Le codage de l'information : Introduction

Qu'est-ce que l'informatique ?

- Le mot informatique est un mot formé à partir de deux autres mots : **Informa**tion et auto**matique**. L'informatique désigne donc l'automatisation du traitement de l'information, le plus souvent, par un système, concret (ordinateur, tablette, portable, GPS…).
- Dans son acceptation courante, l'informatique désigne l'ensemble des **Sciences et techniques** en rapport avec le **traitement de l'information**
- Dans le parler populaire, l'informatique peut aussi désigner ce qui se rapporte au matériel informatique (l'électronique), ou la bureautique.

Internet, mail, WWW, …
Outils bureautiques (Traitements de texte, tableurs, …)
Utilisation de logiciels scientifiques (Bio-informatique, visualisation de données)
Développement de logiciels
Programmation

Difficulté croissante ↓

Présentation succinte de l'ordinateur

Un **ordinateur** est un ensemble de **composants électroniques modulaires**, c'est-à-dire des composants pouvant être remplacés par d'autres composants ayant éventuellement des caractéristiques différentes, capables de faire fonctionner **des programmes informatiques**. On parle ainsi de « **hardware** » pour désigner l'ensemble des éléments **matériels** de l'ordinateur et de « **software** » pour désigner la partie **logicielle**.

Chaque élément est associé à une fonction:

Ecran (« voir »), Disque Dur ou DD (Hard Disc ou HD) (« écrire et lire »), souris (« faire des choix »), RAM (« mémoire »), lecteurs (DVD, CD, clé USB « retrouver des données»…), processeur (« calculer»), clavier (« taper des mots »), etc. … (Imprimante, carte mère, joystick, scanner, bus, modem….).

Notion clef en informatique : « l'information »

Récapitulatif de la liste précédente vers quelques fonctions essentielles autour de cette notion:

Echanger de l'information (clavier, souris, bus, écran, imprimante, modem…).
Mémoriser de l'information (DD, RAM, ROM, DVD, CD, clé USB, …).
Calculer de l'information (processeur).

Les différentes mémoires

Quels types de mémoire trouve-t-on dans un ordinateur ?

Qu'est ce qui les différencie ?

Mémoire	Principe	Maintien	Rôle	Taille	Vitesse
ROM (read only memory)	Gravée dans un circuit en dur	Tout le temps (gravée)	Lancer le système d'exploitation	n Ko	
Disque Dur	Magnétique	Tout le temps (stable sans énergie)	Stockage massif à long terme	n Go – 4 To	Lente (accès disque)
RAM (random acces memory)	Electrique	Uniquement sous tension	Mémoire de travail des données et instructions en cours	128 Mo –64 Go	Rapide
Cache	Electrique	Uniquement sous tension	Idem RAM mais optimisé	100 Ko – 64 Mo	Très rapide

Le Processeur

Quel est le rôle du processeur ? (Intel Pentium, AMD Athlon, Dual core , multi cœurs etc…)
Calculer de nouvelles données (informations) avec des données (registres de données) et des instructions (registres d'instructions). Les instructions commandent des portes logiques (des formes d'aiguillages) qui déterminent comment sont combinées les données entre elles.

Les systèmes d'exploitation

C'est le lien entre la machine et nous ; Il existe différents SE (OS) : Windows, Linux, MacOs, Unix, …
Un SE est un programme de base permettant de gérer :

1. la mémoire et le processeur ;
2. l'exécution des autres programmes ;
3. la communication interne (le bus) ;
4. les périphériques externes et aussi le graphisme ;
5. l'accessibilité de l'information aux utilisateurs.

Windows est à la fois un système d'exploitation et une interface graphique.

Les fichiers :

Le disque dur d'un ordinateur conserve toutes les informations (textes, figures, films et programmes) dans des fichiers. Afin de comprendre un peu mieux comment tout cela marche, quelques explications sont nécessaires.

Contenu et représentation d'un fichier

Il est important de comprendre la différence entre
• Le contenu d'un fichier
• La représentation d'un fichier
Le contenu d'un fichier est ce qui est vraiment écrit sur le disque.
La représentation d'un fichier c'est son affichage à l'écran au moyen d'un utilitaire. Le même fichier peut s'afficher différemment suivant l'utilitaire utilisé. C'est ce que nous allons voir.

