Informatique tronc commun – TP 3 et TP 4

5 octobre 2016

1. **Lisez attentivement tout l'énoncé avant de commencer.**
2. Commencez la séance en créant un dossier au nom du TP dans le répertoire dédié à l'informatique de votre compte.
3. Après la séance, vous devez rédiger un compte-rendu de TP et l'envoyer au format électronique à votre enseignant.
4. Vous rendrez un compte-rendu pour chaque séance sous forme d'un fichier d'extension `.py`, en respectant exactement les spécifications données plus bas.
5. Ce TP est à faire en binôme, vous ne rendrez donc qu'un compte-rendu pour deux.
6. Ayez toujours un crayon et un papier sous la main. Quand vous réfléchissez à une question, utilisez les !
7. Vous devez être autonome. Ainsi, avant de poser une question à l'enseignant, merci de commencer par :
   - relire l'énoncé du TP (beaucoup de réponses se trouvent dedans) ;
   - relire les passages du cours [1] relatifs à votre problème ;
   - effectuer une recherche dans l'aide disponible sur votre ordinateur (ou sur internet) concernant votre question.

   Il est alors raisonnable d'appeler votre enseignant pour lui demander des explications ou une confirmation !

Le but de ce TP est d'apprendre à maîtriser les bases de la programmation en Python.

TP03 : Vous rendrez les questions 8, 17, 24, 26, 28, 38.

TP04 : Vous rendrez toutes les questions de la partie 6.

Ces exercices ne sont pas forcément évidents, vous êtes libres de vous entraîner sur les autres exercices, qui sont de difficulté croissante. À vous de gérer votre progression !

Instructions de rendu

Attention : suivez précisément ces instructions. Votre fichier portera un nom du type `tpXX_berne_zannad.py`, où `XX` est à remplacer par le numéro du TP et les noms de vos enseignants par ceux des membres du binôme. Le nom de ce fichier ne devra comporter ni espace, ni accent, ni apostrophe. Dans ce fichier, vous respecterez les consignes suivantes.

1. Dans le cas fort improbable où vous ne vous en souviendriez pas.

— Écrivez d'abord en commentaires (ligne débutant par #), le titre du TP, les noms et prénoms des étudiants du groupe.
— Commencez chaque question par son numéro écrit en commentaires.
— Les questions demandant une réponse écrite seront rédigées en commentaires.
— Les questions demandant une réponse sous forme de fonction ou de script respecteront pointilleusement les noms de variables et de fonctions demandés.

1 Expressions et types simples en Python

Q1 Évaluer les expressions suivantes en repérant auparavant celles qui donnent des résultats de type `int`.

a) `4+2`
b) `25-3`
c) `-5+1`
d) `117*0`
e) `6*7-1`
f) `52*(3-5)`
g) `5*(-2)`
h) `22/(16-2*8)`
i) `42/6`
j) `18/7`
k) `(447+3*6)/5`
l) `0/0`

Q2 Calculer les restes et les quotients des divisions euclidiennes suivantes :

a) $127 \div 8$
b) $54 \div 3$
c) $58 \div 5$
d) $58 \div (-5)$
e) $-58 \div 5$
f) $-58 \div (-5)$
g) $17583 \div 10$
h) $17583 \div 100$
i) $17583 \div 10^4$
j) $(2^7 + 2^4 + 2) \div 2^5$
k) $(2^7 + 2^4 + 2) \div 2^7$
l) $(2^7 + 2^4 + 2) \div 2^{10}$

Calculer les nombres suivants avec une expression Python en repérant auparavant ceux qui donnent un résultat de type `int`.

a) 3^5
b) 2^{10}
c) $(-3)^7$
d) -3^7
e) 5^{-2}
f) $7^{(5^4)}$
g) $\left(7^5\right)^4$
h) 5^{7+6}
i) $5^7 + 6$
j) $2^{(10^4)}$

Q3 Évaluer les expressions suivantes.

a) `4.3+2`
b) `2.5-7.3`
c) `42+4.`
d) `42+4`
e) `42.+4`
f) `12*0.`
g) `11.7*0`
h) `2,22/(1.6-2*0.8)`
i) `42/6`
j) `1,8/7`
k) `(447+3*6)/5`
l) `0/0`

