|  |  |  |
| --- | --- | --- |
| Numérique et Sciences Informatiques | | |
| 3h | **Représentation d'un texte en machine** |  |
| **Objectif** : connaitre comment sont codés les textes dans un ordinateur. | | |
| **Matériel**: Python | | |

**ASCII**

Avant 1960 de nombreux systèmes de codage des caractères alphanumériques existaient, ils étaient souvent incompatibles entre eux.

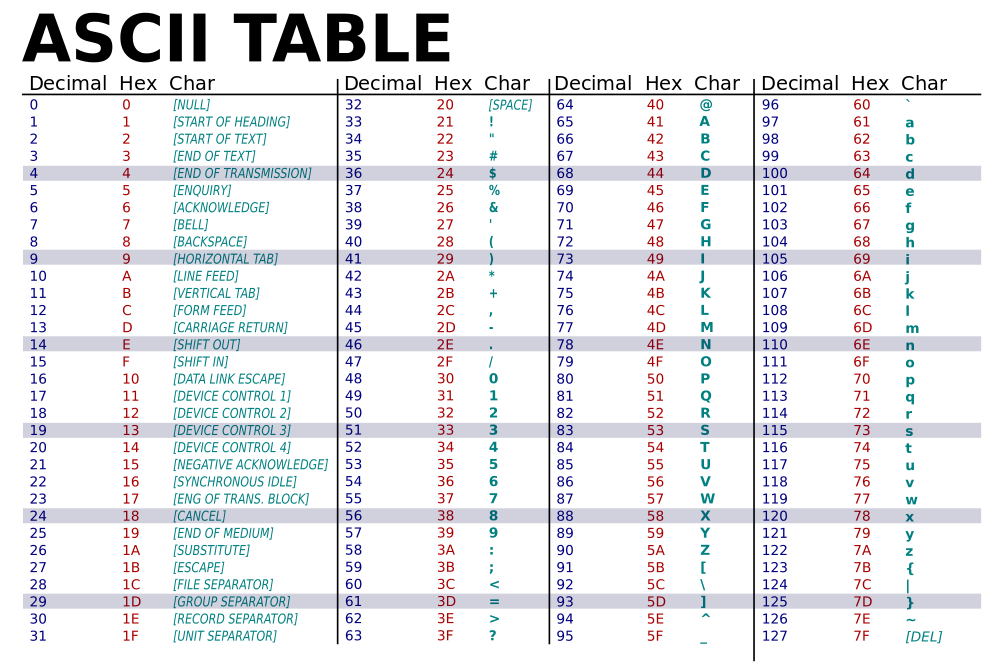
En 1960, l'organisation internationale de normalisation (ISO) décide de mettre un peu d'ordre dans ce bazar en créant la norme ASCII (American Standard Code for Information Interchange).

À chaque caractère est associé un code sur 1 octet.

Seuls 7 bits sont utiles pour coder les 128 caractères de la table ASCII (27 = 128).

Le 8e bit de l'octet sert de bit de contrôle (bit de parité) pour la détection des erreurs de transmission du code.

**Table ASCII :**



On constate que les codes de (0)10 à (32)10 ne correspondent pas à des caractères imprimables.

On constate qu'au 'A' majuscule correspond le code (65)10 = (41)16 = (100 0001)2

En python, on obtient donc 'A' par **'A'** ou **"A"** ou **chr(65)** ou **chr(0x41)** ou **chr(0b1000001)**

En python, on obtient le code décimal ASCII de 'A' par **ord('A')**

* Quel est le code binaire de 'a' en ASCII ?
* Le code binaire (0101 1000)2 correspond à quel caractère ASCII ?

**Informations sur les chaînes de caractères**

Une chaîne de caractères est composée de plusieurs caractères, chaque caractère est accessible grâce à son indice entre crochets.

Les indices allant de 0 jusqu’à la longueur de la chaîne de caractères - 1.

Par exemple, pour la chaîne de caractères **mot="bonjour",** **mot[0]** permet d’accéder au premier caractère, soit 'b'.

Pour obtenir la longueur de la chaîne de caractère on peut utiliser l’instruction **len(mot)**.