Codages binaire et hexadécimal

Sur le disque un fichier est enregistré sous forme binaire : une série de zéros et de uns (001010101010101). Comme cette notation est absolument impossible à lire pour nous et prendrait surtout une place trop importante, ce codage est généralement transformé en hexadécimal. Dans la vie de tous les jours nous utilisons le système décimal, c'est à dire un codage avec dix chiffres de 0 à 9. Dans les ordinateurs, tout est stocké sous forme élémentaire dans des bytes (octets de 8bits) : une série de huit chiffres à la suite (soit zéro soit un) ; Exemple : 0100 0010 est la façon dont est codé dans votre machine la lettre "B". Il est bien évidemment impossible pour un œil humain de lire la chaine de zéros et de uns. Pour pouvoir lire ces valeurs plus facilement elles sont souvent transformées en hexadécimal.
L'hexadécimal est un codage sur 16 caractères de 0 à 9 puis de A à F. Le nombre décimal 0 reste 0, 5 reste 5, 9 reste 9 ; 10 devient donc A et 15 devient F.
Le mot binaire 1111 se code donc F (8+4+2+1)

Pourquoi l'hexadécimal ?

Parce que cela permet de transformer un byte (octet) en deux lettres d'où un gain de place dans l'écriture : le codage hexadécimal sur quatre chiffres permet de coder seize possibilités différentes. Donc le nombre hexa 3F7A se traduit en binaire : 011 1111 0111 1010.

Correspondances binaire, décimal, hexadécimal :

Binaire	Décimal	Hexadécimal	Binaire	Décimal	Hexadécimal
0000	0	0	1010	10	A
0001	1	1	1011	11	B
0010	2	2	1100	12	C
0011	3	3	1101	13	D
0100	4	4	1110	14	E
0101	5	5	1111	15	F
0110	6	6	10000	16	10
0111	7	7	10001	17	11
1000	8	8	etc	etc	etc
1001	9	9			

voir le cours sur les changements de bases

Visualisation d'un fichier :

Ouvert avec not pad ce fichier est

un peu plus abordable

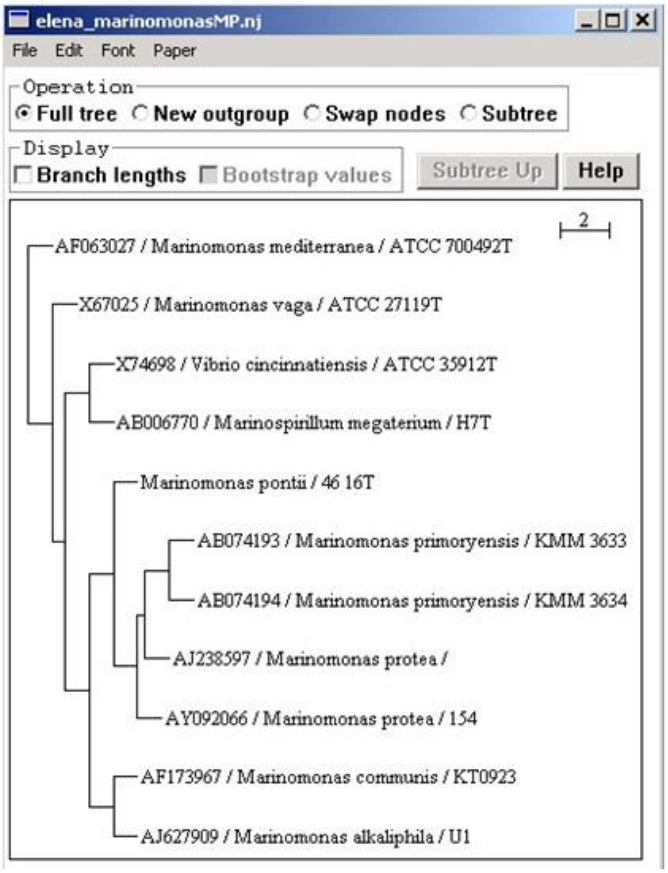
```
(((((AJ627909 / Marinomonas
alkaliphila / U1:1.0,AF173967
Marinomonas communis / KT0923:1.0):1.0,((AY092066
/ Marinomonas protea
/ 154:1.0,(AJ238597 / Marinomonas protea /
:1.0,(AB074194 / Marinomonas primoryensis / KMM
3634:1.0,AB074193 / Marinomonas primoryensis /
KMM
3633:1.0):1.0):0.3):1.0,Marinomonas pontii / 46
16T:1.0):1.0):1.0,(AB006770
/ Marinospirillum megaterium
/ H7T:1.0,X74698 / Vibrio
cincinnatiensis / ATCC
35912T:1.0):1.0):0.5,X67025 / Marinomonas vaga /
ATCC
27119T:1.0):1.0,AF063027 / Marinomonas mediterranea /
ATCC 700492T:1.0);
```

