Séance 7: AGENDA

Université Paris-Diderot

Objectifs:

- Listes unidimensionnels
- Listes de chaînes de caractères
- Listes d'entiers
- Parcours de listes

Dans ce TP nous allons écrire des fonctions pour manipuler un agenda personnel.

Placez les fichiers dans le bon répertoire. Pour ce TP, vous devez compléter le fichier `agenda.py` et avoir dans votre répertoire le fichier `agenda.dat` qui contient l'agenda personnel.

L'agenda contient au plus un évènement par jour. Nous allons le représenter par une liste de chaînes de caractères. Dans un agenda `a`, la chaîne de caractère `a[i]` est la description de l'évènement qui se produit le $i+1$-ème jour de l'année. Par exemple, `a[0]` est la chaîne de caractères `"Jour de l'An"`. S'il n'y a aucun évènement particulier au $i+1$-ème jour, `a[i]` est la chaîne de caractères vide `""`. L'agenda ne concerne que l'année 2016, qui est une année bissextile, donc `a` est une liste de longueur 366.

Exercice 1 (Afficher l'agenda jour après jour, $\star$)

1. *Dans le fichier `agenda.py` vous trouverez une fonction `loadAgenda` qui vous permet de lire un agenda d'un fichier externe (par exemple le fichier `agenda.dat` fourni également). Cette fonction retourne comme résultat l'agenda lu. Par exemple, vous pouvez lier la variable `a` à l'agenda lu du fichier `agenda.dat` par*

```
a = loadAgenda("agenda.dat")
```

   Écrire une procédure `affiche(a)` qui prend un agenda `a` en paramètre, et l'affiche jour par jour en rappelant le numéro du jour au début de la ligne.

   Contrat:
   `affiche(a)` *affiche*

   ```
   0 Jour de l'An
   1
   2
   ...
   86 Pâques
   ...
   ```

2. *Modifiez votre procédure pour qu'elle n'affiche que les jours où il y a un évènement.*

   Contrat:
   `affiche(a)` *affiche*

   ```
   0 Jour de l'An
   ```

86 Pâques
87 Lundi de Pâques
...

3. *Initialiser une liste*

```
wDays = ["Lun", "Mar", "Mer", "Jeu", "Ven", "Sam", "Dim"]

```

puis utilisez-la pour que votre procédure affiche le jour de la semaine en début de ligne.

Contrat:
`affiche(a)` *affiche*

Ven 0 Jour de l'An
Dim 86 Pâques
Lun 87 Lundi de Pâques
...

4. *Affichez le nombre d'évènements rentrés dans l'agenda avant d'afficher l'agenda.*

Contrat:
`affiche(a)` *affiche*

Il y a 12 évènements dans l'agenda.
Ven 0 Jour de l'An
Dim 86 Pâques
Lun 87 Lundi de Pâques
...

5. *Modifiez la procédure* `affiche` *pour qu'elle prenne en arguments une liste de chaînes de caractères* `a` *contenant l'agenda, un entier* `startDay` *contenant le jour de départ, et un booléen* `reverse`*. Si* `reverse` *vaut* `True`*, la procédure affiche les évènements planifiés avant le jour* `startDay`*, dans l'ordre chronologique inverse. Sinon, elle affiche les évènements planifiés après le jour* `startDay` *dans l'ordre chronologique.*[1]

Contrat:
Un appel à `affiche(a,0,False)` *affiche l'agenda comme à la question précédente. Un appel à* `affiche(a,87,True)` *affiche le lundi de Pâques, puis Pâques, puis le jour de l'An.*

□

Exercice 2 (Affichage de la date, ⋆⋆)
On veut maintenant remplacer le numéro du jour dans l'année par une date lisible. On a déjà vu la conversion d'un numéro de jour en une date dans un TP précédent, mais on va suivre une autre approche. La première étape est d'écrire une fonction `monthOfDayNumber` *qui prend en argument le numéro d'un jour et renvoie le mois dans lequel tombe ce jour.*

Contrat:
Par exemple `monthOfDayNumber(0)` *renvoie* 1 *car le jour* 0 *est le premier janvier, et janvier est le mois* 1*. De même,* `monthOfDayNumber(86)` *renvoie* 3 *car le jour* 86 *est le 27 mars.*