Il est aussi possible de concaténer (mettre bout à bout) des chaînes de caractères à l’aide de l’opérateur **« + »**.

**Code 1\*** : Ecrivez un programme qui demande à l’utilisateur d’écrire un mot et qui affiche la longueur de ce mot ainsi que l’indice associé à chaque lettre.

Exemple d’exécution :

Entrez un mot : bonjour

Votre mot est composé de 7 caractères

Le caractère n°0 est b

Le caractère n°1 est o

Le caractère n°2 est n

Le caractère n°3 est j

Le caractère n°4 est o

Le caractère n°5 est u

Le caractère n°6 est r

**Code 2\*** : Améliorez le programme pour qu’il affiche en plus le code décimal ASCII de chaque caractère du mot.

Exemple d’exécution :

Entrez un mot : bonjour

Votre mot est composé de 7 caractères

Le caractère n° 0 est b, son code ASCII en décimal est 98

**Code 3\*** : Ecrivez un programme qui demande à l’utilisateur d’entrer un nombre (qui sera sous forme de chaine de caractères) et qui calcule la somme des chiffres qui le compose.

Si la somme des chiffres est égale à 10 alors le programme affichera "Bravo" sinon il affichera "Perdu, la somme des chiffres de votre nombre fait : ?? ".

Exemple d’exécution :

Entrez un nombre : 361

Bravo

Entrez un nombre : 362

Perdu, la somme des chiffres de votre nombre fait : 11

**Code 4\*** : écrivez un programme qui demande à l’utilisateur d’entrer des mots tant que le mot entré n’est pas vide. Le programme affichera la phrase construite en concaténant l’ensemble des mots.

Exemple d’exécution :

Entrez un mot : Lycée

Entrez un mot : Jean

Entrez un mot : Moulin

Entrez un mot :

Lycée Jean Moulin

**Code 5\*** : En étudiant le programme suivant et son exécution proposée, devinez quel est le mot qui a été entré par l’utilisateur ?

mot=input("Entrez un mot : ")

for i in range(len(mot)):

print(chr(ord(mot[i])+8))

NSI

Exécution du programme :

Entrez un mot : \*\*\*

V

[

Q

**Code 6\*\*** : Ecrivez un programme qui transforme un mot qui est écrit en lettres minuscules en ce même mot en lettres majuscules.

Exemple d’exécution :

Entrez un mot en lettres minuscules : bonjour

BONJOUR

**Code 7\*\*** : Améliorez ce programme pour qu'il fonctionne pour des mots composés d’un mélange de lettres minuscules et majuscules.

Exemple d’exécution :

Entrez un mot : BonJoUr

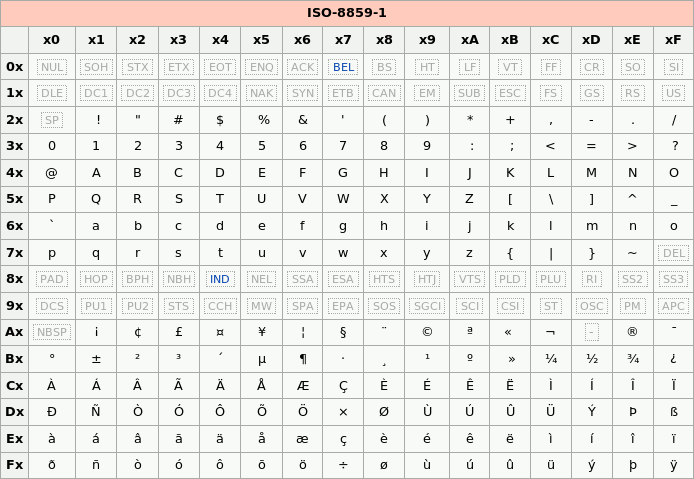
bONjOuR

**ISO-8859-1**

La norme ASCII convient bien à la langue anglaise, mais pose des problèmes dans d'autres langues, par exemple le français avec ses lettres accentuées.

C'est pour répondre à ce problème qu'est née la norme ISO-8859-1 qui reprend les mêmes principes que l'ASCII (codage sur 7 bits → 128 caractères) en utilisant un 8ième bit (codage sur 8 bits → 256 caractères).

