Programmer en Python
SVE Licence 1, Mathématiques
TP3 - Boucles et instructions conditionnelles

1. Listes

Une liste est une structure de données qui contient une série de valeurs. Python autorise la construction de listes contenant des valeurs de types différents (par exemple entier et chaîne de caractères), ce qui leur confère une grande flexibilité.
Une liste est déclarée par une série de valeurs (ne pas oublier les guillemets, simples ou doubles, s'il s'agit de chaînes de caractères) séparées par des virgules, et le tout encadré par des crochets. En voici quelques exemples :

```
>>> animaux = ["girafe","tigre","singe","souris"]
```

Dans l'exemple ci-dessus, `animaux` est le nom qu'on donne à la liste. Et `"girafe"`, `"tigre"`, `"singe"`, `"souris"` sont les éléments contenus par la liste. On utilise des guillemets `"girafe"` pour indiquer à python que c'est la chaine de caractères `"girafe"` qu'on veut stocker dans la liste.

```
>>> tailles = [5, 2.5, 1.75, 0.15]
```

Dans l'exemple ci-dessus, `taille` est le nom qu'on donne à la liste. Et `5, 2.5, 1.75, 0.15` sont les éléments contenus par la liste. On n'utilise pas de guillemets car on stocke ici des nombres.

```
>>> mixte = ["girafe", 5, "souris", 0.15]
>>> print(mixte)
["girafe", 5, "souris", 0.15]
>>> print(tailles)
[5, 2.5, 1.75, 0.15]
```

Un des gros avantages d'une liste est que vous pouvez appeler ses éléments par leur position. Ce numéro est appelé indice (ou index) de la liste.
Attention : on compte à partir de 0.

```
>>> animaux = ["girafe","tigre","singe","souris"]
>>> print(animaux[0])
"girafe"
>>> print(animaux[1])
"tigre"
```

On peut aussi afficher plusieurs éléments de la liste.

```
>>> print(animaux[2:4])
["singe","souris"]
```

La fonction `len` permet d'afficher la longueur d'une liste.

```
>>> print(len(animaux))
```

2. Boucle for

Imaginez que vous ayez une liste de quatre éléments dont vous voulez afficher les éléments les uns après les autres. Dans l'état actuel de vos connaissances, il faudrait taper quelque chose du style :

```
>>> animaux = ["girafe","tigre","singe","souris"]
>>> print(animaux[0])
>>> print(animaux[1])
>>> print(animaux[2])
>>> print(animaux[3])
```

Si votre liste ne contient que quatre éléments, ceci est encore faisable, mais imaginez qu'elle en contienne 100 voire 1000 ! Pour remédier à cela, il faut utiliser les boucles. Regardez l'exemple suivant :

```
>>> animaux = ["girafe","tigre","singe","souris"]
>>> for i in range(0,len(animaux)):
>>>   print(animaux[i])
girafe
tigre
singe
souris
```

En fait, la variable « i » va prendre successivement les valeurs des différents indices (0, 1,... générées par la fonction range) de la liste animaux à chacune des itérations. D'ores et déjà, remarquez l'indentation ; vous voyez que l'instruction `print(animaux[i])` est décalée vers la droite par rapport à l'instruction `for i in range(0,len(animaux)):`.

On peut aussi écrire plus simplement,

```
>>> animaux = ["girafe","tigre","singe","souris"]
>>> for i in animaux:
>>>   print(i)
girafe
tigre
singe
souris
```

Dans ce cas, la variable « i » prend ses valeurs directement dans la liste animaux et on ne passe plus par les indices.

Un autre exemple

```
adn = "atcacgtta"
for i in range(0,len(adn)):
    print("la base no", i, "est", adn[i])
```

Ceci s'écrit aussi

```
adn = "atcacgtta"
i = 0
for a in adn:
    print("la base no", i, "est", a)
    i = i+1
```

ou encore

```
adn = "atcacgtta"
i = 0
for i,a in enumerate(adn):
        print("la base no", i, "est", a)
```

En pratique, vous choisirez l'écriture qui convient le mieux selon le problème de programmation posé.

3. Exercices dans le domaine de la biologie

1. Dans cet exercice, on cherche à compter les atomes de soufre d'une séquence protéique.
   (a) Dans la séquence protéique (utilisant le code international des acides aminés) `proteine = "CVAPGPMCAWCDSTAC"` ,
   compter le nombre de Cystéines (code C). On pourra utiliser une boucle `for`.
   Rappel : Une chaine de caractères est une liste.
   (b) Dans la même séquence protéique, compter le nombre d'atomes de soufre. Indication : les acides aminés soufrés sont la Cystéine (C) et la Méthionine (M)

2. En biologie de l'évolution, une orthologie est un lien évolutif entre deux gènes présents chez deux espèces différentes. Deux séquences génétiques homologues de deux espèces différentes sont orthologues si elles descendent d'une séquence unique présente dans le dernier ancêtre commun aux deux espèces. Dans cet exercice, on cherche à comparer la séquence de 3 protéines orthologues chez le chien, le chat et l'homme. On voudrait savoir, du point de vue évolutif, qui sont le plus proches parmi l'homme, le chat et le chien.

(a) Dans les deux séquences suivantes, comptez à l'aide d'une boucle le nombre de positions identiques entre les 2 séquences ?
```
chat = "MWDRTLIILLKVCWT"
chien = "MYERTIIILLKVTTL"
```

(b) En déduire quel est le pourcentage d'identité entre ces 2 séquences ?

(c) Dans les deux séquences suivantes, comptez à l'aide d'une boucle le nombre de positions identiques entre les 2 séquences ?
```
homme = "MWERTLIVVKATTL"
chat = "MWDRTLIILLKVCWT"
```

(d) En déduire quel est le pourcentage d'identité entre ces 2 séquences ?

(e) Selon cette séquence, le chat est-il plus proche du chien ou de l'homme ? Une des deux espèces est-elle plus proche de l'homme que l'autre ?

3. On cultive une population de bactéries. Tous les jours, la population de bactérie est multipliée par 1.6.

(a) Si la population de départ est de 10 bactéries, combien y en a t-il au bout de 3 jours ?

(b) Si la population de départ est de 10 bactéries, calculez à l'aide d'une boucle combien il y a de bactéries au bout de 3 jours ; puis de 51 jours.

(c) Tracez une courbe montrant la progression de la population en fonction des jours de culture (vous aurez besoin des éléments que vous avez appris au TP précédent !).
Indications : pour tracer une courbe, on doit garder en mémoire les résultats intermédiaires (par exemple dans une liste).
Pour ajouter un élément `a` à une liste existante `L`, on peut utiliser l'instruction `L.append(a)`

4. On définit deux suites $(a_n)_{n\in\mathbf{N}}$ et $(b_n)_{n\in\mathbf{N}}$, par leur premiers termes respectifs $a_0 = 1$ et $b_0 = 2$, et pour tout entier n, par les relations

$$a_{n+1} = \frac{a_n + b_n}{2} \text{ et } b_{n+1} = (a_n b_n)^{1/2}$$

Écrire un programme qui demande un entier n et affiche les valeurs a_n et b_n correspondantes.
Qu'observez vous si n est petit (par exemple égal à 2) ? Si n est grand (par exemple égal à 40) ?