info223 : Science informatique
TD 2 : représentation des caractères, dates et autres

Pierre Hyvernat
Laboratoire de mathématiques de l'université de Savoie
bâtiment Chablais, bureau 22, poste : 94 22
email : Pierre.Hyvernat@univ-savoie.fr
www : http://www.lama.univ-savoie.fr/~hyvernat/

Partie 1 : Caractères

Rappels :
- ISO-8859-n sont des extensions de l'ASCII pour utiliser le bit de poids fort (sur un octet),
- UTF-8 (UCS Transformation Format 8 bit, UCS = Universal Character Set) est un encodage de l'Unicode (sur un, deux, trois ou quatre octets).

En UTF-8, la taille du code d'un caractère est variable :
- `0.......` : caractère ASCII sur un seul octet,
- `110.....` `10......` : caractère sur deux octets,
- `1110....` `10......` `10......` : caractère trois sur octets,
- `11110...` `10......` `10......` `10......` : caractère sur quatre octets.

Question 1. Comment peut-on traduire des fichiers textes entre différents codages des caractères :
- de l'ASCII vers l'ISO-8859-n
- de l'ISO-8859-n vers l'ASCII
- de l'ASCII vers l'UTF-8
- de l'UTF-8 vers l'ASCII
- de l'ISO-8859-n vers l'ISO-8859-m
- ...

Que peut-on faire lorsque la traduction est impossible ?

Question 2. Avec l'ASCII ou les ISO-8859-n, la taille en octets d'une chaîne de caractères est égale à son nombre de caractères. Ce n'est plus le cas pour l'UTF-8.

Pour chacune des chaîne si dessous, dites si la chaîne (représentée en hexadécimal) est de l'UTF-8 valide, et si oui, combien de caractères elle contient :
- `63 6f 75 63 6f 75`
- `48 e9 20 21`
- `48 c3 a9 20 21`

Question 3. Combien de symboles peut-on représenter avec le codage UTF-8 ?

Partie 2 : Dates

Sur le système de fichier FAT, les dates sont codées sur 2 octets et les bits sont répartis de la manière suivante :

`aaaaaaam` `mmmjjjjj`

- `aaaaaaa` (7 bits) représente l'année depuis 1980,
- `mmmm` (4 bits) représente le mois,
- et `jjjjj` (5 bits) représente le jours.

Une heure est représentée sur deux octets également :

`hhhhhmmm` `mmmsssss`

- `hhhhh` (5 bits) représent l'heure,
- `mmmmmm` (6 bits) représente les minutes,
- et `sssss` (5 bits) représente les secondes.

Question 1. Que pensez-vous du codage d'une heure sur 2 octets et en particulier du codage des secondes sur 5 bits ?

Question 2. Calculez la date correspondant à `4d 4e 76 d9` si la représentation est celle du système de fichier FAT (date sur les octets de poids fort, heure sur les octets de poids faible),

Question 3. Les dates (avec l'heure) sur les systèmes Unix sont codées sur 4 octets : on code un entier (en complément à 2) donnant le nombres de secondes depuis le 1er janvier 1970. (Les nombres négatifs représentent donc des dates avant 1970...)

Quelle est la date correspondants aux octets `4d 4e 76 d9` ?

Question 4. Calculez la date du "bug de l'an 2000" pour les dates au format FAT et les date au format Unix, c'est à dire, l'instant où le temps va "boucler" à cause d'un dépassement de capacité...

Question 5. Donnez une manière simple de récupérer l'année d'une date au format FAT si elle vous est donnée sur un entier 32 bits.

Partie 3 : Images

Question 1. Écrivez un petit programme en Python qui crée une image "`.pbm`" (noir et blanc) carrée de taille 32 pixels par 32 pixels avec :
- un fond noir,
- une diagonale blanche.

Question 2. Quelle sera la taille (en octets) du fichier images correspondant ?

Même question si l'image est de taille 320 par 320 pixels ?

Question 3. Proposez une manière de coder les images noir et blanc inspirée du format `pbm` qui est plus efficace.