Notions importantes du codage de caractères

- En anglais, la capacité de la mémoire informatique est généralement exprimée en **bytes**, alors qu'en français on l'exprime plutôt en **octets**.
- Un **octet**
  - un byte de 8 bits codant une information.
  - permet de représenter 2^8 (2 puissance 8) nombres, soit 256 valeurs différentes ou jusqu'à 256 caractères différents.

La norme Unicode

- données de type string sont des chaines Unicode = les identifiants numériques de leurs caractères sont uniques.
- norme Unicode ne précise pas la manière dont ces valeurs numériques doivent être encodées
- chaque système informatique est libre d'encoder « en interne » cet identifiant comme bon lui semble

* L'ordinateur ne comprend qu'une suite de 1 et de 0 (les « bits ») souvent traités par groupes de 8 (les « octets »): 16, 32, … 64.

- Pour toutes les entrées ou sorties de chaines de car., on doit toujours considérer qu'il s'agit concrètement de séquences d'octets, et utiliser divers mécanismes pour convertir ces séquences d'octets en chaines de car., ou vice-versa.
- On peut ouvrir un fichier en mode « binaire » en transmettant l'argument **"rb"** à la fonction **open()**. Dans ce mode, les octets sont transférés à l'état brut, sans conversion d'aucune sorte.

```
>>> chaine = "Amélie et Eugène\n"
>>> of =open("test.txt", "w")
>>> of.write(chaine)
17
>>> of.close()
```

```
>>> of =open("test.txt", "rb")          # "rb" => mode lecture (r) binaire (b)
>>> octets =of.read()
>>> of.close()
>>> type(octets)
<class 'bytes'>
```

- Le mode « binaire » de lecture (l'argument **"rb"** de la fonction **open()**).
  - Dans ce mode, les octets sont transférés à l'état brut, sans conversion d'aucune sorte.
- La lecture avec **read()** ne nous fournit plus une ch. de car. mais une chaine d'octets, et la variable qui les accueille est pour cette raison automatiquement typée comme variable du type bytes
- En procédant ainsi, nous ne récupérons donc pas notre ch. de car. initiale, mais bien sa traduction concrète en octets

```
>>> print(octets)
b'Am\xc3\xa9lie et Eug\xc3\xa8ne\n'
```

```
>>> for oct in octets:
...     print(oct, end =' ')
...
65 109 195 169 108 105 101 32 101 116 32 69 117 103 195 168 110 101 10
```

Module Python spécialisé : le module **pickle**

Permet d'enregistrer des données avec conservation de leur type

```
>>> import pickle
>>> a, b, c = 27, 12.96, [5, 4.83, "René"]
>>> f =open('donnees_test', 'wb')
>>> pickle.dump(a, f)
>>> pickle.dump(b, f)
>>> pickle.dump(c, f)
>>> f.close()
>>> f = open('donnes_test', 'rb')
>>> j = pickle.load(f)
>>> k = pickle.load(f)
>>> l = pickle.load(f)
>>> print(j, type(j))
27 <class 'int'>
>>> print(k, type(k))
12.96 <class 'float'>
>>> print(l, type(l))
[5, 4.83, 'René'] <class 'list'>
>>> f.close()
```

- Les contenus des trois variables **a**, **b** et **c** sont enregistrés dans le fichier ***donnees_test***, et ensuite fidèlement restitués, avec leur type, dans les variables **j**, **k** et **l**.
- Les fichiers traités à l'aide des fonctions du module *pickle* ne seront pas des fichiers texte, mais bien des *fichiers binaires =>* ils doivent obligatoirement être ouverts à l'aide de la fonction **open()**.
- On utilise l'argument 'wb' pour ouvrir un fichier binaire en écriture et l'argument 'rb' pour ouvrir un fichier binaire en lecture

