TD4 : Fonctions récursives (2)

Compétences

- Lire et prévoir le résultat d'une fonction récursive
- Évaluer la terminaison d'une fonction récursive

Exercices

** **Ex. 1 —** On considère la fonction `checkNumber` suivante :

```
1  def checkNumbers(L, n, i, j) :
2      if i >= n - 1:
3          return True
4      else :
5          if j == n:
6              return checkNumbers(L, n, i+1, i+2)
7          else :
8              if L[i] == L[j]:
9                  return False
10             else:
11                 return checkNumbers(L, n, i, j+1)
```

1.Décrire précisément la pile d'exécution pour l'appel de fonction `checkNumbers([2,3,6,2], 0, 1)`
2.Décrire précisément la pile d'exécution pour l'appel de fonction `checkNumbers([2,3,6,1], 0, 1)`
3.Que fait cette fonction?
4.Quelle est sa complexité?
5.Donner la version itérative de `checkNumbers`
6.Quelle est sa complexité

** **Ex. 2 —** On considère la fonction `checkNumberTwo` suivante :

```
1  def checkNumbersTwo( L1, n1, L2, n2) :
2      if n1 == 0:
3          return L2
4      else :
5          elt = L1[n1-1]
6          if elt not in L2:
7              L2[n2 - n1] = elt
8      return checkNumbersTwo(L1, n1-1, L2, n2)
9
10 L1 = [3,5,3,4,2,1,1,3,9]
11 L2 = [None]*len(L1)
12 checkNumbersTwo(L1, len(L1), L2, len(L2)))
```

1.Décrire précisément la pile d'exécution lors de l'appel de `checkNumbersTwo`
2.Quel est le rôle de cette fonction?
3.Modifier la fonction `checkNumbersTwo` pour que le résultat soit `[9,3,1,2,4,5,None,None,None]`
4.Écrire une version permettant de conserver l'ordre initial des éléments.
5.Quelle est la complexité de cet algorithme?