En hexadécimal (ouvert avec hex éditor)

```
28 28 28 28 28 41 4A 36 32 37 39 30 39 20 2F 20 ; (((((AJ627909 /
4D 61 72 69 6E 6F 6D 6F 6E 61 73 20 61 6C 6B 61 ; Marinomonas alka
6C 69 70 68 69 6C 61 20 2F 20 55 31 3A 31 2E 30 ; liphila / U1:1.0
2C 41 46 31 37 33 39 36 37 20 2F 20 4D 61 72 69 ; ,AF173967 / Mari
6E 6F 6D 6F 6E 61 73 20 63 6F 6D 6D 75 6E 69 73 ; nomonas communis
20 2F 20 4B 54 30 39 32 33 3A 31 2E 30 29 3A 31 ;  / KT0923:1.0):1
2E 30 2C 28 28 41 59 30 39 32 30 36 36 20 2F 20 ; .0,((AY092066 /
4D 61 72 69 6E 6F 6D 6F 6E 61 73 20 70 72 6F 74 ; Marinomonas prot
65 61 20 2F 20 31 35 34 3A 31 2E 30 2C 28 41 4A ; ea / 154:1.0,(AJ
32 33 38 35 39 37 20 2F 20 4D 61 72 69 6E 6F 6D ; 238597 / Marinom
6F 6E 61 73 20 70 72 6F 74 65 61 20 2F 20 3A 31 ; onas protea / :1
2E 30 2C 28 41 42 30 37 34 31 39 34 20 2F 20 4D ; .0,(AB074194 / M
61 72 69 6E 6F 6D 6F 6E 61 73 20 70 72 69 6D 6F ; arinomonas primo
72 79 65 6E 73 69 73 20 2F 20 4B 4D 4D 20 33 36 ; ryensis / KMM 36
33 34 3A 31 2E 30 2C 41 42 30 37 34 31 39 33 20 ; 34:1.0,AB074193
2F 20 4D 61 72 69 6E 6F 6D 6F 6E 61 73 20 70 72 ; / Marinomonas pr
69 6D 6F 72 79 65 6E 73 69 73 20 2F 20 4B 4D 4D ; imoryensis / KMM
20 33 36 33 33 3A 31 2E 30 29 3A 31 2E 30 29 3A ;  3633:1.0):1.0):
30 2E 33 29 3A 31 2E 30 2C 4D 61 72 69 6E 6F 6D ; 0.3):1.0,Marinom
6F 6E 61 73 20 70 6F 6E 74 69 69 20 2F 20 34 36 ; onas pontii / 46
20 31 36 54 3A 31 2E 30 29 3A 31 2E 30 29 3A 31 ;  16T:1.0):1.0):1
2E 30 2C 28 41 42 30 30 36 37 37 30 20 2F 20 4D ; .0,(AB006770 / M
61 72 69 6E 6F 73 70 69 72 69 6C 6C 75 6D 20 6D ; arinospirillum m
65 67 61 74 65 72 69 75 6D 20 2F 20 48 37 54 3A ; egaterium / H7T:
31 2E 30 2C 58 37 34 36 39 38 20 2F 20 56 69 62 ; 1.0,X74698 / Vib
72 69 6F 20 63 69 6E 63 69 6E 6E 61 74 69 65 6E ; rio cincinnatien
73 69 73 20 2F 20 41 54 43 43 20 33 35 39 31 32 ; sis / ATCC 35912
54 3A 31 2E 30 29 3A 31 2E 30 29 3A 30 2E 35 2C ; T:1.0):1.0):0.5,
58 36 37 30 32 35 20 2F 20 4D 61 72 69 6E 6F 6D ; X67025 / Marinom
6F 6E 61 73 20 76 61 67 61 20 2F 20 41 54 43 43 ; onas vaga / ATCC
20 32 37 31 31 39 54 3A 31 2E 30 29 3A 31 2E 30 ;  27119T:1.0):1.0
2C 41 46 30 36 33 30 32 37 20 2F 20 4D 61 72 69 ; ,AF063027 / Mari
6E 6F 6D 6F 6E 61 73 20 6D 65 64 69 74 65 72 72 ; nomonas mediterr
61 6E 65 61 6E 20 2F 20 41 54 43 43 20 37 30 30 34 ; anea / ATCC 7004
39 32 54 3A 31 2E 30 29 3B 0D 0A                ; 92T:1.0);..
```

Conclusion : ce fichier est incompréhensible

Finalement ouvert avec l'application adéquate ce fichier devient plus lisible



Cette fois ci cela paraît plus compréhensible !