Q4 Calculer, sans utiliser la fonction `sqrt` ni la division flottante `/`, les nombres suivants.

a) $\dfrac{1}{7,9}$
b) $\sqrt{6,2}$
c) $\dfrac{1}{\sqrt{3,5}}$
d) $2\sqrt{2}$

De base, on ne peut réaliser que des calculs élémentaires avec Python. Cependant, il est possible d'avoir accès à des possibilités de calcul plus avancées en utilisant une *bibliothèque*. Par exemple, la bibliothèque `math` permet d'avoir accès à de nombreux outils mathématiques. On peut donc taper

```
from math import sqrt, log, exp, sin, cos, tan, pi, e
```

pour avoir accès à toutes ces fonctions.
Calculer les nombres suivants (on n'hésitera pas à consulter l'aide en ligne).

a) e^2 *c)* $\cos\left(\frac{\pi}{5}\right)$ *e)* $\ln 2$ *g)* $\log_2 10$

b) $\sqrt{13}$ *d)* $e^{\sqrt{5}}$ *f)* $\ln 10$ *h)* $\tan\left(\frac{\pi}{2}\right)$

Q5 Les expressions suivantes sont-elles équivalentes ?

a) `8.5 / 2.1` *b)* `int(8.5) / int(2.1)` *c)* `int(8.5 / 2.1)`

Et celles-ci ?

a) `float(8 * 2)` *b)* `8 * 2` *c)* `8. * 2.`

Prévoir la valeur des expressions suivantes puis vérifier cela (avec IDLE).

a) `1.7 + 1.3` *c)* `2. - 1` *e)* `(2 - 1).` *g)* `4 / (9 - 3**2)`

b) `2 - 1` *d)* `2 - 1.` *f)* `.5 + .5` *h)* `4 / (9. - 3**2)`

Q6 Déterminer de tête la valeur des expressions suivantes avant de le vérifier (avec IDLE).

a) `0 == 42` *h)* `2*True + False` *o)* `(2 == 3-1) or (1/0 == 5)`

b) `1 = 1` *i)* `-1 <= True` *p)* `(1/0 == 5) or (2 == 3-1)`

c) `3 == 3.` *j)* `1 == True` *q)* `True or True and False`

d) `0 != 1` *k)* `False != 0.` *r)* `False or True and False`

e) `0 < 1` *l)* `True and False` *s)* `not (1 == 1 or 4 == 5)`

f) `4. >= 4` *m)* `True or False` *t)* `(not 1 == 1) or 4 == 5`

g) `0 !< 1` *n)* `True or True` *u)* `not True or True`

Q7 Dans IDLE, cliquer sur File/New File. Une nouvelle fenêtre apparaît. Dans cette fenêtre, taper les lignes suivantes.

```
3*2
print(2*3.)
17*1.27
```

Enregistrer le document produit puis, toujours dans cette fenêtre, exécutez-le (touche F5). Observez le résultat dans l'interpréteur interactif. Modifiez les instructions pour que tous les résultats de calcul s'affichent dans l'interpréteur interactif.

Q8 Dans chaque cas, indiquez le type que vous utiliseriez pour modéliser les grandeurs suivantes dans leur contexte scientifique usuel.

a) La taille d'un individu en mètres.

b) Le tour de taille d'un manequin, en millimètres.

c) Le nombre d'Avogadro

d) Le nombre de Joules dans une calorie.

e) Le nombre de secondes dans une année.

f) Le plus grand nombre premier représentable avec 20 chiffres en écriture binaire.

