I11: Contrôle terminal - session 1
Licence 1 MATHS, PC, SI

Janvier 2018 (semestre 1) - Durée : 2h00

- Tous les documents, calculatrices et appareils de communication sont interdits.
- Le barême est donné à titre indicatif
- Tous les scripts devront être clairement indentés et les noms de variables choisis de façon appropriée.
- Seules les instructions et fonctions internes à Python **vues en cours** sont autorisées.

EXERCICE 1. (2 points)

On considère les déclarations de variables suivantes:

```
nb = "12345"
L=[nb, 3.14, ["je","tu","il","nous","vous","ils"], (-1,2)]
```

Donner le **type** et la **valeur** des expressions suivantes:

`L[0]`, `L[1]+str(L[0][1])`, `(2,-1)+L[3]`,
`L[2][1:4]`, `L[-2][1::2]`, `L[0][:2]*L[-1][-1]`,

EXERCICE 2. (1 points) Écrire un script qui demande deux nombres entiers à l'utilisateur et affiche toutes les valeurs de la suite $U_0 = 1, U_{n+1} = \frac{U_n^2}{2} - 2U_n + 1$ pour n compris entre ces deux valeurs (incluses).

Exemple:

```
>>>
 Saisir un entier: 1
 Saisir un entier: 3
-0.5
2.125
-0.9921875
```

Exercice 3. (2 points)

On considère le script suivant :

```
n = int(input())
crible = [1]*n
i=2
while i<= n**0.5:
  if crible[i]==1:
    j=i*i
    while j<n:
      crible[j]=0
      j=j+i
  i=i+1
for i in range(2,len(crible)):
  if crible[i]==1:
    print(i)
```

Faire une table des valeurs de ce script pour l'entrée `n=16` sur le modèle suivant:

i	j	crible[i]==1	j<=n	ECRAN

Exercice 4. (4 points)

1. Écrire un script qui calcule la moyenne des nombres strictement positifs d'une part et la moyenne des nombres strictement négatifs d'autre part d'une série de nombres saisie par l'utilisateur; la saisie s'arrête quand le nombre 0 est rentré.

   Exemple:

   ```
   >>>
    Saisir un nombre: 2
    Saisir un nombre: 10
    Saisir un nombre: -3
    Saisir un nombre: 9
    Saisir un nombre: -6
    Saisir un nombre: 0
    Moyenne positifs: 7.0
    Moyenne negatifs: -4.5
   ```

2. Écrire un script qui demande 2 chaînes de caractères `ch1` et `ch2` à l'utilisateur, la première de longueur quelconque et la deuxième de longueur 2 (inutile de faire la vérification dans le script), et affiche le nombre d'occurences de `ch2` dans `ch1`. Par exemple

   ```
   >>>
     ch1 = ceci est certainement un exemple recent
     ch2 = ce
     nombres d'occurences: 3

   >>>
     ch1 = aaa
     ch2 = aa
     nombres d'occurences: 2
   ```

3. On considère la liste prédéfinie `L=[1,2,3,4,5,6,7,8,9,10]`. Écrire un script qui demande un nombre n à l'utilisateur et effectue une rotation de n cases sur la droite des éléments de la liste. Par exemple, pour $n = 2$ la liste deviendra `[9,10,1,2,3,4,5,6,7,8]`.

4. On définit une matrice comme étant une liste de listes de nombre. Par exemple:

$$M = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = \texttt{[[1,2,3],[4,5,6],[7,8,9]]}.$$

   On dit qu'une matrice est creuse si strictement plus de la moitié de ses coefficients sont nuls. Écrire une fonction `est_creuse(A)` qui retourne `True` si la matrice `A` est creuse et `False` sinon. Par exemple:

```
>>> est_creuse([0,0,1],[1,0,0],[1,1,0])
 True
>>> est_creuse([0,1],[1,0])
 False
```

Exercice 5. (5 points)

On considère que le script et les fonctions suivants sont écrits dans le même fichier.

1. La distance entre deux mots est le nombre de lettres en lesquelles ils diffèrent. Par exemple la distance entre `caste` et `vaste` vaut 1, celle entre `part` et `partir` vaut 2 et celle entre `crypte` et `egyptien` vaut 5. Écrire un script qui retourne la distance entre deux mots saisis par l'utilisateur. Écrire une fonction `distance(ch1, ch2)` qui retourne la distance entre les deux chaînes de caractères `ch1` et `ch2`.

2. Écrire une fonction `decoupe(ch)` retourne une liste contenant tous les mots de la chaine `ch`. On suppose que `ch` est une suite de mots séparés uniquement par un ou plusieurs espaces.

```
>>> decoupe("ceci est un exemple     avec des espaces")
 ["ceci","est","est","un","exemple","avec","des","espaces"]
```

3. On considère une chaîne de caractères prédéfinie `vocabulaire` ayant le même format que la chaîne `ch` précédente. Écrire un script qui affiche les deux mots les plus proches de cette chaîne au sens de la distance de la question 1. Par exemple pour la chaîne `vocabulaire = "mais ou et donc or ni car"` le script devra afficher `ou` et `or`.

EXERCICE 6. (5 points)

On considère que les fonctions suivantes sont écrites dans le même fichier. Une équipe de football sera représentée en Python par un tuple contenant son nom, une liste de caractères représentant ses résultats (”D” pour défaite, ”N” pour nul et ”V” pour victoire) et son nombre de buts marqués. Par exemple (après 5 matchs) `("Toulon", ["V","D","N","N","V"], 12)`. Un championnat sera représenté par une liste d'équipes.

1. Écrire une fonction `nb_points(e)` qui retourne le nombre de points correspondant au résultat de l'équipe `e`. On considèrera qu'une victoire rapporte 3 points, un nul 1 point et une défaite 0 point.

2. Écrire une fonction `compare(e1,e2)` qui retourne 1 si l'équipe `e1` est devant `e2` au classement, 0 si elles sont à égalité et -1 sinon. Pour comparer deux équipes on commencera par regarder laquelle possède le plus de point puis, en cas d'égalité celle, qui a marquée le plus de buts.

3. Écrire une fonction `champion(c)` qui retourne le nom de la ou des équipes du championnat `c` première(s) au classement.