Il s'agit tout simplement d'un arbre qui représente les parentés entre certaines espèces de bactéries

Les flèches mettent en évidence des endroits où se situent la présence d'ancêtres communs à certaines espèces.

Résumons la situation !

Il s'agit bel et bien du même fichier, mais ouvert avec des logiciels différents.

Ce qu'il faut retenir :

- Un fichier enregistré sur le disque n'est qu'une suite de 0 et 1.

- C'est le logiciel utilisé pour ouvrir ce fichier qui va le "traduire" afin de l'afficher sous un certain format. L'extension du fichier (.txt, .doc, .pdf) n'est qu'une indication qui permet d'associer un utilitaire (par défaut) pour ouvrir ce fichier.

Si vous cliquez sur un fichier .txt, windows associe par défaut le notpad (bloc note) pour l'ouvrir.
Si vous cliquez sur un fichier .doc, windows associe par défaut MS Word pour l'ouvrir.

Le langage binaire

Vers la fin des années 30, Claude Shannon démontra qu'à l'aide de « contacteurs » (interrupteurs) fermés pour « vrai » et ouverts pour « faux ». Il était possible d'effectuer des opérations logiques en associant le nombre « un pour vrai « et « zéro pour faux ».
Ce codage de l'information est nommé base binaire. C'est avec ce codage que fonctionnent les ordinateurs. Il consiste à utiliser 2états (représentés par 0 et 1) pour coder les informations. L'homme calcule depuis 2000 ans avant Jésus-Christ avec 10 chiffres (0, 1, 2, 3, 4, 5, 6, 9), on parle alors de base décimale (ou base 10). Toutefois dans des civilisations anciennes ou pour certaines applications actuelles d'autres bases de calcul ont et sont toujours utilisées : base **sexagésimale** (60), utilisée par les Sumériens. Cette base est également utilisée dans le système horaire actuel, pour les minutes et les secondes ; base **vicésimale** (20), utilisée par les Mayas ; base **duodécimale** (12), utilisée par les anglo-saxons dans leur système monétaire jusqu'en 1960 : un « pound » représentait vingt « shilling » et un « shilling » représentait douze « pences ». Le système d'heure actuel fonctionne également sur douze heures (notamment dans la notation anglo-saxonne) ; base **quinaire** (5), utilisée par les Mayas; base **binaire** (2), utilisée par l'ensemble des technologies numériques.

Le Bit :

Le terme bit (b avec une minuscule dans les notations) signifie « binary digit », c'est-à-dire 0 ou 1 en numérotation binaire. Il s'agit de la plus petite unité d'information manipulable par une machine numérique. Il est possible de représenter physiquement cette information binaire : par un signal électrique ou magnétique qui au-delà d'un certain seuil correspond à la valeur 1 ; Avec un bit il est ainsi possible d'obtenir deux états : soit 1 soit 0 ; grâce à 2 bits, il est possible d'obtenir 4 états différents 2^2
Avec 3 bits, il est possible d'obtenir huit états différents (2x2x2) ou 2^3 ;
Avec n bits il est possible de représenter 2^n valeurs.

Poids des bits : Dans un nombre binaire, la valeur d'un bit, appelée poids, dépend de la position du bit en partant de la droite. A la manière des dizaines, des centaines et des milliers pour un nombre décimal, le poids croit d'une puissance de 2 en allant de la droite vers la gauche comme le montre le tableau suivant :

Nombre binaire	1	1	1	1	1	1	1	1
Poids	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$

Conversions : Pour convertir un mot binaire en nombre décimal, il suffit de multiplier la valeur de chaque bit par son poids puis d'additionner chaque résultat : ainsi le mot 0101 vaut en décimal :
$2^3x0 + 2^2x1 + 2^1x0 + 2^0x1 = 8x0 + 4x1 + 2x0 + 1x1 = 5$

