LS4 - Travaux pratiques 2

27 janvier 2014

1 Prendre le langage en main

Les exercices suivants portent sur les chaînes de caractères (structure de donnée `string`) et les tableaux mutables (structure de donnée `list`). Vous pouvez vous référer à la librairie standard de python stdtypes [1].

1.1 Chaînes de caractères

Question 1

Déterminer les valeurs des variables à l'issue des lignes de commandes suivantes. Donner les chaînes en utilisant les guillemets simples ' '

```
h = "Hello "
w = "World "
hw = h + w + "!"
l = len(h)
hhh = 3*h
lll = len(hhh)
```

hw	:
l	:
hhh	:
lll	:

```
s = "Tranches ou Slices"
t = s[1:4]
c = s[0]
u = s[0::3]
uu = s[0:3]
uuu = s[-3:0]
uuuu = s[-3:]
r = s[-1::-1]
rr = s[-1:0:-1]
rrr = s[::-1]
```

1. http://docs.python.org/3/library/stdtypes.html#sequence-types-list-tuple-range

```
t      :
c      :
u      :
uu     :
uuu    :
uuuu   :
r      :
rr     :
rrr    :
```

```
s = 'Test'
b = 'e' in s
bb = 's' not in s
```

```
b    :
bb   :
```

Question 2

Écrire des fonctions :

1. `prefixe(s, n)` qui retourne le préfixe de longueur `n` de la chaîne `s` s'il existe et `False` sinon.
Par exemple :
`prefixe('',0)` s'évalue en `''`,
`prefixe('prefixe',3)` s'évalue en `'pre'`,
`prefixe('a',0)` s'évalue en `''`,
`prefixe('a',1)` s'évalue en `'a'`,
`prefixe('a',3)` s'évalue en `False`.
2. `suffixe(s, n)` qui retourne le suffixe de longueur `n` de la chaîne `s` s'il existe et `False` sinon.
Par exemple :
`suffixe('', 0)` s'évalue en `''`,
`suffixe('suffixe', 3)` s'évalue en `'ixe'`,
`suffixe('a', 0)` s'évalue en `''`,
`suffixe('a', 3)` s'évalue en `False`.
3. `miroir(s)` retourne la chaîne `s` dans l'ordre inverse.
Par exemple :
`miroir('reflet')` s'évalue en `'telfer'`,
`miroir('')` s'évalue en `''`
4. `palindrome(s)` retourne le booléen `True` si la chaîne `s` est un palindrome et `False` sinon.
Par exemple :
`palindrome('mon nom')` s'évalue en `True`,
`palindrome('my name')` s'évalue en `False`,
`palindrome('')` s'évalue en `True`,
`palindrome('a')` s'évalue en `True`.
5. `centre(s)` retourne la ou les deux lettres centrales de la chaîne `s` selon la parité de sa longueur.
Par exemple :
`centre('ab')` s'évalue en `'ab'`,
`centre('abc')` s'évalue en `'b'`,
`centre('centre')` s'évalue en `'nt'`.
6. `pairs(s)` retourne la chaîne des lettres d'indice pair de `s`.
Par exemple :
`pairs('ma chaine')` s'évalue en `'m hie'`,
`pairs('')` s'évalue en `''`,
`pairs('abc')` s'évalue en `'ac'`.

Question 3 : Boucles sur les chaînes

1. Déterminer les valeurs des expressions suivantes à l'issue des lignes de commandes suivantes.

```
s = "on peut iterer sur les chaines"
t = ''
for c in s:
    t = t + ' ' + c
```

t :

```
s = "iterer sur les caracteres et les indices des chaines"
for i,c in enumerate(s):
    if c == 'e':
       j = i
k = s.index('e')
```

s[j]	:
s[j+1]	:
s[k]	:
s[k+2]	:

Question 4

Écrire des fonctions :

1. `sans_e` qui renvoie la chaîne de caractères en argument sans les 'e'.

Par exemple :

`sans_e('Test')` s'évalue en `'Tst'`,
`sans_e('Bonjour')` s'évalue en `'Bonjour'`,
`sans_e('eee')` s'évalue en `''`,
`sans_e('')` s'évalue en `''`.

2. Écrire une fonction `compte_espace` qui renvoie le nombre d'espaces dans la chaîne de caractères passée en argument.

Par exemple :

`compte_espace('un espace')` s'évalue en `1`,
`compte_espace("pasd'esapce")` s'évalue en `0`,
`compte_espace(' espace au debut')` s'évalue en `3`,
`compte_espace('espace a la fin ')` s'évalue en `4`.

Remarque : Ne pas utiliser l'instruction `s.count(' ')`.