2 Expressions et types composés en Python

Q9 Prévoir les résultats des expressions suivantes, puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a) `(1,2)`
b) `(1)`
c) `(1,)`
d) `(,)`
e) `()`
f) `()+()`
g) `()+() == ()`
h) `(1,2)+3`
i) `(1,2)+(3)`
j) `(1,2)+(3,)`
k) `(1,2)+(3,4,5)`
l) `len((1,7,2,"zzz",[]))`
m) `len(())`
n) `len(("a","bc")+("cde",""))`

Q10 Pour chaque séquence d'instruction, prévoir son résultat puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a)
```
t = (2,"abra",9,6*9,22)
print(t)
t[0]
t[-1]
t[1]
t[1] = "cadabra"
```

b)
```
res = (45,5)
x,y = res
(x,y) == x,y
(x,y) == (x,y)
print x
print(y)
x,y = y,x
print(y)
```

c)
```
v = 7
ex = (-1,5,2,"","abra",8,3,v)
5 in ex
abra in ex
(2 in ex) and ("abr" in ex)
v in ex
```

Q11 Prévoir les résultats des expressions suivantes, puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a) `"abba"`
b) `abba`
c) `""`
d) `"" == " "`
e) `""+""`
f) `""+"" == ""`
g) `"May"+" "+"04th"`
h) `"12"+3`
i) `"12"+"trois"`
j) `len("abracadabra")`
k) `len("")`
l) `len("lamartin"+"2015")`

Q12 Pour chaque séquence d'instruction, prévoir son résultat puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a)
```
t = "oh oui youpi !"
print(t)
t[0]
t[-1]
t[1]
t[2]
t[1] = "o"
```

b)
```
ex = "abdefgh"
"a" in ex
a in ex
"def" in ex
"adf" in ex
```

Q13 Prévoir les résultats des expressions suivantes, puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a) `[1,2,3,"a"]`

b) `123a`

c) `[]`

d) `[]+[]`

e) `[]+[] == []`

f) `[1,2] + [5,7,9]`

g) `[0,0]+[0]`

h) `len(["a","b"])`

i) `len([])`

j) `len([[]])`

k) `len([[[]]])`

l) `len([0,0]+[1])`

Q14 Pour chaque séquence d'instruction, prévoir son résultat puis le vérifier grâce à l'interpréteur interactif d'IDLE.

a)
```
t = [1,2,3,4,5,6]
u = ["a","b","c","d"]
print(t+u)
t[0]
t[-1]
z = t[3]
print(z)
t.append(7)
print(t)
```

b)
```
ex = ["sin","cos","tan","log","exp"]
"log" in ex
log in ex
"l" in ex
z = ex.pop()
print(z)
z in ex
print(ex)
```

c)
```
  u = [1,2,3,4,5,6]
  L = u
  u = [1,2,3,42,5,6]
  print(L)
```

d)
```
u = [1,2,3,4,5,6]
L = u
u[3] = 42
print(L)
```

Q15 Calculer cette suite d'expressions.

```
[i for i in range(10)]
[compt**2 for compt in range(7)]
[j+1 for j in range(-2,8)]
```

Sur ce modèle, obtenir de manière synthétique :

a) la liste des 20 premiers entiers naturels impairs ;

b) la liste de tous les multiples de 5 entre 100 et 200 (inclus) ;

c) La liste de tous les cubes d'entiers naturels, inférieurs ou égaux à 1000.

d) une liste contenant tous les termes entre -20 et 5 d'une progression arithmétique de raison $0,3$ partant de -20.

Q16

a) Affecter à `v` la liste `[2,5,3,-1,7,2,1]`

b) Affecter à `L` la liste vide.

c) Vérifier le type des variables crées.

d) Calculer la longueur de `v`, affectée à `n` et celle de `L`, affectée à `m`.

e) Tester les expressions suivantes : `v[0]`, `v[2]`, `v[n]`, `v[n-1]`, `v[-1]` et `v[-2]`.

f) Changer la valeur du quatrième élément de `v`.

g) Que renvoie `v[1:3]` ? Remplacer dans `v` les trois derniers éléments par leurs carrés.

h) Que fait `v[1] = [0,0,0]` ? Combien d'éléments y a-t-il alors dans `v` ?

Q17 Quel type choisiriez-vous pour représenter les données suivantes ?

a) Le nom d'une personne.

b) L'état civil d'une personne : nom, prénom, date de naissance, nationalité.

c) Les coordonnées d'un point dans l'espace.

d) L'historique du nombre de 5/2 dans la classe de MP du lycée.

e) Un numéro de téléphone.

f) *Plus difficile :* l'arbre généalogique de vos ancêtres.

