

EXAMEN DE FIN DU SEMESTRE 1

Matière : INFORMATIQUE

Classes : 1^{ère} Année MP – T – PC Durée : 2 h



N.B. :

- Toutes les réponses doivent être rédigées en Python. Il faut numéroté les questions.
- L'indentation dans les instructions Python sera prise en considération dans le barème.
- Le sujet comporte 4 pages.
- Vous trouvez, en Annexe, quelques méthodes prédéfinies sur les conteneurs.

Exercice 1 (10 points)

Un polynôme **P** de *degré n* est de la forme : $P(X) = a_0 + a_1X + a_2X^2 + a_3X^3 + \dots + a_nX^n$ où a_n est **non nul**. Les réels $a_0, a_1, \dots, a_n$ sont les *coefficients* du polynôme. Le terme de *degré k* est a_kX^k . L'exposant du terme de degré le plus élevé donne le *degré* du polynôme.

Exemples : Le polynôme $2X + 3X^2 + 7X^5$ est de *degré 5*. Le polynôme $3 + 4X$ est de *degré 1*. Le polynôme constant 3 est de *degré 0*.

Nous avons choisi de représenter un polynôme par une liste Python contenant ses coefficients $[a_0, a_1, \dots, a_n]$; et tels que la valeur à l'indice *i* dans la liste correspond au coefficient de X^i dans le polynôme. Par exemple, la liste $[7.0, 2.3, -9.12, 0.0, 7.8]$ représente le polynôme $7.0 + 2.3X - 9.12X^2 + 7.8X^4$. Par ailleurs, une liste représentant un polynôme doit contenir au moins une valeur, éventuellement nulle (c'est-à-dire $[0]$). On considère aussi que le degré du polynôme nul est 0.

Travail demandé :

Écrire un script Python permettant de :

1. Saisir, puis, afficher la liste **L** non vide et contenant les coefficients **réels** d'un polynôme **P**. Les coefficients sont saisis un par un et on demande à l'utilisateur à chaque fois s'il souhaite continuer (l'utilisateur tape le caractère 'o' pour continuer et 'n' pour terminer la saisie).

Voici un exemple d'exécution :

```
Entrez un coefficient : 7
** Souhaitez-vous continuer ? k
** Souhaitez-vous continuer ? o
Entrez un coefficient : 2
** Souhaitez-vous continuer ? o
Entrez un coefficient : 3
** Souhaitez-vous continuer ? n
La liste des coefficients de P est [7.0, 2.0, 3.0]
```

2. Saisir le **réel** x , puis, afficher, la valeur de l'évaluation numérique de P en x , c'est-à-dire $P(x)$. Par exemple, si $x=2$ et $L=[7.0, 2.0, 3.0]$, alors le résultat affiché est 23.0 (étant égal à $7.0 + 2.0 \times 2.0 + 3.0 \times 2.0^2$).

Voici un exemple d'exécution :

```
>>> Entrez un nombre réel, x= 2
      Pour x = 2.0, P = 23.0
```

3. Afficher le degré du polynôme P à partir de la liste de ses coefficients L , c'est-à-dire l'**exposant** de X dans le terme de plus haut degré ayant un **coefficient non nul**. Par exemple, si $L=[7.0, 2.0, 3.0, 0.0]$, alors, le résultat affiché est "Le degré de $P = 2$ "

4. Construire, puis, afficher les deux tuples **TP** et **TIMP** contenant respectivement les coefficients de **degrés pairs** et **impairs** du polynôme P .

Par exemple, si $L=[4.0, 0.0, 12.0, -2.0, 0.0, 4.0, 9.3]$, alors, on affiche "TP=(4.0, 12.0, 0.0, 9.3) et TIMP=(0.0, -2.0, 4.0)."

5. Construire, puis, afficher la chaîne de caractères **Expr** correspondant à l'expression du polynôme P à partir de sa liste de coefficients L .

