Informatique - TP 4

Révisions

I. Suites récurrentes

Ex. 4.1 Soit v la suite définie par $v_0 = 4$ et $\forall n \in \mathbb{N}$, $v_{n+1} = v_n + (-1)^{n+1} \times \dfrac{4}{2n+3}$.

Écrire une fonction `alterne(n)` qui renvoie la valeur de v_n.
Vers quelle limite semble converger la suite v ?

Ex. 4.2 ***Tchebychev***

a. Pour $x \in \mathbb{R}$, la suite de Tchebychev est définie par $P_0 = 1$, $P_1 = x$ et $P_{n+2} = 2xP_{n+1} - P_n$.
Écrire une fonction `Tchebychev(n,x)` qui renvoie, pour x donné, la valeur de P_n.

b. La plupart des fonctions mathématiques ne sont pas directement accessibles en langage Python.
Pour les obtenir, il faut ***importer un module*** mathématique qui fournit ces fonctions.
Importer le module `numpy` grâce à la ligne :
`import numpy as np`
Ensuite `np.cos`, par exemple, vous donnera la fonction cosinus, etc...
Comparer `Tchebychev(n,np.cos(x))` et $\cos(nx)$.

II. Chaînes de caractères

Ex. 4.3 Écrire une fonction `listeCarac(chaine)` qui renvoie la liste des caractères présents dans la chaine de caractères `chaine`.
Par exemple : `listeCarac("bonbon")`
doit renvoyer
`["b", "o", "n"]`

Ex. 4.4 Écrire une fonction `listeNombreCarac(chaine)` qui renvoie la liste des couples `(c,n)` où `c` décrit l'ensemble des caractères présent dans `chaine` et où `n` est le nombre d'occurrences de `c` dans `chaine`.
Par exemple : `listeNombreCarac("bonbon")`
doit renvoyer
`[("b",2), ("o",2), ("n",2)]`
et `listeNombreCarac("oppose")`
doit renvoyer
`[("o",2), ("p",2), ("s",1), ("e",1)]`

Ex. 4.5 Écrire une fonction `memesLettres(mot1,mot2)` qui renvoie `True` si et seulement si `mot1` et `mot2` sont formés avec les mêmes lettres (mais pas forcément en même nombre).
Par exemple : `memesLettres("bonbon", "bon")`
doit renvoyer
`True`
et `memesLettres("oppose", "appose")`
doit renvoyer
`False`

Ex. 4.6 Écrire une fonction `anagrammes(mot1,mot2)` qui renvoie `True` si et seulement si `mot1` et `mot2` sont formés avec les mêmes lettres, en même nombre.
Par exemple : `anagrammes("bonbon", "bon")`
doit renvoyer
`False`
et `anagrammes("chien", "chine")`
doit renvoyer
`True`

Ex. 4.7 Dans cet exercice, on représente un polynôme par la liste de ses coefficients, donnés par ordre croissant des degrés. Par exemple, le polynôme $P = X^3 + X + 1$ est représenté par la liste `[1,1,0,1]`.

a. Écrire une fonction `poly(n)` renvoyant la liste de ***tous les polynômes*** de degré inférieur ou égal à n dont les coefficients valent 0 ou 1. Un polynôme étant lui-même représenté par une liste, cette fonction doit donc renvoyer une liste de listes.
Par exemple, `poly(2)` doit renvoyer la liste
`[[0],[1],[0,1],[1,1],[0,0,1],[1,0,1],[0,1,1],[1,1,1]]`

b. Écrire une fonction `evalue(P,z)` prenant en paramètre un polynôme P et un complexe z et renvoyant la valeur de $P(z)$.

c. Pour effectuer des représentations graphiques (diverses et variées), on importe le module `matplotlib` :
```
import matplotlib.pyplot as plt
```
Une fois le module importé, la fonction `plot` permet d'effectuer des représentations graphiques. Nous aurons besoin dans cette question de représenter graphiquement des points du plan, ce qui se fait de la manière suivante :
```
plt.plot(abscisse,ordonnee,'.r')
```
où le point apparaîtra sous la forme d'un • rouge.
Écrire un code qui, étant donné un entier `n` et un complexe `z`,
- calcule la liste des polynômes `poly(n)`,
- pour chaque polynôme `P`, évalue sa valeur `P(z)` en `z`
- affiche le point d'affixe `P(z)`.

Tester votre code pour $n \in [\![10; 12]\!]$ et $z = \dfrac{1+i}{2}$ par exemple - on pourra aussi tester pour $z = \dfrac{1+i\sqrt{7}}{4}$, ou $z = ...$