 'écrire un algorithme qui permet de' ; ftq ;

Exercice 1 :EAQP réécrire un texte en majuscule.

Exercice 2 :EAQP déterminer le nombre de lettres distinctes entre deux mots donnés. On supposera, par exemple, que A et a correspondent à la même lettre.

Exercice 3 :EAQP associer a une chaîne ne contenant que des chiffres, le nombre entier correspondant.

Exercice 4 :EAQP déterminer le nombre de mots d'une phrase terminée par un point.

Exercice 5 :Étant donné deux chaînes. EAQP déterminer si la première est une sous-chaine de la deuxième. Exemple : 'bon' et 'jou' sont des sous-chaines de 'bonjour' mais 'ojo' et 'Jour' ne le sont pas.

Exercices supplémentaires

Exercice 6 : 1. EAQP déterminer si une phrase affirmative vérifie les règles de typographie :

- une phrase commence par une majuscule et se termine par un point;
- les mots sont séparés par exactement un espace sauf s'il y a une ponctuation,
- la virgule et le point sont collés au mot qui précède et sont suivis par un espace;
- le point virgule et les deux point sont précédés et suivis par un espace.

2. EAQP corriger une phrase affirmative pour qu'elle vérifie les précédentes règles.

Exercice 7 : Étant donné deux chaines. EAQP afficher les caractères de la première qui n'apparaissent pas dans la deuxième (dans l'ordre).

Exercice 8 : EAQP associer a une chaîne contenant une phrase avec ponctuation, un tableau de chaînes contenant les mots de cette phrase.

Exercice 9 : EAQP associer a un texte sans ponctuation son dictionnaire de mots. On utilisera pour le dictionnaire un tableau ordonné de chaînes contenant chacune un mot distinct de ce texte.

Exercice 10 : EAQP afficher pour une chaîne impaire un sablier et un nœud papillon comme suit :

```
bonjour      b      r
 onjou       bo    ur
  njo        bon our
   j         bonjour
  njo        bon our
 onjou       bo    ur
bonjour      b      r
```

Table ASCII

Dec	Char	Dec	Chr	Dec	Chr	Dec	Chr
0	NUL (null)	32	Space	64	@	96	`
1	SOH (start of heading)	33	!	65	A	97	a
2	STX (start of text)	34	"	66	B	98	b
3	ETX (end of text)	35	#	67	C	99	c
4	EOT (end of transmission)	36	$	68	D	100	d
5	ENQ (enquiry)	37	%	69	E	101	e
6	ACK (acknowledge)	38	&	70	F	102	f
7	BEL (bell)	39	'	71	G	103	g
8	BS (backspace)	40	(	72	H	104	h
9	TAB (horizontal tab)	41	)	73	I	105	i
10	LF (NL line feed, new line)	42	*	74	J	106	j
11	VT (vertical tab)	43	+	75	K	107	k
12	FF (NP form feed, new page)	44	,	76	L	108	l
13	CR (carriage return)	45	-	77	M	109	m
14	SO (shift out)	46	.	78	N	110	n
15	SI (shift in)	47	/	79	O	111	o
16	DLE (data link escape)	48	0	80	P	112	p
17	DC1 (device control 1)	49	1	81	Q	113	q
18	DC2 (device control 2)	50	2	82	R	114	r
19	DC3 (device control 3)	51	3	83	S	115	s
20	DC4 (device control 4)	52	4	84	T	116	t
21	NAK (negative acknowledge)	53	5	85	U	117	u
22	SYN (synchronous idle)	54	6	86	V	118	v
23	ETB (end of trans. block)	55	7	87	W	119	w
24	CAN (cancel)	56	8	88	X	120	x
25	EM (end of medium)	57	9	89	Y	121	y
26	SUB (substitute)	58	:	90	Z	122	z
27	ESC (escape)	59	;	91	[	123	{
28	FS (file separator)	60	<	92	\	124	|
29	GS (group separator)	61	=	93	]	125	}
30	RS (record separator)	62	>	94	^	126	~
31	US (unit separator)	63	?	95	_	127	DEL