Informatique tronc commun – TP n° 3

4 octobre 2017

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`.

Instructions de rendu

Attention : suivez précisément ces instructions. Votre fichier portera un nom du type

`tp03_berne_durif.py`,

où les noms de vos enseignants sont à remplacer par ceux des membres du binôme. Le nom de ce fichier ne devra comporter ni espace, ni accent, ni apostrophe, ni majuscule. Dans ce fichier, vous respecterez les consignes suivantes.

- 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.

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

— 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 Dans chaque cas, indiquez le type que vous utiliseriez pour modéliser les grandeurs suivantes dans leur contexte scientifique usuel. Vous justifierez brièvement chaque réponse.

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

Q2 Quel type choisiriez-vous pour représenter les données suivantes ? Vous justifierez brièvement chaque réponse.

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 Fonctions en Python

Q3 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`.

Q4 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`).

4 Boucles IF, FOR, WHILE

Q5 Indenter de deux manières différentes la suite d'instructions suivante afin que la variable `t` contienne `True` pour une indentation, puis `False` pour l'autre.

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

Q6 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")
```

Q7 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
```

Q8 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 écrira une fonction `nb_exos()` ne prenant pas d'argument et renvoyant le résultat demandé.

5 Fonctions plus avancées

Q9 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.