Fonctions : exercices

Si vous n'avez pas encore réussi à installer PYTHON chez vous, vous pouvez utiliser, en attendant :

- Jupyter en ligne : http://jupyter.org/try
- Python Tutor, qui permet en plus de bien comprendre comment cela fonctionne : http://www.pythontutor.com/visualize.html

✔ Avant de traiter ces exercices, il faut avoir relu le cours sur les tuples et les fonctions et terminé le TP n° 4 sur les fonctions.

Rappel : La syntaxe pour définir une fonction est :

PYTHON
```
def nom_fonction(arg1, arg2, ...):
    """Chaîne de documentation."""
    # instructions
    # à
    # réaliser
    return valeur
```

Par exemple :

PYTHON
```
import math

def volume_sphere(r):
    """Volume d'une sphère de rayon `r`."""
    return (4/3) * math.pi * r**3
```

On peut ensuite faire un *appel* à la fonction :

PYTHON
```
volume_soleil = volume_sphere(696342)
print(volume_soleil)
```

1 Mauvais élèves

On demande à l'élève Marius d'écrire une fonction permettant de vérifier si le triplet $(65, 72, 97)$ forme un triplet pythagoricien. Il propose le programme suivant :

PYTHON
```
def triplet_pythagoricien(x, y, z):
    x = 65
    y = 72
    z = 97
    return x**2 + y**2 == z**2
```

1. Est-ce que cette fonction permet de répondre à la question ?
2. Expliquer pourquoi Marius n'a pas bien compris le fonctionnement et l'intérêt d'une fonction et corriger.

On demande à une autre élève, Adélie, d'écrire une fonction `hypotenuse`, prenant en arguments les longueurs des deux cathètes d'un triangle rectangle et renvoyant la longueur de l'hypoténuse. Adélie propose le programme suivant :

PYTHON
```
def hypotenuse(a, b):
    (a**2 + b**2) ** 0.5
```

3. Que vaut `hypotenuse(3, 4)` ?

Adélie corrige :

PYTHON
```
def hypotenuse(a, b):
    print((a**2 + b**2) ** 0.5)
```

4. Qu'en pensez-vous ?

2 Valeur de retour d'une fonction

Dans la séquence interactive suivante, après l'avoir essayée et bien comprise, indiquer pour chaque ligne (y compris les lignes vides) s'il s'agit :

- D'une expression ;
- D'une affectation ;
- D'une valeur (affichée en mode interactif uniquement) ;
- De la valeur `None` (qui n'est pas affichée en mode interactif) ;
- D'un affichage à l'écran (en mode interactif ou non).

PYTHON

```
In[1] retour = print("Bonjour")
Bonjour
In[2] retour

In[3] type(retour)
NoneType
In[4] retour == None
True
```

3 Exercices

⚠ Il faut impérativement et toujours tester vos fonctions pour vous assurer qu'elles font bien ce que vous vouliez. Il faut vérifier leur comportement sur des cas particuliers. Et n'oubliez pas les docstrings !

EXERCICE 1

On reprend ici l'exercice n° 6 du TP n° 3 en utilisant des fonctions. Définir en PYTHON :

1. une fonction `div_par_5(n)` qui vérifie si `n` est divisible par `5`, c'est-à-dire qui renvoie le booléen `True` si `n` est divisible par `5` et le booléen `False` sinon ;
2. une fonction `multiples(m, n)` qui vérifie si les entiers m et n sont tels que l'un est multiple de l'autre ;
3. une fonction `meme_signe(m, n)` qui vérifie si les entiers m et n sont de même signe ;
4. une fonction `tous_meme_signe(m, n, p)` qui vérifie si les trois entiers m, n et p sont de même signe. *On réutilisera la fonction `meme_signe`.*

Faire de même avec l'exercice n° 7 du TP n° 3.

EXERCICE 2

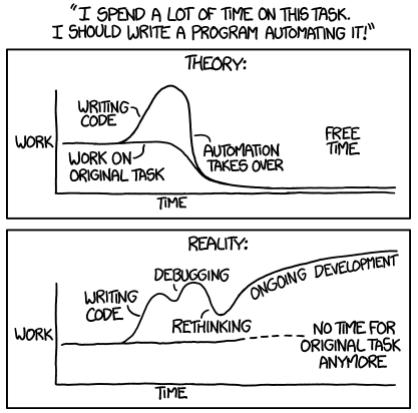
Écrire une fonction `bissextile` qui vérifie si une année est bissextile ou non. Une année est bissextile si elle est divisible par 4, sauf pour les siècles qui doivent être divisibles par 400.

EXERCICE 3

Écrire une fonction `jours_vers_secondes` qui prend en arguments un nombre entier de jours, d'heures, de minutes et de secondes, et qui renvoie le nombre entier de secondes correspondant.

EXERCICE 4

Écrire une fonction `secondes_vers_jours` qui prend en arguments un nombre entier de secondes et renvoie le tuple `(jours, heures, minutes, secondes)` correspondant. On pourra utiliser les opérateurs `//` et `%` ou la fonction `divmod`.



https://xkcd.com/1319/