Cette norme va être principalement utilisée dans les pays européens puisqu'elle permet d'encoder les caractères utilisés dans les principales langues européennes (la norme **ISO-8859-1** est aussi appelée **latin-1**).



Il existe beaucoup d'autres langues dans le monde qui n'utilisent pas l'alphabet latin, comme par exemple le chinois.

D'autres normes ont donc vu le jour et cette multiplication des normes a très rapidement posé problème.

Si un chinois envoie un mail dans sa langue vers la France, les caractères chinois seront décodés avec la norme latin-1 ce qui pourra donner :*新字体  → q@éjm|à*

**UNICODE**

En 1991 la norme **Unicode** a vu le jour.

Unicode a pour ambition de rassembler tous les caractères existant dans le monde afin qu'une personne utilisant unicode puisse, sans changer la configuration de son traitement de texte, à la fois lire des textes en français, en chinois, en russe ...

Unicode est une table (points de codes) qui regroupe tous les caractères existant dans le monde.

* Points de code (en hexadécimal) de **U+0000 et U+FFFF** : plan multilingue de base (le premier, numéroté 0) est le plus utilisé car il contient la plupart des caractères utilisés par les langues modernes les plus courantes dans le monde.
* Points de code (en hexadécimal) de **U+10000 à U+1FFFF** : plan complémentaire (numéro 1) est utilisé pour les caractères d’écritures anciennes et des jeux de symboles.
* ...

En python, on obtient 'é' par **'é'** ou **"é"** ou **chr(233)** ou **chr(0xE9)** ou **chr(0b11101001)** ou **"\u00E9".**

En python, on obtient 'Ω' par **chr(937)** ou **"\u03A9".**

En python, on obtient le **point de code** **décimal** Unicode de 'é' par **ord('é') → 233**

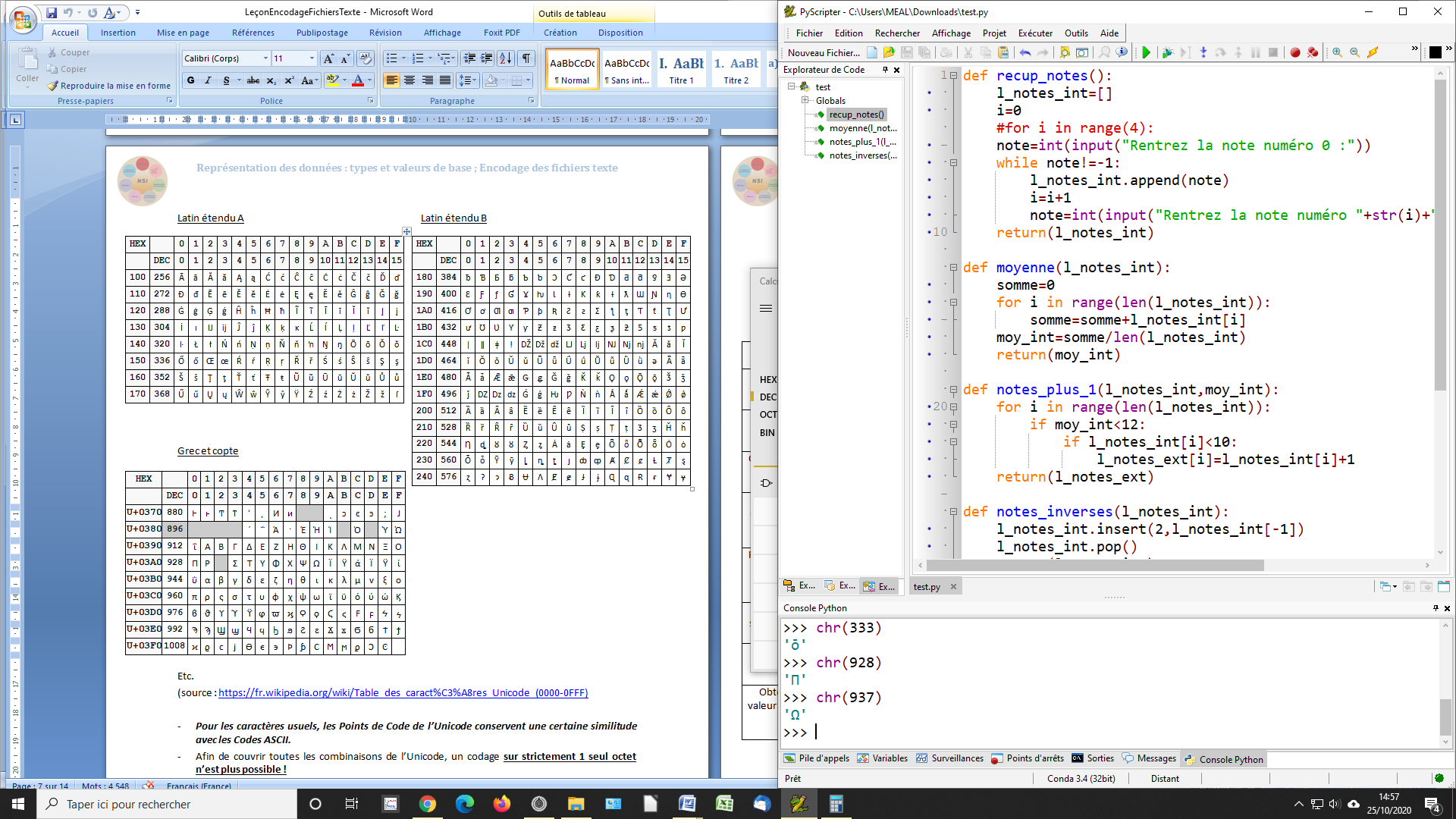
En python, on obtient le **point de code** **hexadécimal** Unicode de 'é' par **ord('é') → U+00E9**

