Encodage du texte

• ASCII

i En 1960, le développement de l'informatique est essentiellement anglophone. A cette époque, le codage `ASCII`, pour American Standard Code for Information Interchange est crée pour écrire des textes en anglais.
Cette norme ne définissait que $128 = 2^7$ codes.
Ils sont en fait représentés par mots de huit bits (un octet), le premier étant toujours un zéro.

La table `ASCII` ci-dessous fournit la correspondance entre $128 = 2^7$ caractères et leur représentation binaire. 95 caractères sont imprimables :

- les chiffres de 0 à 9,
- les lettres minuscules de a à z et les majuscules de A à Z,
- des symboles mathématiques et de ponctuation

Voir la table `ASCII`

Exercice 8

Trouver la représentation binaire en ASCII du texte

`Salut !`

Exercice 9 ★

Décoder le texte représenté en ASCII binaire par la suite de bits :

01000011001001110110010101110011 0
01000010000001100110011000010110001 1
011010010110110001100101

Exercice 10

Trouver la représentation binaire en ASCII du texte

`«Je pense, donc je suis.»`

? Comment coder ce message le message `Bonjour à tous.` ?

• Norme ISO-8859

La nécessité de représenter des textes comportant des caractères non présents dans la table ASCII tels ceux de l'alphabet latin utilisés en français comme le 'à', le 'é' ou le 'ç' impose l'utilisation d'un autre codage que l'ASCII.
Afin de faciliter les choses, ces propositions sont des extensions du codage ASCII :

1. le codage des caractères présents dans la table ASCII est conservé ;
2. le principe du codage de chacun des caractères sur un octet est conservé.

Mais les 8 bits de l'octet vont être utilisés. Cela permet de coder $2^8 = 256$ caractères, soit 128 caractères supplémentaires.
L'ISO, organisation internationale de normalisation, propose de son côté plusieurs variantes de codages adaptées aux différentes langues. La plus utilisée concerne les langues européennes occidentales. Il s'agit de l'ISO-8859-1
Mais il existe d'autres variantes adaptées à d'autres langues :

— ISO-8859-2 pour les pays d'Europe de l'est,

— ISO-8859-3 pour les pays du sud est de l'Europe...

• UTF-8

UTF = UCS Transformation Format *(UCS = Universal Character Set, norme ISO-10646)*
Le numéro de chaque caractère est donné par le standard Unicode.
Les caractères de numéro 0 à 127 sont codés sur un octet dont le bit de poids fort est toujours nul.
Les caractères de numéro supérieur à 127 sont codés sur plusieurs octets. Dans ce cas, les bits de poids fort du premier octet forment une suite de 1 de longueur égale au nombre d'octets utilisés pour coder le caractère, les octets suivants ayant 10 comme bits de poids fort.
Définition dun nombre d'octets utilisés

Représentation binaire UTF-8	Signification
0xxxxxxx	1 octet codant 1 à 7 bits
110xxxxx 10xxxxxx	2 octets codant 8 à 11 bits
1110xxxx 10xxxxxx 10xxxxxx	3 octets codant 12 à 16 bits
11110xxx 10xxxxxx 10xxxxxx 10xxxxxx	4 octets codant 17 à 21 bits

Caractère	Numéro du caractère	Codage binaire UTF-8
A	65	01000001
é	233	11000011 10101001
€	8364	11100010 10000010 10101100
𝄞	119070	11110000 10011101 10000100 10011110

Voir la table des codes https ://unicode-table.com/fr/cjk-unified-ideographs-extension-a

? Quel encodage choisir ?

Sans hésitation il faut choisir systématiquement l'encodage UTF-8.
UTF-8 est en effet un codage de caractères conçu pour coder l'ensemble des caractères Unicode, tout en restant compatible avec la norme ASCII.
C'est devenu l'encodage par défaut sur Python3 et c'est aussi l'encodage le plus utilisé sur le Web.

• Python & les chaines de caractères

En Python les chaines de caractères sont des objets de type **`str`** (strings). Ces objets sont des itérables qui possèdent des méthodes qu'il est intéressant de connaître.
Comme il s'agit de méthodes appliquées à l'objet str, on appelle ces méthodes en mettant un point après la variable string.
instancie la variable chaine de type "str" attention il ne faut pas d'apostrophe dans la chaine

```
1 chaine = 'Bonjour à tous'
```

instancie la variable chaine de type "str"

```
1 chaine = "Bonjour à tous"
```

idem (permet le retour à la ligne)

```
1 chaine = '''Bonjour à tous'''
```

idem (permet le retour à la ligne)

```
1 chaine = """Bonjour à tous"""
```

`chaine.upper()`	renvoi la chaîne en majuscule
`chaine.lower()`	renvoi la chaîne en minuscule
`chaine.capitalize()`	renvoi la chaîne capitalizer (première lettre de chaque mot en majuscule)
`chaine.find("jo")`	renvoie l'indice du début de trouvé ou -1 si pas trouvé
`chaine.replace("jo" , "oj")`	remplace toutes les occurrences de "jo" par "oj"
`chaine.replace("jo" , "oj", 1)`	remplace la première occurrence de "jo" par "oj"
`chaine.count("jo")`	compte le nombre d'occurrences de "jo"

Les autres bases

• Écriture en base 16 (hexadécimal)

Pourquoi l'hexadécimal

> Le codage hexadécimal est particulièrement commode car il permet un compromis entre le code binaire des machines et une base de numération pratique à utiliser pour les ingénieurs. En effet, chaque chiffre hexadécimal correspond exactement à quatre chiffres binaires (ou bits), rendant les conversions très simples et fournissant une écriture plus compacte.
> L'hexadécimal a été utilisé la première fois en 1956 par les ingénieurs de l'ordinateur Bendix G-15.
>
> *source : Wikipédia*

Pour écrire un nombre en base 16, il faut disposer d'un caractère pour chacun des entiers de 0 à 15. Or, on ne dispose pas d'assez de chiffres pour écrire les 16 chiffres de la base 16. On complète donc les chiffres de 0 à 9 par les six premières lettres de l'alphabet : A, B, C, D, E et F.

Base 10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	...
Base 16	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	...

Notation

Il existe plusieurs notations pour les nombres codés en hexadécimal :

$$A2E_{16} \Leftrightarrow 0xA2E \Leftrightarrow A2E_{\text{hex}}$$

,

• Conversion binaire vers hexadécimal

Exemple

$b_2 \Rightarrow b_{16}$

Quelle est la représentation du nombre $110100111000100_{\overline{2}}$ en hexadécimal ?

- On sépare en bloc de 4 bits

$$110\ 1001\ 1100\ 0100$$

- On converti chacun des blocs :

$$\underbrace{0110}_{6}\underbrace{1001}_{9}\underbrace{1100}_{C}\underbrace{0100}_{4}$$

On a donc :

$110100111000100_{\overline{2}} = 69C4_{\overline{16}}$

• Conversion hexadécimal vers binaire

Exemple

$b_{16} \Rightarrow b_2$

Quelle est la représentation du nombre $56E4_{\overline{16}}$ en binaire ?

- On converti chaque digit en binaire sur 4 bits :

$$\underbrace{5}_{0101}\underbrace{6}_{0110}\underbrace{E}_{1110}\underbrace{4}_{0100}$$

On a donc :

$56E4_{\overline{16}} = 0101011011100100_{\overline{2}}$

Comme souvent on enlevera les zéros non significatif au début du nombre :

$56E4_{\overline{16}} = 101011011100100_{\overline{2}}$