Haute École Bruxelles-Brabant
École Supérieure d'Informatique
Bachelor en Informatique

2023 – 2024
DEV1 – DEV1L

DEV1 – DEV1L – Laboratoires Python

TD 05 – Fonctions

Nous allons à présent nous initier à l'écriture de nos propres fonctions en Python.

Table des matières

Version du document
17 septembre 2023

Rappel

Vous devez absolument avoir fait et compris les td précédents *avant* d'aborder celui-ci.

1 Les fonctions en Python

Voici un exemple de script Python définissant et utilisant une fonction `perimetre_cercle` calculant le périmètre d'un cercle de rayon passé en paramètre :

fonction.py

```
1
2  # *** Les fonctions en Python ***
3
4  # Une fonction calculant le périmètre d'un cercle de rayon donné:
5  def perimetre_cercle(rayon):
6      return 2*3.14*rayon
7
8
9  r = float(input("Entrez la longueur (en cm) du rayon du cercle: "))
10
11 perimetre = perimetre_cercle(r)
12
13 print("Le périmètre du cercle de rayon",r,"vaut",perimetre,"centimètres")
```

Exercice 1 La notion de fonction

Lancez le script Python fourni ci-dessus trois fois de suite pour calculer le périmètre d'un cercle de rayon `2`, `5.5` et `345.28` centimètres respectivement.

Mesure du rayon	Périmètre du cercle
1	6.28
1.5	9.42
345.28	2168.3584

La syntaxe pour la déclaration d'une fonction prenant un paramètre et renvoyant une valeur est donc :

fonctionsyntaxe.py

```
1  # Déclation d'une fonction prenant un paramètre et renvoyant une valeur
2  def nom_de_la_fonction(parametre):
3      #code de la fonction
4      return valeur_de_retour
```

Exercice 2 La notion de fonction (suite)

Écrivez un script Python définissant la fonction `perimetre_cercle` ci-dessus et affichant à l'écran le résultat des deux appels suivants :

Appel de fonction	Résultat ?
`perimetre_cercle(-6)`	-37.68
`perimetre_cercle('bonjour !')`	could not convert string to float: '"bonjour !"'

Type de données des paramètres

Comme vous avez pu le constater lors de l'exercice précédent, et contrairement à d'autres langages (comme Java), Python n'effectue *aucune vérification* sur les types de données passés en paramètre à une fonction ! Si les données passées à la fonctions sont incorrects, la fonction peut renvoyer un résultat absurde (comme dans le premier appel de l'exercice précédent) ou même déclencher une erreur (comme dans le second appel).

Faites également très attention à l'endroit ou se trouve votre instruction `return`. Analysons par exemple le code suivant :

return.py

```
1
2  # *** L'instruction return en Python ***
3
4  def afficher(x):
5      print(x)
6      return 0
7      print("Les fonctions en Python")
8
9
10 afficher(5)
11 afficher('salut')
12 afficher([1,2,3])
```

Exercice 3 **Code mort**

Exécutez le script ci-dessus. Qu'affiche ce script à l'écran ? Le script affiche :

```
5
salut
[1, 2, 3]
```

La chaîne de caractère `"Les fonctions en Python"` n'est jamais affichée. Pourquoi ? Votre réponse :

```
print("Les fonctions en Python ne s'affichera pas car après le return")
```

2 Découpe en fichier

L'exemple de la section précédente comportait un seul fichier avec la fonction et le code appelant cette fonction. Une règle de bonne pratique est cependant de placer ses fonctions dans un (des) module(s) et d'inclure ceux-ci dans un fichier contenant votre programme principal qui fera alors appel aux fonctions, comme illustré ci-dessous :

cercle.py

```python

# Code du module contenant des fonctions

# Une fonction calculant le périmètre d'un cercle de rayon donné:
def perimetre_cercle(rayon):
    return 2*3.14*rayon

# Une fonction calculant l'aire d'un cercle de rayon donné:
def aire_cercle(rayon):
    return 3.14*rayon*rayon


# Eventuelles autres fonction en rapport avec le cercle ...
```

main.py

```python
# On importe le module désiré ...
import cercle

r = float(input("Entrez la longueur (en cm) du rayon du cercle: "))

# ... et on fait appel aux fonctions nécessaires
perimetre = cercle.perimetre_cercle(r)

print("Le périmètre du cercle de rayon",r,"vaut",perimetre,"centimètres")
```