L'octet (Byte)

L'octet (en anglais byte ou B avec une majuscule dans les notations) est une unité d'information composée de 8 bits. Il permet par exemple de stocker un caractère, tel qu'une lettre ou un chiffre. Ce regroupement de nombres par série de 8 permet une lisibilité plus grande, au même titre que l'on apprécie, en base décimale, de regrouper les nombres par trois pour pouvoir distinguer les milliers. Le nombre « 1 256 245 » est par exemple plus lisible que « 1256245 ». Une unité d'information composée de 16 bits est généralement appelée mot (en anglais word). Une unité d'information de 32 bits de longueur est appelée mot double (en anglais double word, d'où l'appellation dword). Pour un octet, le plus petit nombre est 0 représenté par 8 zéros (00000000) et le plus grand et 255 représenté par 8 uns (11111111), ce qui représente 256 possibilités de valeurs différentes ;

2^7 =128	2^6 =64	2^5 =32	2^4 =16	2^3 =8	2^2 =4	2^1 =2	2^0 =1
0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1

KiloOctets , MégaOctets GigaOctets

Longtemps l'informatique s'est singularisée par l'utilisation de différentes valeurs pour les unités du système international. Ainsi beaucoup d'informaticiens ont appris que 1 kilooctet valait 1024 octets. Or, depuis décembre 1998, l'organisme international IEC a statué sur la question (http://physics.nist.gov/cuu/Units/binary.html) voici donc les unités standardisées :
1kilooctet (ko ou kB) = 1000 octets ;
1Mégaoctet (Mo ou MB) = 1000 ko = 1 000 000 octets ;
1Gigaoctet (Go ou GB) = 1000 Mo = 1 000 000 000 octets ;
1Téraoctet (To) = 1000 Go = 1 000 000 000 000 octets.
Attention ! De nombreux logiciels (parfois même certains systèmes d'exploitation) utilisent

toujours la notation antérieure à 1998 pour laquelle :

1kilooctet (ko) = 2^7 x 1000 = 2^{10} octets = 1024 octets

1Mégaoctet (Mo) = 2^{20} octets = 1024 ko = 1 048 576 octets

1Gigaoctet (Go) = 2^{30} octets = 1024 Mo = 1 073 741 824 octets

1Téraoctet (To) = 2^{40} octets = 1024 Go = 1 099 511 627 776 octets

Il est également utile de noter que la communauté internationale dans son ensemble utilise préférentiellement le nom de « byte » plutôt que le terme « octet » purement francophone. Cela donne les notations suivantes pour kilobyte, mégabyte, gigabyte et terabyte : kB, MB, GB, TB

Notez l'utilisation d'un B majuscule pour différencier Byte et bit.

Transcodages

Il peut être utile de savoir passer d'une base à une autre en ayant recours à un système de transcodage, c'est-à-dire un système qui va permettre de convertir du binaire en hexadécimal ou du décimal en binaire. C'est ce que nous allons voir dans le TP « Algorithmes de Transcodages ».

Opérations binaires

Les opérations arithmétiques simples telles que l'addition, la soustraction et la multiplication sont faciles à effectuer en binaire :

L'addition en binaire L'addition en binaire se fait avec les mêmes règles qu'en décimal :
On commence à additionner les bits de poids faible (les bits de droite) puis on a des retenues lorsque la somme de deux bits de même poids dépasse la valeur de l'unité la plus grande (dans le cas du binaire 1), cette retenue est reportée sur le bit de poids plus fort suivant … par exemple :

	0	1	1	0	1
+	0	1	1	1	0
	-	-	-	-	-
	1	1	0	1	1

Multiplication binaire

0x0=0
0x1=0
1x0=0
1x1=1

La multiplication se fait en formant un produit partiel pour chaque digit du multiplicateur (seuls les bits non nuls donneront un résultat non nul). Lorsque le bit du multiplicateur est nul, le produit partiel est nul, lorsqu'il vaut un, le produit partiel est constitué du multiplicande décalé du nombre de positions égal au poids du bit du multiplicateur. Par exemple :

		0	1	0	1	
x		0	0	1	0	
	-	-	-	-	-	-
		0	0	0	0	
	0	1	0	1		
0	0	0	0			
-	-	-	-	-	-	-
	0	1	0	1	0	

