Stage Février 2016 Du 08/02/16 au 12/02/16

Exercices sur les boucles avec Python (Pyzo)

Le but du TP est d'effectuer des premiers calculs avec Python ainsi que de réaliser des premières boucles et instructions conditionnelles. Vous utiliserez Python via Pyzo.

On commence par ouvrir Pyzo (dans 'Programme' puis 'Prepa' puis 'Pyzo') qui se présente sous forme de trois fenêtres :

- ☞ Un interpréteur : là où l'on fait les calculs et où on a les sorties de nos programmes
- ☞ Un éditeur : là où l'on écrit nos programmes
- ☞ Une fenêtre où sont indiquées les noms de variable utilisés ainsi que leurs types et valeur

On rappelle quelques opérations accessibles :

Opérations sur les entiers et les flottants (nombres à virgules)

- ☞ + et - : addition et soustraction
- ☞ * et / : multiplication et division
- ☞ ** : exponentiation (puissance)
- ☞ abs : valeur absolue

Opérations supplémentaires sur les entiers

- ☞ // : division entière
- ☞ % : reste dans la division euclidienne (ou modulo)

Opérations sur les chaines de caractères

- ☞ + : concaténation
- ☞ s[k] : donne le caractère de rang k (le k+1-ième) de la chaine de caractères s
- ☞ len(s) : donne la longueur de la chaine de caractères s
- ☞ s.upper() ou s.lower() : transforme la chaine de caractères s en ne transformant les minuscules en majuscules (ou l'inverse)
- ☞ chr(nb) : donne le caractère de code ASCII nb : chr(65) est le 'A', chr(90) est le 'Z', chr(97) est le 'a', chr(122) est le 'z'.
- ☞ ord() : réalise l'opération inverse.

Opérations sur les listes

- ☞ + : concaténation
- ☞ L[k] : donne le terme de rang k (le k+1-ième) de la liste L
- ☞ len(L) : donne la longueur de la liste L
- ☞ max(L), min(L), sum(L) : maximum, minimum, somme des termes de la liste L (si ce sont des nombres)
- ☞ L.sort() : tri la liste L dans l'ordre croissant

Opérations sur les booléens

- ☞ and, or, not
- ☞ in : si L est une liste de nombres flottants ou entiers et x un entier ou un flottant, la condition 'x in L' donne le booléen True si x est dans L, False sinon
- ☞ in : si s est une chaine de caratères et x un caractère, la condition 'x in s' donne le booléen True si x est dans s, False sinon

Exercice 1. 1. Ecrire une suite d'instructions permettant de calculer 40!. Idem avec $n!$ lorsque n est un entier donné dans le script ou demandé à l'utilisateur
On rappelle l'algorithme correspondant :

$p \leftarrow 1$
Pour k allant de 1 jusqu'à 40, Faire
| $p \leftarrow p \times k$
| Fin Faire
Retourner p

Exercice 2.
1. Ecrire une suite d'instructions permettant d'échanger les contenus des variables a et b
2. Ecrire une suite d'instructions permettant, étant données trois variables a, b et c, d'affecter le contenu de a dans b, celui de b dans c et celui de c dans a

Exercice 3. Dans un jeu de bataille navale, on doit rechercher un couple d'entiers (X, Y) choisi au hasard par l'ordinateur. On propose un couple (P, Q) d'entiers. L'ordinateur doit alors répondre si on a trouvé le bon nombre ou pas. En cas d'échec, l'ordinateur doit répondre si on est dans le bon alignement (indifféremment horizontal ou vertical), si on est près du point (à une distance inférieure à 2 par exemple), ou si notre tentative est totalement ratée...

Exercice 4. Un jeu de lettres télévisé consiste à deviner un mot inconnu de 7 lettres en donnant une (ou plusieurs) propositions d'un mot ayant également 7 lettres.

1. On suppose ici que l'on ait affecté deux mots de 6 lettres **motcache** et **motessai** (on pourra commencer, en guise d'essai, par affecter à **motcache** le mot 'algebre' et à **motessai** le mot 'analyse'). Ecrire une séquence d'instructions qui donne le message 'Trouvé' si les deux mots sont les mêmes et le message 'Non' sinon
2. Modifier les instructions pour que la réponse soit le nombre de lettres de **motessai** qui sont dans **motcache** et à la même place. Modifier une nouvelle fois la séquence pour avoir également le nombre de lettres **motessai** qui sont dans **motcache** mais à des places différentes.
3. Ecrire une séquence d'instructions demandant à un utilisateur d'entrer un mot de 7 lettres et affectant ce mot dans la variable de nom **motcache**.
4. Ecrire une séquence d'instructions demandant à un 'autre' utilisateur d'entrer un mot de 7 lettres et affectant ce mot dans la variable de nom **motessai** et qui donne les réponses prévues par les questions précédentes.

