Devoir surveillé n°5 – NSI
12/12/2024

Exercice 1 (3 points - Question de cours)
Soit `l` une liste (de type `list`) en Python.
Rappeler et nommer les trois manières de parcourir la liste `l` données en cours.

Exercice 2 (4 points - Fonctions vues en cours)
Écrire le code Python des fonctions décrites ci-après :

1. Écrire une fonction `moyenne_simple` qui prend en argument une liste de nombres et qui retourne la moyenne de cette liste.

```
>>> moyenne_simple([3,5,7,5,9])
5.8
```

```
>>> (3+5+7+5+9)/5
5.8
```

2. Écrire une fonction `liste_indices(liste,x)` qui prend en argument une liste de nombres et un nombre `x` et retourne la liste des indices de tous les éléments égaux à `x`, éventuellement vide si il n'y a aucune occurrence de `x` dans `liste`.

```
>>> liste_indices([3,5,9,5,9,1],5)
[1, 3]
```

```
>>> liste_indices([3,5,9,5,9,1],7)
[]
```

Exercice 3 (3 points)
On suppose l'existence d'une fonction `contient(l,v)` qui prend en argument une liste `l` et une valeur `v`, et qui retourne `True` si `l` contient la valeur `v`, et qui retourne `False` sinon :

```
>>> contient([1,2,3,4],3)
True
```

```
>>> contient([1,2,3,4],5)
False
```

Recopier et compléter le code suivant de la fonction `sommeCondition(l1,l2)` qui prend en argument deux variables `l1` et `l2` de type `list` et qui retourne la somme des valeurs de `l1` qui sont contenues dans `l2`.

```
def sommeCondition(l1,l2):
  somme = ........
  for i in range(len(l1)):
    if contient(........, ........):
      ........
  return ........
```

```
>>> sommeCondition([4,5,2,1,7],[5,1,8])
6 # car, de la première liste, seuls 5 et 1 sont dans la seconde
```

Devoir surveillé n°5 – NSI
12/12/2024

Exercice 1 (3 points - Question de cours)
Soit `l` une liste (de type `list`) en Python.
Rappeler et nommer les trois manières de parcourir la liste `l` données en cours.

Exercice 2 (4 points - Fonctions vues en cours)
Écrire le code Python des fonctions décrites ci-après :

1. Écrire une fonction `occurrences(liste,x)` qui prend pour argument une liste et une valeur `x` et qui retourne le nombre d'occurrences (autrement dit d'apparitions) de `x` dans la liste.

```
>>> occurrences([3,5,7,5,9],5)
2
```

```
>>> occurrences([3,5,7,5,9],2)
0
```

2. Écrire une fonction `est_croissante` qui prend en argument une liste de nombres quelconques et qui retourne `True` si les valeurs de la liste sont rangées par ordre croissant, `False` sinon.
On considérera qu'une liste ayant au plus un élément est croissante.

```
>>> est_croissante([1, 2, 3])
True
```

```
>>> est_croissante([1, 3, 2])
False
```

Exercice 3 (3 points)
On suppose l'existence d'une fonction `contient(l,v)` qui prend en argument une liste `l` et une valeur `v`, et qui retourne `True` si `l` contient la valeur `v`, et qui retourne `False` sinon :

```
>>> contient([1,2,3,4],3)
True
```

```
>>> contient([1,2,3,4],5)
False
```

Recopier et compléter le code suivant de la fonction `copieSansDoublon(l)` qui prend en argument une variable `l` de type `list` et qui retourne une nouvelle liste contenant les éléments de `l` mais sans doublon.

```
def copieSansDoublon(l):
  nouvelle_liste = ........
  for i in range(len(l)):
    if not(contient(........, ........)):
      ........
  return ........
```

```
>>> copieSansDoublon([2,4,5,2,1,4,7])
[2, 4, 5, 1, 7]
```