

# Chapitre 3 : Représentation d'un texte en machine

## Comment sont représentées et encodées les lettres en machine ?

### Activité 1 : Recherche historique

Sujet : Faire une recherche documentaire permettant de traiter du sujet suivant :

Donner les caractéristiques et les raisons de leurs mise en place, pour le code ASCII, la norme ISO 8859-1 puis la norme l'UNICODE (avec le codage UTF-8)

Consignes :

Pour l'**oral** : Votre oral devra traiter de l'ensemble du sujet ci-dessus (vous pouvez vous aider des pistes de travail). Cet oral devra s'appuyer sur un support visuel (afin de faciliter le suivi de votre présentation pour vos camarades) qui comportera mots clés et illustrations. Il devra durer environ 5 minutes et pourra se poursuivre par un temps de questions. Vous devrez présenter vos sources qui vous auront permis de réaliser ce travail.

Pour la **trace écrite** : Vous devez faire un bilan de l'ensemble des informations que vous avez pu recueillir lors de vos recherches. Ce travail doit tenir sur une page maximum et être organisé avec un plan.

Pistes de travail :

- Expliquer l'origine de la mise en place de ces différentes normes
- Présenter ce qui caractérise chacune de ces normes
- Donner les atouts et les limites de chacun des exemples.

**Exercice 1 : Décoder les caractères ASCII**

La table ci-dessous donne le code associé à chacun des caractères ASCII imprimables. Le code du caractère en hexadécimal s'obtient en écrivant le numéro de la ligne puis le numéro de la colonne.

Ex : la lettre M a pour code hexadécimal 4D16

 Les cases vides correspondent à des caractères non imprimables, comme les caractères de contrôle.

|   | 0      | 1 | 2 | 3 | 4  | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|--------|---|---|---|----|---|---|---|---|---|---|---|---|---|---|---|
| 0 |        |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |
| 1 |        |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |
| 2 | espace | ! | " | # | \$ | % | & | ' | ( | ) | * | + | , | - | . | / |
| 3 | 0      | 1 | 2 | 3 | 4  | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 4 | @      | A | B | C | D  | E | F | G | H | I | J | K | L | M | N | O |
| 5 | P      | Q | R | S | T  | U | V | W | X | Y | Z | [ | \ | ] | ^ | _ |
| 6 | `      | a | b | c | d  | e | f | g | h | i | j | k | l | m | n | o |
| 7 | p      | q | r | s | t  | u | v | w | x | y | z | { |   | } | ~ |   |

Table de correspondance entre un code et son caractère ASCII imprimable

- Donner le nom correspondant à la liste des codes ASCII suivants donnés en hexadécimal : 47 65 6F 72 67 65 20 42 6F 6F 6C 65

George Boole

- À l'aide de ce même tableau, écrire votre prénom en hexadécimal.

4D 61 78 65 6E 63 65

**Exercice 2 : Décoder les caractères UTF-8**

En UTF-8, le codage des caractères coïncide avec l'ASCII pour les 128 premiers caractères. Les autres caractères sont représentés par plusieurs octets.

On a retrouvé la série d'octets suivants (en hexadécimal) dans un fichier codé en UTF-8.

43 6F 64 C3 A9 20 65 6E 20 55 54 46 2D 38

Il contient uniquement des caractères de la table ASCII à l'exception d'un « é ».

- À l'aide de la table ci-dessus, écrire le texte affiché si le logiciel décode cette suite d'octets en supposant que la norme est UTF-8.

Expliquer pourquoi ce message n'apparaît pas correctement.

Codé en UTF-8

- Réécrire la suite d'octet pour qu'elle soit bien lisible avec la norme UTF-8.