Conversion d'une chaîne bytes en chaîne string

```
>>> ch_car = octets.decode("utf8")
>>> ch_car
'Amélie et Eugène\n'
>>> type(ch_car)
<class 'str'>
```

```
>>> for c in ch_car:
...     print(c, end =' ')
...
A m é l i e   e t   E u g è n e
```

Conversion d'une chaîne string en chaîne bytes

```
>>> chaine = "Bonne fête de Noël"
>>> octets_u = chaine.encode("Utf-8")
>>> octets_l = chaine.encode("Latin-1")
>>> octets_u
b'Bonne f\xc3\xaate de No\xc3\xabl'
>>> octets_l
b'Bonne f\xeate de No\xebl'
```

Conversions automatiques lors du traitement des fichiers

- Lorsqu'on ouvre un fichier en écriture en choisissant **"w"** ou **"a"**, Python encode automatiquement les chaînes à enregistrer en suivant la norme par défaut du système d'exploitation utilisé
- Enregistrer des textes en leur appliquant un encodage différent de celui qui est prévu par défaut

```
>>> chaine ="Amélie et Eugène\n"
>>> of =open("test.txt", "w", encoding ="Latin-1")
>>> of.write(chaine)
17
>>> of.close()
>>> of =open("test.txt", "rb")
>>> octets =of.read()
```

```
>>> of.close()
>>> print(octets)
b'Am\xe9lie et Eug\xe8ne\n'
```

Conversions automatiques lors du traitement des fichiers

- Lorsqu'on ouvre un fichier en lecture, Python considère que le fichier est encodé suivant la norme par défaut du système d'exploitation
- Re-ouvrons le fichier *test.txt:*

```
>>> of =open("test.txt", "r")
>>> ch_lue =of.read()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/usr/lib/python3.1/codecs.py", line 300, in decode
    (result, consumed) = self._buffer_decode(data, self.errors, final)
UnicodeDecodeError: 'utf8' codec can't decode bytes in position 2-4:
invalid data
```

Tout rentre dans l'ordre si nous précisons :

```
>>> of =open("test.txt", "r", encoding ="Latin-1")
>>> ch_lue =of.read()
>>> of.close()
>>> ch_lue
'Amélie et Eugène\n'
```

Cas des scripts Python

- Afin que Python puisse interpréter correctement les textes, il est conseillé d'y inclure toujours l'un des pseudo-commentaires suivants (obligatoirement à la 1e ou à la 2e ligne) :

  # -- coding:Latin-1 -*-*

ou bien :

  # -- coding:Utf-8 -*-*

- On peut omettre ce pseudo-commentaire si l'on est certain que les scripts sont encodés en *Utf-8*, l'encodage-norme par défaut pour les scripts Python

Accéder à d'autres caractères que ceux du clavier

- La fonction **ord(ch)**
  - accepte n'importe quel caractère comme argument;
  - Fournit la valeur de l'identifiant numérique correspondant à ce caractère.

    ord("A") renvoie à la valeur 65

    ord("Ĩ") renvoie la valeur 296
- La fonction **chr(num)**
  - fait exactement le contraire, en présentant le caractère typographique dont l'identifiant Unicode est égal à **num**

    chr(65) renvoie le caractère **A**

    chr(1046) renvoie le caractère cyrillique **Ж.**

Travail à faire

- Ecrivez un script qui recopie un fichier texte en remplaçant tous ses espaces par le groupe de trois caractères **-*-** . Le fichier à copier sera fourni encodé à la norme Latin-1, et le fichier destinataire devra être encodé en Utf-8. Les noms des 2 fichiers devront être demandés en début de script.

Travail à faire

- Ecrivez un script qui recopie en *Utf-8* un fichier texte encodé à l'origine en *Latin-1*, en veillant en outre à ce que chaque mot commence par une majuscule. Le programme demandera les noms des fichiers à l'utilisateur. Les opérations de lecture et d'écriture des fichiers auront lieu en mode texte ordinaire.