Exercice 5. Que fait le programme suivant :

```
a = int(input())
b = int(input())
c = int(input())
d = int(input())
if b == 0 or d == 0:
  print(''Denominateur nul interdit !'')
else :
  print( a * d + c * b)
  print( b * d)
```

Exercice 6.
1. Ecrire un programme qui demande à l'utilisateur d'entrer 3 nombres (rèels) a, b et c et qui affiche le nombre de solutions réelles de l'équation $ax^2 + bx + c = 0$ ainsi que ces racines réelles le cas échéant
2. Transformer ce programme pour qu'il donne également les racines les racines complexes non réelles
3. Tester ce programme avec $a = 1.0$, $b = 6.0$ et $c = 9.0$. Faites de même avec $a = 0.1$, $b = 0.6$ et $c = 0.9$. Que peut-on constater ? Corriger le programme pour éviter ce comportement.

Exercice 7. 1. Ecrire une suite d'instructions permettant, étant données deux variables a et b contenant des nombres, d'affecter à a la valeur maximale de ces deux nombres et à b la valeur minimale.

2. Ecrire une suite d'instructions permettant, étant données trois variables a, b et c contenant des nombres, d'afficher la plus grande de ces valeurs.

Exercice 8. L'algorithme ci-dessous permet de déterminer le coefficient directeur d'une droite passant par deux points d'abscisses différentes.

Variables : a, b, c, d, m réels
Entrées et initialisation
| Afficher "Saisir les coordonnées du point A"
| Lire a, b
| Afficher "Saisir les coordonnées du point B"
| Lire c, d
Traitement
| $m \leftarrow \dfrac{d-b}{c-a}$

Sorties : Afficher m

1. Compléter cet algorithme pour qu'il fournisse l'ordonnée à l'origine de cette droite
2. Ecrire le programme Python correspondant
3. L'algorithme proposé ne prend pas en compte le cas d'une droite parallèle à l'axe des ordonnées. Modifier l'algorithme pour que ce cas soit traité.

Exercice 9. Soit f la fonction définie par : $f(x) = \begin{cases} -x+2 & \texttt{Si } x < 1 \\ x^2 - 2x + 2 & \texttt{Sinon} \end{cases}$ Ecrire une suite d'instructions qui, pour une valeur de$x$ donnée renvoie la valeur de $f(x)$

Exercice 10. Etant donnée la suite $(u_n)_{n\in\mathbb{N}}$ définie par :
$u_0 = 1$ et la relation :$\forall n \in \mathbb{N}, u_{n+1} = 3u_n - 1$

1. Ecrire un programme permettant de calculer les valeurs de u_5, u_{10}, u_{20} de la suite.
2. Ecrire un programme permettant de calculer la somme de tous les termes de la suite pour des rangs compris entre 0 et 20

Exercice 11. Etant donnée la suite $(u_n)_{n\in\mathbb{N}}$ définie par :
$u_0 = 2$, $u_1 = 4$ et la relation :$\forall n \in \mathbb{N}, u_{n+2} = 4u_{n+1} - u_n$

1. Ecrire un programme permettant de calculer les valeurs de u_5, u_{10}, u_{20} de la suite.
2. Ecrire un programme permettant de calculer la somme de tous les termes de la suite pour des rangs compris entre 0 et 20

Exercice 12. Etant donnée la suite $(u_n)_{n\in\mathbb{N}}$ définie par :
$u_0 = 1$ et la relation :$\forall n \in \mathbb{N}, u_{n+1} = \dfrac{u_n}{1+u_n}$

1. Ecrire un programme permettant de calculer les valeurs de u_5, u_{10}, u_{20} de la suite.
2. Ecrire un programme permettant de calculer la somme de tous les termes de la suite pour des rangs compris entre 0 et 20
3. Ecrire un programme permettant de calculer la différence de deux termes successifs de la suite $u_{n+1} - u_n$ pour $n = 5$, $n = 10$ et $n = 20$. Que peut-on supposer ? Le montrer.