Représentation d'un nombre sur un ordinateur

On appelle représentation (ou codification) d'un nombre la façon selon laquelle il est décrit sous forme binaire. La représentation des nombres sur un ordinateur est indispensable pour que celui- ci puisse les stocker, les manipuler. Toutefois le problème est qu'un nombre mathématique peut être infini (aussi grand que l'on veut), mais la représentation d'un nombre dans un ordinateur doit être faite sur un nombre de bits prédéfini. Il s'agit donc de prédéfinir un nombre de bits et la manière de les utiliser pour que ceux-ci servent le plus efficacement possible à représenter l'entité. Ainsi il serait idiot de coder un caractère sur 16 bits (65536 possibilités) alors qu'on en utilise généralement moins de 256...

Entier naturel, entier relatif, nombre réel :

Le choix à faire (c'est-à-dire le nombre de bits à utiliser) dépend de la fourchette des nombres que l'on désire utiliser. Pour coder des nombres entiers naturels compris entre 0 et 255, il nous suffira de 8 bits (un octet) car 2^8=256. D'une manière générale un codage sur n bits pourra permettre de représenter des nombres entiers naturels compris entre 0 et 2^{n-1}. Les nombres réels seront codés sur plus de bits.

Le code ASCII

La mémoire de l'ordinateur conserve toutes les données sous forme numérique. Il n'existe pas de méthode pour stocker directement les caractères. Dans les années 60, le code ASCII (American Standard Code for Information Interchange) est adopté comme standard. Chaque caractère possède donc son équivalent en code numérique. Il permet le codage de caractères sur 8bits soit 256 caractères possibles

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	Null	32	20	Space	64	40	@	96	60	`
1	01	Start of heading	33	21	!	65	41	A	97	61	a
2	02	Start of text	34	22	"	66	42	B	98	62	b
3	03	End of text	35	23	#	67	43	C	99	63	c
4	04	End of transmit	36	24	$	68	44	D	100	64	d
5	05	Enquiry	37	25	%	69	45	E	101	65	e
6	06	Acknowledge	38	26	&	70	46	F	102	66	f
7	07	Audible bell	39	27	'	71	47	G	103	67	g
8	08	Backspace	40	28	(	72	48	H	104	68	h
9	09	Horizontal tab	41	29	)	73	49	I	105	69	i
10	0A	Line feed	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage return	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	46	2E	.	78	4E	N	110	6E	n
15	0F	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	Neg. 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	End 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	Substitution	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	□

Le code ASCII n'est pas unique et dépend fortement de la plateforme utilisée

La norme ASCII permet ainsi à toutes sortes de machines de stocker, analyser et communiquer de l'information textuelle. En particulier, la quasi-totalité des ordinateurs personnels et des stations de travail utilisent l'encodage ASCII. Le code ASCII de base représentait les caractères sur 7 bits (c'est-à-dire 128 caractères possibles, de 0 à 127).

Les codes 0 à 31 ne sont pas des caractères. On les appelle caractères de contrôle car ils permettent de faire des actions telles que : retour à la ligne (Carriage return) bip sonore (Audible bell)
Les codes 65 à 90 représentent les majuscules

Les codes 97 à 122 représentent les minuscules (il suffit de modifier le 6ème bit pour passer de majuscules à minuscules, c'est-à-dire ajouter 32 au code ASCII en base décimale).

Le code ASCII a été mis au point pour la langue anglaise, il ne contient donc pas de caractères accentués, ni de caractères spécifiques à une langue. Le code ASCII a donc été étendu à 8 bits pour pouvoir coder plus de caractères (on parle d'ailleurs de code ASCII étendu...). Cette norme s'appelle ISO-8859 et se décline par exemple en ISO-8859-1 lorsqu'elle étend l'ASCII avec les caractères accentués d'Europe occidentale, et qui est souvent appelée Latin-1 ou Europe occidentale.
Il existe d'autres normes que l'ASCII, comme l'Unicode par exemple, qui présente l'avantage de proposer une version unifiée des différents encodages de caractères complétant l'ASCII mais aussi de permettre l'encodage de caractères autres que ceux de l'alphabet latin. Unicode définit des dizaines de milliers de codes, mais les 128 premiers restent compatibles avec ASCII.
Dans le codage UTF-8, chaque point de code est codé sur une suite d'un à quatre octets. Il a été conçu pour être compatible avec certains logiciels originellement prévus pour traiter des caractères d'un seul octet.