Par exemple, si $L=[4.0, 0.0, 12.0, -2.0, 0.0, 4.0, 9.3]$, alors, le résultat affiché est " $P(X) = 4.0 + 12.0X^2 - 2.0X^3 + 4.0X^5 + 9.3X^6$ ".

Remarque : Les termes à coefficients **nuls** ne doivent pas apparaître dans la chaîne (comme $0.0X^1$ dans notre exemple) et le terme de degré 0 est affiché sans exposant.

Soient, maintenant, les deux listes L_p et L_q étant déjà créées et correspondants respectivement aux coefficients des deux polynômes P et Q .

On vous demande d'étendre votre script Python avec les instructions permettant de :

6. Créer, puis, afficher la liste L_s correspondant à la liste des coefficients du polynôme somme $S = P + Q$.

7. Modifier, puis, afficher la liste L_s , en supprimant tous les coefficients **nuls** de degrés supérieurs au degré du polynôme somme, s'ils existent, c'est-à-dire, tous les 0 qui se trouvent à la fin de la liste L_s .

Par exemple, si $L_s=[4.0, 0.0, 12.0, -2.0, 0.0, 4.0, 9.3, 0.0, 0.0, 0.0]$, alors, le résultat affiché est " $L_s=[4.0, 0.0, 12.0, -2.0, 0.0, 4.0, 9.3]$ ".

Attention ! Si tous les coefficients de S sont nuls, alors S est le polynôme nul et la liste modifiée devra contenir un seul 0.

8. Construire, puis, afficher le tuple **TH** correspondant au **terme** du plus haut degré du polynôme somme. Par exemple, si $L_s=[7.0, 2.0, 3.0]$, alors, le résultat affiché est " $TH=(3.0, 2)$ ".

Exercice 2 (7 points)

Le Scrabble est un jeu de lettres dont l'objectif est de cumuler des points, sur la base de tirages aléatoires de lettres, en créant des mots sur une grille carrée appelée aussi plateau de jeu. Le jeu consiste à former plusieurs fois des mots qui rapportent le plus de points. La lettre E est la plus fréquente (15 occurrences) et ne vaut qu'un point, tandis que les « lettres chères » J, K, Q, W, X, Y et Z sont uniques mais valent 8 (J et Q) ou 10 points (K, W, X, Y et Z). Les lettres B, C et P valent 3 points, F, H et V valent 4 points, D, G et M valent 2 points.

En vue de simuler ce jeu d'une manière simplifiée, on se propose d'écrire un programme Python qui ne traite qu'un **seul mot proposé** par un **seul joueur**. Pour cela, on suppose qu'on a le dictionnaire Python **D** dont les clés sont les lettres alphabétiques en majuscule et leurs valeurs correspondantes sont des tuples formés chacun par deux entiers : le premier entier est le nombre d'occurrences de la lettre dans le jeu et le second est le nombre de points qui lui est attribué. Le dictionnaire **D** est représenté ci-après :

```
D = {'A':(9,1), 'B':(2,3), 'C':(2,3), 'D':(3,2), 'E':(15,1), 'F':(2,4), 'G':(2,2),  
'H':(2,4), 'I':(8, 1), 'J':(1, 8), 'K':(1,10), 'L':(5,1), 'M':(3,2), 'N':(6,1),  
'O':(6,1), 'P':(2,3), 'Q':(1,8), 'R':(6,1), 'S':(6,1), 'T':(6,1), 'U':(6,1),  
'V':(2,4), 'W':(1,10), 'X':(1,10), 'Y':(1,10), 'Z': (1,10)}
```

On suppose aussi qu'on a la liste python **T** contenant les lettres placées sur le plateau du jeu et que le joueur ne doit utiliser qu'une seule lettre de cette liste pour former un mot (pour simplifier).