43 6F 64 65 20 65 6E 20 55 54 46 2D 38

 <http://www.science-phynum.fr/>

|   | 0      | 1 | 2 | 3 | 4  | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|--------|---|---|---|----|---|---|---|---|---|---|---|---|---|---|---|
| 2 | espace | ! | " | # | \$ | % | & | ' | ( | ) | * | + | , | - | . | / |
| 3 | 0      | 1 | 2 | 3 | 4  | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 4 | @      | A | B | C | D  | E | F | G | H | I | J | K | L | M | N | O |
| 5 | P      | Q | R | S | T  | U | V | W | X | Y | Z | [ | \ | ] | ^ | _ |
| 6 |        | a | b | c | d  | e | f | g | h | i | j | k | l | m | n | o |
| 7 | p      | q | r | s | t  | u | v | w | x | y | z | { |   | } | ~ |   |
| 8 |        |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |
| 9 |        |   |   |   |    |   |   |   |   |   |   |   |   |   |   |   |
| A |        | ì | é | £ | ¤  | ¥ | ¦ | § | ¨ | © | ª | « | ¬ |   | ® | ¯ |
| B | °      | ± | ² | ³ | ´  | µ | ¶ | · | , | ¹ | º | » | ¼ | ½ | ¾ | ¿ |
| C | À      | Á | Â | Ã | Ä  | Å | Æ | Ç | È | É | Ê | Ë | Ì | Í | Î | Ï |
| D | Ð      | Ñ | Ò | Ó | Ô  | Õ | Ö | × | Ø | Ù | Ú | Û | Ü | Ý | Þ | ß |
| E | à      | á | â | ã | ä  | å | æ | ç | è | é | ê | ë | ì | í | î | ï |
| F | ð      | ñ | ò | ó | ô  | õ | ö | ÷ | ø | ù | ú | û | ü | ý | þ | ÿ |

Table de correspondance entre un code et son caractère avec la norme UTF-8

**Exercice 3 : Utiliser Python**

Il existe 2 fonctions qui vont nous intéresser pour convertir un caractère en représentation décimale ASCII et inversement.

La première est **ord()**, elle donne le caractère décimal (et non hexadécimal comme dans les tables présentées précédemment) d'un caractère

Exemple : `ord('A')`, sortie 65

La seconde est **chr()**, elle donne le caractère correspondant à un décimal Exemple : `chr(65)`, sortie A

1. Écrire un petit programme Python qui va afficher les 256 caractères de la table ASCII étendue. Vous utiliserez les notions expliquées juste avant.
2. Écrire un second programme Python qui va afficher une lettre en minuscule, son code associé puis la majuscule et son code associé. Le programme doit traiter l'ensemble des lettres de l'alphabet.

**Exercice 4 : Aller plus loin avec l'UTF-8**

L'encodage UTF-8 utilise 1, 2, 3 ou 4 octets en respectant certaines règles :

Un texte en ASCII de base est codé de manière identique en UTF-8. On utilise un octet commençant par un bit 0 à gauche (bit de poids fort). Exemple : « A » est codé par 65<sub>10</sub>, ce qui correspond à 10000012 (7 bits). En utilisant le codage UTF-8, le binaire devient 010000012 (8 bits).

Les octets ne sont pas remplis entièrement. Les bits de poids fort du premier octet forment une suite de 1 indiquant le nombre d'octets utilisés pour coder le caractère. Les octets suivants commencent tous par le bloc binaire 10.

| 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 |

Tableau de correspondance entre la représentation binaire UTF-8 et sa signification

Le symbole € correspond à 8364<sub>10</sub>.

1. Convertir 8364<sub>10</sub> en binaire.
2. Donner le codage UTF-8 de ce caractère : €

On peut se demander combien d'octets sont nécessaires pour coder un caractère (codé en binaire sur  $n$  bits) en UTF-8.

Pour répondre à ce point nous pouvons nous appuyer sur l'exemple précédent. On peut constater que l'on regroupe les bits par 6 afin de former les octets, et on ajoute un octet pour indiquer le nombre total d'octets.

Par exemple ici,  $n=14$ , le nombre d'octets nécessaires en UTF-8 est donc :

$$E\left(\frac{14}{6}\right)+1, \text{ donc ici il faut 3 octets}$$

La fonction  $E(x)$  désigne la partie entière de  $x$ .

Si on généralise on peut donc affirmer que le nombre d'octets nécessaires est de

$$E\left(\frac{n}{6}\right)+1.$$

3. Écrire un algorithme qui permet, à partir de la valeur décimale d'un caractère, de trouver le nombre d'octets nécessaires pour coder ce caractère en UTF-8.