3 Variables

Q18 Dans les cas où c'est possible, affecter les valeurs aux variables correspondantes à l'aide de l'interpréteur interactif d'IDLE. On notera `var` ← `a` pour dire que l'on affecte la valeur `a` à la variable `var`.

a) `ArthurDent` ← `42`
b) `4` ← `0.`
c) `L` ← `[]`
d) `list` ← `[1,2,3]`
e) `int` ← `5`
f) `s` ← `""`
g) `True` ← `1`
h) `ok` ← `ok`
i) `x` ← `"x"`
j) `a` ← `1<0`
k) `lam` ← `1/0`
l) `or` ← `"xor"`

Q19 On considère les affectations `a` ← `-1.` et `b` ← `5`. Prévoir la valeur de chacune de ces expressions, puis le vérifier à l'aide de l'interpréteur interactif d'IDLE.

a) `a * a`
b) `a ** a`
c) `a ** a == a`
d) `a * b`
e) `a+b == 5`
f) `a+6>=b`
g) `b<100 | a**2==-1`
h) `b-3`

Q20 On part des affectations suivantes : `a` ← `5` et `b` ← `0`. Pour la suite d'instructions suivante, prévoir ligne à ligne le résultat affiché par l'interpréteur interactif de Python ainsi que l'état des variables. Le vérifier grâce à l'interpréteur interactif d'IDLE, en prenant soin de partir d'une nouvelle session.

```
a*b
x = a**b + a
print(x)
print(y)
z = x
x = 5
print(z)
a = a+a**b
print(a)
```

Q21 Affecter des valeurs toutes différentes aux variables `a`, `b`, `c` et `d`.
Á chaque fois, effectuer les permutations suivantes de manière naïve (c'est-à-dire, sans utiliser de `tuple`).

a) Échanger les contenus de `a` et de `b`.
b) Placer le contenu de `b` dans `a`, celui de `a` dans `c` et celui de `c` dans `b`.
c) Placer le contenu de `a` dans `d`, celui de `d` dans `c`, celui de `c` dans `b` et celui de `b` dans `a`.

Reprendre cet exercice en effectuant chaque permutation en une instruction à l'aide d'un `tuple`.

Q22

a) Affecter à la variable `mon_age` l'âge que vous aviez il y a 13 ans.
b) Écrire l'opération qui vous permet d'actualiser votre âge, tout en conservant la même variable.
c) Que donne l'interpréteur après exécution des expressions suivantes ? Pourquoi ?

```
mon-age = 18
2013_mon_age = 18
True = 18
```

d) À partir d'une nouvelle session d'IDLE, exécuter les expressions suivantes et commenter le résultat.

```
age = 5
age = Age + 14
```

4 Fonctions en Python

Q23 Ouvrir IDLE, écrire la fonction suivante dans un fichier, l'enregistrer, taper *run* (F5) puis utiliser la fonction dans l'interpréteur interactif d'IDLE. Décrire ensuite précisément ce que réalise cette fonction.

```
def split_modulo(n):
  """A vous de dire ce que fait cette fonction !"""
  return (n%2,n%3,n%5)
```

Q24 Écrire une fonction `moy_extr(L)` qui prend en argument une liste `L` et renvoie en sortie la moyenne du premier et du dernier élément de `L`.

Q25 Écrire une fonction `norme` qui prend en argument un vecteur de $\mathbb{R}^2$ donnée par ses coordonnées et renvoie sa norme euclidienne. Vous devrez spécifier clairement le type de l'argument à l'utilisateur via la *docstring*.

Q26 Écrire une fonction `lettre` qui prend en argument un entier `i` et renvoie la $\texttt{i}^{\text{e}}$ lettre de l'alphabet.

Q27 Écrire une fonction `carres` qui prend en argument un entier naturel `n` et qui renvoie la liste des `n` premiers carrés d'entiers, en commençant par 0.