Travail demandé :

On rappelle que le dictionnaire **D** et la liste **T** sont déjà créés, écrire un script Python permettant de :

1. Construire, à partir du dictionnaire **D**, la liste **L** formée par **toutes** les lettres disponibles dans notre jeu, c'est-à-dire, d'abord, neuf fois la lettre 'A', ensuite, deux fois la lettre 'B', puis, deux fois la lettre 'C' et ainsi de suite ...
2. Construire, puis, afficher la liste **jeu** contenant 7 lettres tirées au hasard à partir de la liste **L** créée précédemment. À chaque fois, l'ordinateur choisit au hasard un nombre entier **i** compris entre 0 et la (longueur de la liste **L**) -1. L'entier **i** représente l'indice de la lettre dans la liste **L** qui doit être, à la fois, retirée de cette liste et ajoutée à la liste **jeu**.

Indication : Utiliser la fonction `randint(a,b)`, du module `random`, qui retourne un nombre entier au hasard dans l'intervalle `[a,b]` (avec `a` et `b` inclus).

3. Demander la saisie de la lettre en majuscule **pl** qui représente la lettre du plateau utilisée et se trouvant dans la liste **T**. Si **pl** existe dans **T**, alors, elle va être transférée à la liste **jeu**, sinon, on doit redemander la saisie de la lettre **pl** (voir exemple d'exécution à la fin de l'exercice).
4. Demander la saisie d'une chaîne de caractères **mot** formée par des lettres alphabétiques en majuscule uniquement et qui représente la proposition qu'on va traiter (voir exemple d'exécution à la fin de l'exercice).
5. On considère maintenant que le mot est valide s'il est formé par des lettres qui figurent dans la liste **jeu**. **Attention** : Une fois utilisée, la lettre de la liste **jeu** ne peut pas être réutilisée de nouveau. Créer, alors, la variable booléenne **valide** qui reçoit la valeur `True`, si le mot est valide, `False`, sinon.
6. On vous demande d'ajouter à votre script Python les instructions permettant de :
 - a. Afficher "Votre mot est accepté", puis, calculer et afficher le nombre **N** en cumulant les points attribués à chacune des lettres du **mot** qui sont indiqués dans le dictionnaire **D**, si le mot est **valide** (voir exemple d'exécution à la fin de l'exercice).
 - b. Sinon, afficher "Votre mot est rejeté, essayez de nouveau SVP !"

Exemple d'exécution :

```
Jeu=['A', 'R', 'C', 'A', 'B', 'E', 'H']      # 7 lettres tirées au hasard
T=['P', 'Y', 'T', 'H', 'O', 'N', 'I', 'R']  # les lettres du plateau
Entrer la lettre du plateau utilisée : r     # choix erroné
Entrer la lettre du plateau utilisée : B     # choix erroné
Entrer la lettre du plateau utilisée : R     # choix valide
Jeu=['A', 'R', 'C', 'A', 'B', 'E', 'H' 'R']
Entrez votre proposition : arb re           # saisie invalide
Saisie non valide, entrez votre proposition : ARBRE
Votre mot est accepté .
Bravo ! Vous avez cumulé 7 points.
```

Exercice 3 (3 points)

Donner la complexité au pire des cas (en O) des exemples suivants :

En algorithmique	En langage Python
<u>{Exemple 1}</u> lire(n) tantque n > 0 faire n ← n DIV 2 fintantque pour i de 1 à n faire {instructions en O(1)} finPour	<u># Exemple 1</u> n = int(input()) while n>0 : n = n // 2 for i in range(1,n+1) : # instructions en O(1)
<u>{Exemple 2}</u> lire(n) pour i de 1 à n-3 faire pour j de i+2 à n-1 faire k ← 1 tantque k<n faire {instructions en O(1)} k ← 2*j fintantque finPour finPour	<u># Exemple 2</u> n = int(input()) for i in range(1,n-2) : for j in range(i+2,n) : k=1 while k<n : # instructions en O(1) k=2*j
<u>{Exemple 3}</u> x ← 0 lire(n) i ← n tantque i > 1 faire pour j de 1 à n faire x ← x + 1 finPour i ← i DIV 2 fintantque	<u># Exemple 3</u> x=0 n = int(input()) i=n while i>1 : for j in range (1,n+1) : x=x+1 i=i//2