Représentation des textes

Compter de A à Z

Nous avons vu qu'un ordinateur manipule des nombres, représentés en binaire. Nous avons vu comment représenter les nombres entiers, relatifs et même décimaux. Mais comment représenter les lettres ? Et les autres symboles permettant d'écrire un texte ?
La solution est simple : attribuer un numéro à chaque caractère. Mais quels caractères retenir ? Et quel numéro donner ? Pendant les années 50, il existait de multiples systèmes d'encodages rendant les communications compliquées entre systèmes (ordinateurs, imprimantes…). Pour remédier à ce problème, l'ANSI (*American National Standards Institute*) propose au début des années 60 la norme ASCII (*American Standard Code for Information Interchange*). Chacun des caractères est représenté par un octet, même si seulement 7 bits sont utilisés. Cela permet donc de représenter 128 caractères. Les numéros des caractères sont soit donnés en base 10 (0 à 127), soit en hexadécimal (00 à 7F). Le 8e bit est appelé **bit de parité**. Il est choisi de telle sorte qu'il y ait toujours un nombre pair de 1 dans l'octet. Cela permet de détecter des erreurs de transmission ou d'enregistrement.

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Les caractères sont séparés en caractères spéciaux servant, entre autre, pour des protocoles de communication (0 à 31), en chiffres (48 à 57), en lettres majuscules (65 à 90), en lettres minuscules (97 à 122) et en caractères de ponctuation. Ainsi, une chaîne de caractères est représentée par une suite de nombres.

B	o	n	j	o	u	r		a		t	o	u	s		!
66	111	110	106	111	117	114	32	97	32	116	111	117	115	32	33

Exercice : Décoder le texte suivant :

74	39	97	105	109	101	32	80	121	116	104	111	110	32	33

ISO-8859-n

Malheureusement, même si cet encodage peut sembler suffisant pour l'anglais, il ne l'est pas pour la plupart des autres langues. Il manque, notamment, les caractères accentués, sans parler des caractères d'autres alphabets (arabe, cyrillique, chinois, japonais…).
Pour permettre d'avoir plus de caractères disponibles, l'ISO (*Organisation Internationale de Normalisation*) a proposé la norme ISO-8859. L'idée est d'utiliser le 8^{e} bit pour coder plus de caractères. On peut donc représenter 256 caractères. Les 128 premiers correspondent aux symboles de la table ASCII. Néanmoins, 128 caractères supplémentaires ne sont pas suffisants pour tout représenter. C'est pourquoi il n'y a pas une mais seize tables qui sont proposées, notées ISO-8859-1 à ISO-8859-16. Certaines de ces tables ont un nom simplifié (latin-1 à latin-10).

Code ISO	Zone
8859-1 (latin-1)	Europe occidentale
8859-2 (latin-2)	Europe centrale ou de l'est
8859-3 (latin-3)	Europe du sud
8859-4 (latin-4)	Europe du nord
8859-5	Cyrillique
8859-6	Arabe
…	…
8859-15 (latin-9)	Révision du latin-1 avec le symbole €
8859-16 (latin-10)	Europe du sud-est

Néanmoins, avec ces tables, il n'est toujours pas possible d'écrire en chinois ou en japonais. Et il est compliqué d'écrire en plusieurs langues dans le même document.

L'Unicode

Afin de régler les problèmes d'encodages qui persistaient, l'ISO a défini un jeu universel de caractères appelé UCS (*Universal Character Set*), aussi appelé ISO-10646. Chaque caractère (lettre, symbole, idéogramme, emoji…) est associé à un nom unique (en anglais et en français) et à un entier positif en base 10 appelé **point de code**. Il y a environ 150 000 caractères recensés dans cette norme, qui devrait contenir tous les symboles nécessaires pour toutes les langues existantes. La capacité maximale est fixée à 4 294 967 295 caractères, c'est-à-dire le maximum pouvant être représenté sur 32 bits. Les 256 premiers points de code correspondent à l'ISO-8859-1.
On note en général U+xxxx les points de code, où les x représentent des chiffres en hexadécimal. Si c'est nécessaire, il est possible de rajouter des chiffres, mais il en faut au moins 4. Cette norme pose le problème de comporter beaucoup de 0 inutiles dans le cas d'un texte avec des caractères des tables ASCII ou latin-1. C'est pourquoi le consortium Unicode a proposé des techniques d'encodage. La plus utilisée est l'UTF-8. L'idée est de pouvoir représenter les points de code sur un nombre variable d'octets, allant de 1 à 4. Ainsi lorsqu'on se restreint à l'ASCII, on n'utilise qu'un octet par caractère.

Plage	Suite d'octets (en binaire)	bits significatifs
U+0000 à U+007F	0xxxxxxx	7
U+0080 à U+07FF	110xxxxx 10xxxxxx	11
U+0800 à U+FFFF	1110xxxx 10xxxxxx 10xxxxxx	16
U+10000 à U+10FFFF	11110xxx 10xxxxxx 10xxxxxx 10xxxxxx	21

L'UTF-16 utilise 2 ou 4 octets. Enfin, l'UTF-32 utilise 4 octets pour chaque caractère, ce qui est plus pratique pour la gestion des chaînes de caractères, mais bien plus gourmand en espace mémoire.