Q28 On cherche à écrire une fonction prenant en argument une liste d'entiers et incrémentant de 1 le premier élément de cette liste.

a) Écrire une telle fonction `incr_sans_effet_de_bord`, qui ne modifie pas la liste initiale et renvoie en sortie une nouvelle liste.

b) Écrire une telle fonction `incr_avec_effet_de_bord`, qui modifie la liste initiale et ne renvoie rien en sortie (ponctuer par un `return None`).

5 Boucles IF, FOR, WHILE

Q29 Indenter de deux manières différentes la suite d'instruction suivante afin que la variable `t` contienne `True` pour une indentation, puis `False` pour l'autre. Le tester avec IDLE.

```
x = 0
y = 5
t = False
if x>=1:
t = True
if y <= 6:
t = True
```

Q30 Réécrire la suite d'instructions suivante de manière plus appropriée.

```
from random import randrange
n = randrange(100) # Un entier aléatoire entre 0 et 99
if n <= 10:
    print("Trop petit")
else:
    if n >= 50:
        print("Trop grand")
    else:
        print("Juste comme il faut")
```

Q31

a) Écrire une fonction `neg(b)` qui renvoie la négation du booléen `b` sans utiliser `not`.

b) Écrire une fonction `ou(a,b)` qui renvoie le ou logique des booléen `a` et `b` sans utiliser `not`, `or` ni `and`.

c) Écrire une fonction `et(a,b)` qui renvoie le et logique des booléen `a` et `b` sans utiliser `not`, `or` ni `and`.

Q32 Indenter de deux manières différentes la suite d'expressions suivante de manière à ce qu'à son exécution, le programme affiche soit la liste de tous les éléments de L inférieurs ou égaux à m, soit juste le dernier.

```
from random import randrange
L = [randrange(100) for i in range(100)] # 100 valeurs entre 0 et 99.
m = 50
for x in L:
if x <= m:
p = x
print(p)
```

Q33 Que fait la fonction suivante ? La corriger pour qu'elle coïncide avec le but annoncé.

```
def inv(n):
    """Somme les inverses des n premiers entiers naturels non nuls"""
    s = 0
    for k in range(n):
        x = 1/k
    s = s+x
    return s
```

Q34 Décrire ce que fait cette suite d'instructions.

```
from random import randrange
n = randrange(1000) # Un entier entre 0 et 999
L = [randrange(100) for i in range(n+1)] # n+1 valeurs entre 0 et 99.
p = 0
for x in L:
    if x <= 10:
        p = p + x**2
```

Et celle-ci ?

```
from random import randrange
n = randrange(1000) # Un entier entre 0 et 999
L = [randrange(100) for i in range(n+1)] # n+1 valeurs entre 0 et 99.
p = 0
for i in range(n+1):
    if L[i] <= 10:
        p = p + i**2
```

Q35 Écrire une suite d'instructions permettant de calculer la somme des racines carrées des cinquante premiers entiers naturels non nuls.

Q36 Écrire la suite d'instructions suivantes dans un fichier, l'enregistrer puis l'exécuter (F5). À l'instant qui vous convient, presser Ctrl+C.

```
a = 1
while a>0:
    a = a+1
```

Q37 Que fait la fonction suivante ? La corriger pour qu'elle coïncide avec le but annoncé.

```
def sqrt_int(n):
    """Renvoie la partie entière de la racine carrée de n"""
    s = 0
    while s**2 <= n:
        s = s+1
        s = s-1
    return s
```

Q38 Un taupin se lance dans un marathon d'exercices, mais se fatigue vite. Il réalise le i^{e} exercice en $\sqrt{i}$ minutes. Combien d'exercices arrive-t-il à faire en 4 heures ? Pour faciliter la correction, on affectera le résultat à la variable `nb_exos`.

6 Exercices avancés en Python

Q39 Écrire une fonction `racine(n)` prenant en argument un entier naturel `n` et renvoyant sa racine carrée comme un entier si c'est un carré parfait, comme un flottant sinon.

Q40 Dans le jeu de la bataille navale, un navire a ses extrémités sur les cases `a` et `b` (qui sont donc des couples d'entiers entre 0 et 9). Un joueur tire sur la case `x`. Écrire une fonction `touche(a,b,x)` qui renvoie un booléen indiquant si le navire est touché ou non.