Question 5

Proposer une correction au programme suivant qui teste les palindromes sans tenir compte des espaces Par exemple :

`palindrome_espace_espace(' a man a plan a canal panama')` s'évalue en `True`,
`compte_espace('le sel')` s'évalue en `True`.
`compte_espace('le poivre')` s'évalue en `False`.

```
def palindrome_espace s:
    s.replace('',' ')
    return s = s[:-1]
```

1.2 Tableaux Mutables

Question 1

Déterminer les valeurs des expressions à l'issue des instructions suivantes.

```
L = [0,1,2,3,4,5]
a1 = L[0]
a2 = L[-1]
S1 = L[2:3]
S2 = L[1:]
S3 = L[-1::-1]
```

L	:
a1	:
a2	:
S1	:
S2	:
S3	:

```
M = ['a','b','c','d']
v1 = M[3]
M.append('e')
M.extend([1,2,3])
del(M[3])
v2 = M[3]
for i,e in enumerate(M):
  v3 = i
```

M	:
v1	:
v2	:
v3	:

```
N = [1,2,3]
NN = ['a','b']
i = id(N)
ii= id(NN)

j = id(N+NN)
N[-1:]=NN
j1 = id(N)

for e in NN:
  N.append(e)
j2 = id(N)

N.extend(NN)
j3 = id(NN)

b1 = (i==ii)
b2 = (i==j)
b3 = (ii==j)
b4 = (i==j1)
b5 = (i==j2)
b6 = (i==j3)
```

N	:
NN	:
b1	:
b2	:
b3	:
b4	:
b5	:
b6	:

Question 2

Mutation de tableaux mutables. Quelles sont les valeurs des variables suivantes ?

```
tab = [1, 34, 5, 4, 7, 8, 93]
t1 = [2*i for i in tab]
t2 = [i for i in tab if i%2 == 0]
t3 = [2*i for i in tab if i%2 == 0]
```

t1	:
t2	:
t3	:

```
tab = 'ceci est un tableau de mots'.split()
t = [u+v for u in tab for v in tab if u != v]
```

tab	:
t	:

```
alphabet = 'ab'
mots_longueur_3 = [i+j+k for i in alphabet for j in alphabet for k in alphabet]
```

mots_longueur_3 :

Question 3 : Tableaux et boucles

On suppose que les tableaux manipulés dans cet exercice sont des tableaux d'entiers. Pour résoudre les exercices suivants, vous utiliserez des boucles sur les tableaux mutables et/ou le mécanisme de compréhension.

1. Écrire une fonction `somme(T)` qui retourne la somme des éléments de `T`.

Par exemple :
`somme([1,2,3])` s'évalue en `6`,
`somme([])` s'évalue en `0`,
`somme([0,4,4,4])` s'évalue en `12`.

2. Écrire une fonction `carre(T)` qui retourne le tableau des éléments de `T` élevés au carré.

Par exemple :
`carre([1,2,3])` s'évalue en `[1,4,9]`.
`carre([])` s'évalue en `[]`.
`carre([0,4,4,4])` s'évalue en `[0,16,16,16]`.

3. Écrire une fonction `pair(T)` qui retourne le tableau des éléments d'indice pair de `T`.

Par exemple :
`pair([1,2,3,4,5,6,7])` s'évalue en `[1,3,5,7]`.
`pair([])` s'évalue en `[]`.
`pair([0,1,2,3,4,4,4])` s'évalue en `[0,2,4,4]`.

4. Écrire une fonction `ma_map(f, T)` qui, étant donnée une fonction `f` et un tableau `T`, retourne le tableau constitué des images des éléments de `T` par `f`.

Par exemple :
`ma_map(sans_e, ["un mot", "un autre", "et encore un autre"])` s'évalue en `['un mot', 'un autr', 't ncor u`
`ma_map(len, [[],[1,2,3],["bla"]])` s'évalue en `[0,3,1]`.

5. Écrire une fonction `occ` qui prend en argument une chaîne de caractères et une lettre et qui renvoie le tableau des indices des occurences de cette lettre dans la chaîne.

Par exemple :
`occ('h', "ma chaine")` s'évalue en `[4]`.
`occ('c', "une phrase avec 'c'")` s'évalue en `[14, 17]`.
`occ('c', "une phrase sans")` s'évalue en `[]`.

