

---

## TRAVAUX PRATIQUES n° 5

### Programmation fonctionnelle

---

Dans ce TP, nous allons écrire plusieurs fonctions permettant de faire des calculs sur les dates du calendrier. Le but est d'apprendre à créer et utiliser des fonctions de façon organisée sous le paradigme de la programmation fonctionnelle.

Il est important de tenir compte des bonnes pratiques de programmation suivantes :

★ **print ou return ?**

- Si l'énoncé vous demande de **renvoyer** un résultat, il faudra utiliser un `return` et non `print`.
- Si l'énoncé vous demande d'**afficher** un résultat, il faudra utiliser un `print` et non `return`.
- En cas de doute, on utilise un `return`.

★ **Eviter la redondance**

L'un des pires ennemi du programmeur est la redondance du code : on doit éviter le plus possible d'écrire plusieurs fois le même code. L'un des avantages principaux des fonctions en informatique est de résoudre ce problème : lorsqu'un morceau de code est destiné à être utilisé plusieurs fois, on l'écrit une seule fois dans une fonction que l'on réutilisera partout où l'on en a besoin.

★ **Tester les fonctions**

Après avoir codé une fonction, on doit s'assurer de son bon fonctionnement. Il faut :

- vérifier qu'il n'y a pas d'erreur à l'exécution ;
- tester la cohérence des résultats sur des exemples.

### Année bissextile

Une année  $n \geq 1$  (l'an 0 n'existe pas) est **bissextile** si et seulement ( $n$  est divisible par 4 mais pas par 100) ou ( $n$  est divisible par 400). Ainsi 2000 et 2012 étaient bissextiles mais 2100 ne le sera pas.

Exercice 1 : Écrire une fonction `bissextile(annee)` prenant un argument `annee`  $\in \mathbb{N}^*$  et qui renvoie `True` si l'année est bissextile et `False` sinon.

*Bonus : faire une fonction en une seule ligne !*

```
>>> bissextile(2024)
True
```

Exercice 2 : Écrire une fonction `nb_jours_annee(annee)` prenant un argument `annee`  $\in \mathbb{N}^*$  et qui renvoie le nombre de jours de l'année.

*On rappelle qu'il faut utiliser la fonction `bissextile`.*

```
>>> nb_jours_annee(2024)
366
```

Exercice 3 : Écrire une fonction `nb_jours_mois(mois, annee)` prenant un argument `mois`  $\in \llbracket 1, 12 \rrbracket$  et un argument `annee`  $\in \mathbb{N}^*$  et qui renvoie le nombre de jours du mois dans l'année.

```
>>> nb_jours_mois(2, 2023)
28
>>> nb_jours_mois(2, 2024)
29
```

Exercice 4 : Écrire une fonction `nb_jours_ecoules(jour, mois, annee)` prenant un argument `jour`  $\in \llbracket 1, 31 \rrbracket$ , un argument `mois`  $\in \llbracket 1, 12 \rrbracket$  et un argument `annee`  $\in \mathbb{N}^*$  et qui renvoie le nombre de jours écoulés depuis le 1<sup>er</sup> janvier de l'an 1 jusqu'à cette date.

```
>>> nb_jours_ecoules(24, 11, 2023)
738848
>>> nb_jours_ecoules(1, 12, 2023)
738855
```

Exercice 5 : Écrire une fonction `difference_de_jours(date1, date2)` prenant deux arguments `date1` et `date2` sous forme de liste de 3 entiers (par exemple `[1, 10, 2021]`) et qui renvoie le nombre de jours écoulés entre ces deux dates.

*On rappelle que pour accéder aux éléments d'une liste, on utilise la syntaxe `liste[indice]`. Par exemple si `date = [24, 11, 2023]` alors `date[0] = 24`, `date[1] = 11` et `date[2] = 2023`.*

Combien de jours avez-vous vécu ?

```
>>> difference_de_jours([1, 1, 2000], [24, 11, 2023])
8728
```

Exercice 6 : Écrire une fonction `date(n)` prenant un argument `n`  $\in \mathbb{N}^*$  et qui renvoie la date, sous forme de liste de 3 entiers, pour laquelle le nombre de jours écoulés depuis le 1<sup>er</sup> janvier de l'an 1 est `n`.

À quelle date se sera-t-il écoulé un million de jours depuis le 1<sup>er</sup> janvier de l'an 1 ?

```
>>> date(738848)
[24, 11, 2023]
```

Exercice 7 : Sachant que le 1<sup>er</sup> janvier de l'an 1 était un lundi, écrire une fonction `jour_de_la_semaine(jour, mois, annee)` prenant un argument `jour`  $\in \llbracket 1, 31 \rrbracket$ , un argument `mois`  $\in \llbracket 1, 12 \rrbracket$  et un argument `annee`  $\in \mathbb{N}^*$  et qui renvoie le jour de la semaine correspondant à cette date.

```
>>> jour_de_la_semaine(24, 11, 2023)
'Vendredi'
```

Exercice 8 (*Paraskevidékatriphobe s'abstenir*) :

Écrire une fonction `prochain_vendredi_13(jour, mois, annee)` prenant un argument `jour`  $\in \llbracket 1, 31 \rrbracket$ , un argument `mois`  $\in \llbracket 1, 12 \rrbracket$  et un argument `annee`  $\in \mathbb{N}^*$  et qui renvoie la date du prochain vendredi 13.

```
>>> prochain_vendredi_13(24, 11, 2023)
[13, 9, 2024]
```