Q41 Un banquier vous propose un prêt de 400 000 euros sur 40 ans «à 3% par an» — ce qui, dans le langage commercial des banquiers, veut dire $0,25\%$ par mois — avec des mensualités de $1431,93$ euros. Autrement dit, vous contractez une dette de 400 000 euros. Chaque mois, cette dette augmente de $0,25\%$ puis est diminuée du montant de votre mensualité. À la fin des 40×12 mensualités, il ne vous reste plus qu'à vous acquitter d'une toute petite dette, que vous rembourserez aussitôt.

a) Écrire une fonction `reste_a_payer(p, t, m, d)` retournant le montant de cette somme à rembourser immédiatement après le paiement de la dernière mensualité, où p est le montant total du prêt en euros, t son taux mensuel (ici $0,25 \times 10^{-2}$), m le montant d'une mensualité en euros et d la durée en années.

Dans le cas donné dans cet énoncé, vous devez trouver un montant restant d'un peu moins de $7,12$ euros.

b) Écrire une fonction `somme_totale_payee(p, t, m, d)` retournant la somme totale (mensualités plus le dernier paiement) que vous aurez payé au banquier.

c) Écrire une fonction `cout_total(p, t, m, d)` donnant le coût total du crédit, c'est-à-dire le total de ce que vous avez payé moins le montant du prêt.

Q42 Un banquier vous propose de vous prêter p euros par mois, à un taux de $12t\%$ par an — ce qui, dans le langage commercial des banquiers, veut dire $t\%$ par mois — avec des mensualités de m euros. Autrement dit, vous contractez une dette de p euros. Chaque mois, cette dette augmente de $t\%$ puis est diminuée du montant de votre mensualité. Lorsque votre dette, augmentée du taux, est inférieure à la mensualité, il suffit de régler le solde en une fois.

Écrire une fonction `duree_mensualite(p,t,m)` permettant de calculer le nombre de mensualités nécessaires au remboursement total du prêt.

Attention : que se passe-t-il si la mensualité est trop petite ?

Indice : dans le cas où le prêt est $p = 4 \times 10^5$*, le taux est* $t = 0,25 \times 10^2$ *et la mensualité* $m = 1431,93$*, on trouvera une durée de remboursement de* 480 *mois.*

Q43 Écrire une fonction `comb(p,n)` calculant $\binom{n}{p}$ (nombre de combinaisons de p éléments parmi n). On pourra bien entendu introduire une fonction auxiliaire [2].

Q44 On appelle suite de Fibonacci la suite F définie par $F_0 = 0$, $F_1 = 1$ et pour tout n, $F_{n+2} = F_{n+1} + F_n$.

Écrire une fonction `fib(n)` calculant et retournant F_n.

Pensez à vérifier le résultat de votre fonction en 0, 1 et en 5 (vous calculerez à la main F_5 *avant* de le faire calculer par votre fonction).

Q45 On pose $u_0 = 1$ et pour tout $n \in \mathbb{N}$,

$$u_{n+1} = \frac{1}{2}\left(u_n + \frac{n+1}{u_n}\right)$$

$$\text{et } v_n = \sum_{k=0}^{n} \frac{1}{u_k^5}$$

Écrire une fonction `f(n)` retournant la valeur de v_n.

On peut montrer que $(v_n)_{n\in\mathbb{N}}$ converge.

Vous pouvez calculer u_n pour de grandes valeurs de n.

Attention : on fera attention à ce que le calcul de `f` de demande pas trop de (re)calculs inutiles. Pour fixer les idées, vous pouvez considérer que `f(10**6)` doit être calculé en (largement) moins d'une minute.

2. C'est-à-dire, comme l'indique l'étymologie, une autre fonction dont le but sera de vous aider à répondre à la question.

Q46 Écrire une fonction `somme1` et une fonction `somme2` prenant en argument un entier n et calculant respectivement

$$\sum_{1\leq i,j\leq n} \frac{1}{i+j^2} \tag{1}$$

$$\sum_{1\leq i<j\leq n} \frac{1}{i+j^2} \tag{2}$$

Au besoin, on introduira des fonctions auxiliaires.