Université de Caen **Année 2011/2012**
UFR de Sciences **session de juin**

Licence L2,

Examen de Programmation Objet / Python

Durée: 2,0 H. Les documents ne sont pas autorisés.

Chaque candidat doit, au début de l'épreuve, porter son nom dans le coin de la copie qu'il cachera par collage après avoir été pointé. Il devra en outre porter son numéro de place sur chacune des copies, intercalaires, ou pièces annexées.

1. **Le code de César**. Construire une **classe** d'objets qui se comportent comme des dictionnaires indexés par des chaînes de caractères, et dont les valeurs sont également des chaînes. Par ex. `d['yeux']` donne `'bhxa'`. Cependant, `d` *n'est pas un vrai dictionnaire*, un tel objet n'accepte que des chaînes, ***et les transforme***. Il prend la chaîne-argument, caractère par caractère, ajoute 3 à code de chacun, reconstruit le caractère résultant, et les compose dans la chaîne résultante. Par ex. `'g'` devient `'j'`. Si le code du caractère dépasse le code de `'z'`, on boucle : `'z'` est suivi par `'a'`, `'b'`, etc. Pour simplicité, ignorons les lettres majuscules et les lettres accentuées (on considère uniquement les 26 lettres latines minuscules).
   Prévoir une paramétrisation globale par une variable de classe – le décalage, pas obligatoirement égal à 3, mais quelconque. Utiliser les fonctions : `ord` qui convertit un caractère en un entier, et `chr` – la conversion inverse. Vous pourrez avoir besoin de l'opérateur modulo : `%`. Construire d'abord la fonction qui effectue le transcodage, qui servira comme une *méthode d'indexation* dans la classe des objets en question.

2. Écrire une fonction ***générateur*** « filtre »parametré par un autre générateur – source, et disposant d'une valeur initiale égale à zéro.

```
def difil(source):
  val0=0.0
  ...
  v = source.next()     # ou une boucle for, peut-être?...
  ...
  yield (...)
  ...
```

   Le générateur lit élément par élément sa source et créé un nouveau flot de résultats, selon la *tendance* (pente) du flot d'entrée. Quand les valeurs dans le flot d'entrée augmentent, le générateur retourne (par `yield`, bien sûr) $+1$; si elles diminuent : -1. Si les valeurs ne changent pas, le filtre émet zéro.
   Par exemple, si la séquence d'entrée est : `1 -2 3 2 1 -2 -3 9 -1 1 -2 -1 3 5 -1 2 -1 ...`, le flot de sortie sera `1 -1 1 -1 -1 -1 -1 1 -1 1 -1 1 1 1 -1 1 -1 ...`.

3. Concevoir et coder la classe `Date` des objets qui représentent la date, avec trois champs numériques : (année, mois, jour). Coder l'arithmétique (restreinte) pour ces objets :
   - On peut ajouter ou soustraire deux objets `Date`.

   C'est tout. Pour simplicité – **oublier l'existence des années bissextiles !** Tous les février ont 28 jours. Le résultat doit être normalisé au sens que les mois et les jours doivent être toujours positifs, l'année peut être négative si on soustrait une date plus tardive ; par ex. « moins un jour » vaut -1 an, 11 mois et 30 jours.

Écrivez de manière lisible, un texte difficile à déchiffrer risque d'être rejeté. Ne jamais copiez des morceaux de code inutile, tout code hors sujet sera *pénalisé.* Évitez toute discussion qualitative (« philosophique ») inutile. Par contre, *commentez votre code*, sinon il risque d'être peu compréhensible.