Un enseignant (un peu fatigué par les corrections de copies) a écrit la fonction suivante.

```
def monthOfDayNumber (dayNumber):
    daysInMonth = [31,29,31,30,31,30,31,31,30,31,30,31]
```

1. [BONUS] Évitez autant que possible d'avoir du code redondant. Vous pouvez utiliser par exemple une fonction pour l'affichage d'un évènement. Vous pouvez aussi écrire une seule boucle dont le déroulement dépend de la variable `reverse`.

```
    res = 1
    for i in range(0,12,1):
        if (dayNumber > daysInMonth[i]):
            dayNumber = dayNumber - daysInMonth[i]
            res = res + 1
    return res
```

code/exo2.py

1. *Vous trouverez le code de cette fonction dans le fichier* `exo2.py` *fourni avec ce TP. Copiez le dans votre fichier* `agenda.py`*, puis modifier votre fonction* `affiche` *pour qu'elle affiche aussi le mois avec chaque évènement :*

   Contrat:
   `affiche(a,0,false)` *affiche*

   ```
   Il y a 12 évènements dans l'agenda.
   Ven 0 1 Jour de l'An
   Dim 86 3 Pâques
   Lun 87 3 Lundi de Pâques
   Dim 121 4 Fête du travail ...
   ```

2. *Si vous regardez bien, il y a un problème avec la fête du travail. La fonction* `monthOfDayNumber` *a des erreurs ! Corrigez-la.*
3. *En vous inspirant beaucoup de la fonction* `monthOfDayNumber` *corrigée, créez une fonction* `dateOfDayNumber` *qui renvoie une liste de deux entiers, le premier entier étant le numéro du jour dans le mois, et le second le numéro du mois dans l'année.*

   Contrat:
   Par exemple `dateOfDayNumber(0)` *renvoie la liste* `[1,1]`*, et* `dateOfDayNumber(86)` *renvoie la liste* `[28,3]`*.*

4. *Utiliser la fonction* `dateOfDayNumber` *pour afficher la date au format habituel.*

   Contrat:
   Le programme affiche

   ```
   Il y a 12 évènements dans l'agenda.
   Ven 1/1 Jour de l'An
   Dim 27/3 3 Pâques
   Lun 28/3 Lundi de Pâques
   ...
   ```

□

Exercice 3 (Modifier l'agenda, ⋆⋆)

1. *Après avoir chargé l'agenda, et avant de l'afficher, ajoutez dans l'agenda votre anniversaire. Vérifiez qu'il s'affiche à la bonne place.*
2. *Ajoutez dans l'agenda l'évènement* `Bac` *sur chacune des entrées du 15 au 22 juin inclus.*

   Contrat:
   Le programme affiche

   ```
   ...
   Mer 15/6 Bac
   Jeu 16/6 Bac
   Ven 17/6 Bac
   ...
   ```

3. *Ajoutez dans l'agenda toutes les dates de séances de TP mini-projet avec leur numéro.*

Contrat:

En supposant que votre TP de mini-projet a lieu le lundi, le programme affiche

```
...
Lun 12/9 TP 1 IP1
Lun 19/9 TP 2 IP1
Lun 26/9 TP 3 IP1
...
Lun 28/11 TP 12 IP1
```

□

Exercice 4 (Question bonus : vue annuelle, ⋆⋆⋆)

Écrivez une fonction qui prend en paramètre une liste représentant un agenda et qui affiche mois par mois, et semaine par semaine, les dates contenant des évènements, en les faisant apparaître dans la colonne du jour de la semaine correspondant.

Contrat:

En supposant que l'agenda est celui des questions précédentes, la fonction affiche

```
Janvier
--------------------------
Lun Mar Mer Jeu Ven Sam Dim
                 01 ... ...
... ... ... ... ... ... ...
... ... ... ... ... ... ...
... ... ... ... ... ... ...
... ... ... ... ... ... ...

Février
--------------------------
Lun Mar Mer Jeu Ven Sam Dim
... ... ... ... ... ... ...
... ... ... ... ... ... ...
... ... ... ... ... ... ...
... ... ... ... ... ... ...
...

Mars
--------------------------
Lun Mar Mer Jeu Ven Sam Dim
    ... ... ... ... ... ...
... ... ... ... ... ... ...
... ... ... ... ... ... ...
... ... ... ... ... ...  27
 28 ... ... ...
```

etc.

□