Mathématiques supérieures — le 6 Novembre 2025

Corrigé du TP n°1 - Recherches séquentielles

Dans tout ce qui suit, l'appellation "tableau" pour une variable Python désignera indéferemment une liste, un tuple ou une chaîne de caractères.

Recherche séquentielle d'un élément

Q1 : Le principe d'une recherche séquentielle d'un élément `x` donné dans un tableau est la suivante :

> On parcourt les éléments du tableau un à un en testant à chaque fois l'égalité avec `x` et en renvoyant "Vrai" si c'est le cas. Si à la fin du parcourt, aucun cas d'égalité n'est avéré, on renvoie "Faux".

En se basant sur le principe précédent, écrire une fonction `recherche(x,L)` qui renvoie `True` si l'objet `x` se trouve dans le tableau `L` et `False` sinon.

Exemples de comportement attendu :

```
>>>recherche(1,[3,9,152,1,4])
True
>>>recherche(6,(3,3,7))
False
>>>recherche('a','blabla')
True
```

Q2 : **a)** Écrire une fonction `recherche_indice(x,L)` qui renvoie l'indice de la première occurrence de `x` dans `L` si l'objet `x` se trouve dans le tableau `L` et `None` sinon.

"première occurrence" signifie celui qui a le plus petit indice

b) Écrire une fonction `recherche_indice2(x,L)` qui renvoie cette fois-ci l'indice de la dernière occurence de `x` (et toujours `None` si `x` n'est pas dans `L`).

Exemples de comportement attendu :

```
>>>recherche_indice(1,[3,9,152,1,4])
3
>>>recherche_indice(3,(3,3,7))
0
>>>recherche_indice2(3,(3,3,7))
1
>>>recherche_indice('z','blabla')
```

Q3 : Écrire une fonction `recherche_indices(x,L)` qui renvoie la liste des indices de **toutes** les occurrences de `x` dans `L` si l'objet `x` se trouve dans le tableau `L` et la liste vide `[]` sinon.

Exemples de comportement attendu :

```
>>>recherche_indices(1,[3,1,152,1,4,1])
[1,3,5]
>>>recherche_indice(9,(3,3,7))
[]
```

Q4 : Pour cette dernière question d'application, on importera le module random et on créera la liste suivante :

```
1 from random import *
2
3 M=[chr(i) for i in range(97,123)]+[' ']
```

a) Que contient la liste `M` ?

b) Écrire une fonction `frazoazar(n)` qui renvoie une chaîne de caractères de longueurs `n` dont les lettres sont tirées aléatoirement et uniformément dans la liste `M`.

c) En se servant fonction `recherche_indices`, compter le nombre de `'a'` et le nombre de `'z'` dans une évaluation de `frazoazar(1000)`. Que remarque-t-on ? Expliquer pourquoi.

d) Écrire une fonction `Ayestu(n,nb)` qui, en utilisant la fonction `recherche`, recherche le caractère `'a'` dans `nb` tirages de phrases aléatoires de longueur `n` et renvoie la proportion des phrases contenant `'a'` parmi toutes les phrases crées.

e) À partir de combien de lettres dans les phrases aléatoires la proportion précédente semble-elle supérieure à 95% ?
Retrouver ce résultat mathématiquement.

Correction.

Q1 :

```
1 def recherche(x,L):
2     for i in range(len(L)):
3         if L[i] == x:
4             return True
5     return False
6
7 # ou avec la syntaxe Python du parcourt des valeurs d'une liste :
8
9 def recherche(x,L):
10    for e in L:
11        if e == x:
12            return True
13    return False
```

Q2 : **a)**

```python
def recherche_indice(x,L):
    for i in range(len(L)):
        if L[i] == x:
            return i
    return False
```

b)

```python
def recherche_indice2(x,L):
    n = len(L)
    for i in range(1,n+1):
        if L[n-i] == x: # ou aussi L[-i] en syntaxe Python
            return n-i
    return False
```

Q3 :

```python
def recherche_indices(x,L):
    Ind = []
    for i in range(len(L)):
        if L[i] == x:
            Ind.append(i)
    return Ind
```

Q4 : **a)**

```
>>> M
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n'
    , 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', ' '
    ]
```

b)

```python
def frazoar(n):
    chaine = ''
    for k in range(n):
        chaine += M[randint(0,len(L)-1)]
    return chaine
```