1.3 Boucles

Question 1

Quel est le résultat renvoyé par la fonction suivante ?

```
def f(n) :
  u = 1
  for x in range(n) :
    u = u * (x+1)
  return u
```

f(0) :
f(2) :
f(3) :
f(5) :
f(n) :

Question 2

Donner le contenu du tableau mutable `T` à l'issue des lignes de commandes suivantes.

```
T = []
n = 2
for i in range(n):
    if i != 0:
       T.append(i)

for i in range(1, 13, 2 * n):
    for j in range(0, n):
        T.append([i, j])

for i in range(1, len(T), 5):
    for j in range(0, i, 2):
        for k in range(0, i, 2):
            if (i > j) and (j > k):
                T.append([i, j, k])
```

T :

Question 3 : Erreur de flottants

Écrire deux fonctions `pi_2_6a` et `pi_2_6d` qui prennent en entrée un entier `n` et retournent la somme des 1/kˆ2 de 1 à n en augmentant et en diminuant. Donner la valeur du tableau `T` contenant les différences `pi_2_6a(n)-pi_2_6d(n)` pour `n` parcourant les puissances de 10 entre 1 et 10^6.

T :

Question 4

Corriger les instructions suivantes et donner la valeur des variables.

```
l = [i for i in range(10)]
for k in L
    c += 1
    s += i
m = s/c
```

c :
s :
l :
m :

Corriger la fonction suivante afin que par exemple :
`double([])` s'évalue en `[]`,
`double([1,2,3])` s'évalue en `[1,2,3,1,2,3]`,
`double([0,0])` s'évalue en `[0,0,0,0]`.

```
def double(l):
  for i in l
  l.append(i)
return l
```

2 Programmer et Tester

2.1 Scripts et modules

Télécharger le script suivant, le compléter, lui donner les droits adequats et le tester. Faire en sorte que l'auteur et le copyright ne s'affichent que quand le script est exécuté dans un terminal et pas quand il est importé comme module, en utilisant la variable __name__ [2].

Voir my_name.py. [3]

2.2 Scripts, Arguments et exceptions

Cet exercice vous permettra de :
- Utiliser les modules Sys pour gérer les options d'un script (référence) [4].
- Utiliser des variables déjà définies (__name__) (référence) [5]
- Utiliser des exceptions sur les listes (référence) [6]

Question

Télécharger le script suivant et le compléter en tenant compte de la documentation qui décrit son comportement. Une fois votre script terminé et testé sur votre machine, le charger sur le serveur pour lui faire passer des tests.

Voir sans.py. [7]

2.3 Module de test

Le but de cet exercice est de vous donner une idée de comment sont testés vos programmes sur Hacking Dojo. Vous utiliserez les tableaux mutables, des boucles et des fonctions. Vous aurez aussi à gérer des erreurs potentielles gràce au mécanisme d'exception [8].

Question 1

Écrire une fonction `Itest` qui prend en arguments une fonction et un tableau d'entrées, évalue la fonction sur chacune des entrées et renvoie True si tous les tests sont passés, sinon renvoie False et imprime le premier test raté.

Tester votre fonction Itest sur une fonction qui marche et une qui ne marche pas. Remarque, on ne peut pas tester des boucles infinies.

Voici une fonction qui marche :

```
def copie_o(l):
  r = []
  for i in l:
    r.append(i)
  return r
```

Voici une fonction qui ne marche pas :

2. http://docs.python.org/3/tutorial/modules.html
3. Fichier fourni avec le sujet.
4. http://docs.python.org/3/library/sys.html
5. http://docs.python.org/3/tutorial/modules.html
6. http://docs.python.org/3.3/tutorial/errors.html#handling-exceptions
7. Fichier fourni avec le sujet.
8. http://docs.python.org/3.3/tutorial/errors.html

```
def copie_n(l):
   for i in range(len(l)+1):
      r.append(i)
   return r
```

Voici un tableau d'entrées : `T = [[1,3,5,2,3],[],['a','b']]`

Question 2

Écrire une fonction "IOtest" qui prend en arguments une fonction python et une liste de couples input/output et retourne True si tous les tests sont passés, imprime la liste des tests ratés et retourne False si des tests sont ratés et imprime l'erreur si la fonction plante.

Tester sur une fonction les fonctions suivantes :

```
def copie_o(l):
     r = []
     for i in l:
             r.append(i)
     return r
```

```
def copie_n(l):
     r=l
     for i in range(len(l)+1):
             r.append(i)
     return r
```

Avec le tableau d'inputs/outputs : `T=[([1,3,5,2,3],[1,3,5,2,3]),([],[]),(['a','b'],['a','b'])]`

Question 3

Créer un module `my_test.py` contenant les fonctions précédentes. Compléter et corriger le fichier ci-dessous. Puis le charger pour lui faire passer des tests.

Voir TestTP2.py. [9]

9. Fichier fourni avec le sujet.