En python, on obtient le **point de code** **décimal** Unicode de 'Ω' par **ord('Ω') → 937**

En python, on obtient le **point de code** **hexadécimal** Unicode de 'Ω' par **ord('Ω') → U+03A9**

* Quel est le **point de code hexadécimal** de 'A' en Unicode sachant que les code ASCII sont les premiers points de code Unicode ?
* Quel est le **point de code hexadécimal** de 'ϕ' en Unicode ?

**Unicode pour le** [**Grec et le Copte**](https://fr.wikipedia.org/wiki/Table_des_caract%C3%A8res_Unicode/U0370) **:**



**UTF-8**

Si on veut stocker sur un support informatique un texte, il faut choisir un codage pour chaque point de code Unicode.

Le codage le plus utilisé est UTF-8 (Unicode Transformation Format-8bits).

Il existe également les codages UTF-16 et UTF-32.

L'intérêt du codage UTF-8 est d'avoir un code à longueur variable (1 à 4 octets), ce qui permet d’optimiser la taille des fichiers.

Pour les 128 premiers points de codes, l'UTF-8 nécessite 1 octet (= code ASCII), pour les 128 points de codes suivant, l'UTF-8 nécessite 2 octets ...

**Exemple :**

UTF-8 de 'A' = (41)16.

UTF-8 de 'é' = (C3A9)16.

Codes UTF-8 du fichier contenant le texte : "Spécialité NSI" :

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 83 | 112 | 195 | 169 | 99 | 105 | 97 | 108 | 105 | 116 | 195 | 169 | 32 | 78 | 83 | 73 |

Indiquez par quelles valeurs sont codées les symboles suivants :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Lettres : | S | p | é | | c | i |
| Valeurs (base 10) : | 83 |  |  |  |  |  |
| Symboles en ISO-8859-1 : | S |  |  |  |  |  |

83 (1 octet) est bien le code UTF-8 de 'S' (= 83 : point de code en Unicode).

195+169 (2 octets) est le code UTF-8 de 'é' (≠ 233 : point de code en Unicode).

**Remarque :**

Sous un logiciel utilisant l'ISO-8859-1 (c'est la cas de Windows), le texte codé en UTF-8 s'affichera comme cela : **SpÃ©cialitÃ© NSI**

**Ã** : codé (195)10 = (C3)16 en ISO-8859-1

**©** : codé (169)10 = (A9)16 en ISO-8859-1