Exercice 13. Etant donnée la suite $(u_n)_{n\in\mathbb{N}}$ définie par :
$u_0 = 0$ et la relation :$\forall n \in \mathbb{N}, u_{n+1} = \dfrac{2u_n + 3}{4 + u_n}$

1. Ecrire un programme permettant de calculer les valeurs de u_5, u_{10}, u_{20} de la suite.
2. On note $v_n = \dfrac{u_n - 1}{u_n + 3}$. Calculer les 10 premières valeurs de v_n. Que peut-on supposer ? Le montrer.

Exercice 14. Etant donnée la suite $(u_n)_{n\in\mathbb{N}}$ définie par :
$u_0 = -1$, $u_1 = 1/2$ et la relation :$\forall n \in \mathbb{N}, u_{n+2} = u_{n+1} - (1/4) * u_n$
On pose $v_n = u_{n+1} - (1/2) * u_n$

1. Ecrire un programme permettant de calculer les 10 premières valeurs de v_n.
2. On pose $w_n = \dfrac{u_n}{v_n}$. Ecrire un programme permettant de calculer les 10 premières valeurs de w_n
3. Des deux questions précédentes, que peut-on supposer ? Le montrer.
4. Ecrire un programme permettant de calculer la somme de tous les termes de la suite pour des rangs compris entre 0 et 20

Exercice 15. Soit l'algorithme ci-dessous

Variables : n, i entiers u, S réels
Entrées et initialisation
| Lire n
| $u \leftarrow 1$, $S \leftarrow 1$ et $i \leftarrow 0$
Traitement
| **tant que** $i < n$ **faire**
| | $u \leftarrow 2u + 1 - i$
| | $S \leftarrow S + u$
| | $i \leftarrow i + 1$
| **Fin faire**
Sorties : Afficher u, S

1. Déterminer les valeurs de u et S pour n variant de 0 à 5
2. Soit $(u_n)_{n\in\mathbb{N}}$ la suite définie par :
   $u_0 = 1$ et la relation :$\forall n \in \mathbb{N}, u_{n+1} = 2u_n + 1 - n$
   Calculer les 5 premiers termes de la suite $(u_n - n)_{n\in\mathbb{N}}$. Que peut-on conjecturer ? Le montrer
3. Que peut bien représenter la variable S , Que vaut-elle ?

Exercice 16. Etant donnée la suite $(u_n)_{n\in\mathbb{N}}$ définie par :
$u_0 = 1$ et la relation :$\forall n \in \mathbb{N}, nu_n = (n + 1)u_{n-1} + 1$

1. Ecrire un programme permettant de calculer les 10 premières valeurs de u_n.
2. Que peut-on supposer ? Le montrer.

Exercice 17. Le lièvre et la tortue
Il s'agit d'un jeu qui se joue avec un dé sur un plateau de sept cases.
Au début du jeu le lièvre et la tortue sont sur la case départ. On lance le dé. Si on obtient 6, le lièvre arrive directement à la case arrivée. Sinon c'est la tortue qui avance du nombre désigné par le dé.
On relance alors le dé avec les mêmes règles : si c'est un 6 qui sort, le lièvre arrive directement à la case arrivée. Sinon c'est la tortue qui avance du nombre désigné par le dé.
L'algorithme est le suivant :

Variables : T, L, D entiers G chaîne
Entrées et initialisation
| $T \leftarrow 0$, $L \leftarrow 0$
Traitement
| **tant que** $T < 7$ et $L < 7$ **faire**
| | D prend la valeur d'un jet de dé
| | **si** $D = 6$ **alors**
| | | $L \leftarrow 7$, $G \leftarrow$ 'Lièvre'
| | **sinon**
| | | $T \leftarrow T + D$
| | **fin Si**
| | **si** $T \geqslant 7$ **alors**
| | | $G \leftarrow$ 'Tortue'
| | **fin Si**
| **fin Si**
Sorties : Afficher "Le gagnant est : " G

1. On simule le lancer de dé par l'appel de la fonction random.randint(1,6) après avoir importé le module random. Écrire un programme qui réalise une partie fictive.
2. Modifier l'algorithme pour réaliser une simulation de 1000 parties
3. L'un des deux protagonistes est-il avantagé par les règles ? Si oui, modifier le nombre de cases du plateau pour rendre le jeu le plus équitable possible