Exercice 4 **L'aire d'un carré**

Écrivez :

1. un module `carre.py` contenant une fonction `aire_carre` qui prend en paramètre un nombre `x` et qui renvoie l'aire du carré de côté `x` cm.
2. un programme principal `main.py` utilisant cette fonction de façon à demander un nombre positif `x` à l'utilisateur et afficher l'aire du carré associé à l'écran. Si le nombre entré par l'utilisateur n'est pas positif, le script affiche un message d'erreur.

Découpe en modules

Soyez attentif dans la suite des exercices (de ce TD ou des suivants) à *toujours* découper votre code en modules et programme principal. Nous ne le préciserons pas à chaque exercice, mais la demande est implicite.

3 Fonctions avec un nombre quelconque de paramètres

Une fonction peut admettre un nombre quelconque de paramètres en entrée (zéro, un, deux, ...) et peut aussi faire appel à d'autres fonctions. Le script suivant utilise deux fonctions pour tirer trois lettres de l'alphabet au hasard. Ces lettres peuvent alors par exemple être utilisées pour développer un jeu, comme le Scrabble [1]. Remarquez l'utilisation de la fonction `randint` dans la fonction `hasard` et des fonctions `len` et `sorted` dans la fonction `lettres`.

1. voir : https://fr.wikipedia.org/wiki/Scrabble

lettres.py

```python
import random

# Retourne un nombre aléatoire entre a et b (appelée par la fonction lettres())
def hasard (a,b):
    return random.randint(a,b)

# Tire 3 lettres de l'alphabet au hasard (répétitions possibles) et les renvoient
def lettres ():
    alphabet = 'abcdefghijklmnopqrsyuvwxyz'
    nb_lettres = len(alphabet)

    lettre1 = alphabet[hasard(0,nb_lettres-1)]
    lettre2 = alphabet[hasard(0,nb_lettres-1)]
    lettre3 = alphabet[hasard(0,nb_lettres-1)]

    return sorted([lettre1,lettre2,lettre3])
```

main1.py

```python
import lettres


mes_lettres = lettres.lettres()
print("Voici trois lettres de l'alphabet tirées au hasard:")
print(mes_lettres[0], mes_lettres[1], mes_lettres[2])
```

Exercice 5 Les couleurs aléatoires

Inspirez-vous du code précédent pour écrire une fonction `couleur` qui renvoie une couleur au hasard parmi les chaînes de caractères :

`les_couleurs = ['rouge', 'vert', 'jaune', 'bleu', 'blanc', 'noir' ]`

Mettez cette fonction en application pour écrire un script qui affiche une couleur tirée au hasard à l'écran.

Exercice 6 Les cartes aléatoires

Écrivez une fonction `tirer_carte` qui permet de tirer une carte au hasard dans un jeu de 52 cartes. Une carte d'un tel jeu contient une valeur choisie parmi :

`les_valeurs = [2,3,4,5,6,7,8,9,10,'valet','dame','roi','as']`

ainsi qu'une couleur, choisie parmi :

`les_couleurs = ['pique','trèfle','cœur','carreau' ]`

La fonction `tirer_carte` doit renvoyer une liste de deux éléments contenant la valeur et la couleur de la carte. Voici un exemple de valeur de retour possible : `[9, 'carreau' ]`.

4 Fonctions renvoyant un booléen

Une fonction peut aussi renvoyer un booléen, comme illustré ci-dessous :

nombres.py

```python
def est_pair (x):
    return x%2 == 0
```

main2.py
```python
import nombres

x = int(input('Entrez un nombre entier : '))

if (nombres.est_pair(x)):
  print('Ce nombre est pair')
else:
  print('Ce nombre est impair')
```

La fonction `est_pair` renvoie `True` si le nombre est pair, et `False` sinon. La valeur de retour de cette fonction peut alors par exemple être utilisée au sein d'une instruction if (comme ci-dessus).

Exercice 7 Divisibilité

Écrivez une fonction `est_divisible(a,b)` qui renvoie `true` si un des deux nombres divise l'autre, et `false` sinon. Par exemple, `est_divisible(2,3)` renvoie `false` tandis que `est_divisible(4,8)`, `est_divisible(8,4)` et `est_divisible(15,3)` renvoient `true`.

Exercice 8 Inclusion de chaînes de caractère

Écrivez une fonction `contient(texte1,texte2)` qui renvoie `True` si `texte1` est contenu dans `texte2` et `False` sinon. Par exemple, `contient('chat','Le chat gris')` renvoie `True` tandis que `contient('chien', 'Le chat gris')` renvoye `False` (Aide : utilisez simplement l'opérateur `in`).

5 Les fonctions qui ne renvoient aucune valeur

Une fonction en Python ne renvoie pas obligatoirement une valeur : il suffit d'omettre le mot clé `return` dans le code de celle-ci. Voici un exemple de scripts utilisant de telles fonctions pour afficher une ligne d'étoile de longueur donnée à l'écran :

affichage.py
```python
# Fonction qui affiche n étoiles à l'écran
def afficher_etoiles (n):
    print(n*'*')
    # Aucun return: la fonctions se contente d'afficher des étoiles
```

main3.py
```python
import affichage

# Affiche 5 étoiles
affichage.afficher_etoiles(5)
# Affiche 10 étoiles
affichage.afficher_etoiles(10)
```

Exercice 9 La ligne de caractères

Écrivez une fonction `affichage(n,car)` qui affiche `n` fois le caractère `car` à l'écran. Par exemple, l'appel de fonction `affichage(10,'-')` affiche

```
----------
```

à l'écran.

6 Exercices récapitulalitfs

Exercice 10 Le volume de la sphère

Écrivez une fonction `volume_sphere(r)` qui renvoie le volume d'une sphère de rayon r passé en paramètre (faites une recherche sur internet si vous ne connaissez plus la formule permettant de calculer le volume d'une sphère).

Exercice 11 La moyenne

Écrivez une fonction `moyenne(a,b,c)` qui renvoie la moyenne des trois nombres a, b et c. Ecrivez un script utilisant cette fonction pour calculer la moyenne de trois nombres entrés au clavier par l'utilisateur.

Exercice 12 L'adresse e-mail

Écrivez une fonction `adresse_valide(adresse)` qui vérifie la validité de l'adresse e-mail. Pour simplifier, nous considérons ici qu'une adresse e-mail est valide si :

1. elle contient une fois le symbole arobase *'@'*,
2. elle contient au moins un point *'.'*,
3. le premier caractère et le dernier caractère ne sont ni *'@'* ni *'.'*

Par exemple, `adresse_valide(charlot@gmail.be)` renvoie `True` tandis que `adresse_valide(charlot)` et `adresse_valide(@toto.be)` renvoient `False`. Ecrivez un script utilisant cette fonction pour vérifier la validité d'une adresse e-mail entrée au clavier par l'utilisateur : si l'utilisateur entre une mauvais adresse e-mail, le script affiche un message d'erreur.

Exercice 13 La boîte

Écrivez une fonction `afficher_boite(n)` qui affiche à l'écran une boite vide (= un rectangle avec un bord mais d'intérieur vide, voir l'illustration ci-dessous) constitué de `n` étoiles de largeur et de 4 étoiles de hauteur.

Par exemple, l'appel de fonction `afficher_boite(10)` affiche

```
**********
*        *
*        *
**********
```

à l'écran.

7 En résumé ...

Principaux points de matière du TD

Voici les principaux points abordés lors de ce TD. Vous devez absolument être à l'aise avec ceux-ci avant d'aborder la prochaine séance d'exercice.

1. Ecrire une fonction en Python en utilisant le mot clé `def`.
2. Découper son script en plusieurs fichiers (modules et programme principal).
3. Utiliser le mot clé `return` pour renvoyer une valeur au programme appelant.
4. Faire appel à une fonction en lui passant des paramètres, utiliser l'éventuelle valeur de retour de la fonction.
5. Ecrire une fonction Python résolvant un problème donné.