BACCALAUREAT

SESSION 2022

Épreuve de l'enseignement de spécialité

NUMERIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°33

DUREE DE L'EPREUVE : 1 heure

Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Le candidat doit traiter les 2 exercices.

EXERCICE 1 (4 points)

On modélise la représentation binaire d'un entier non signé par un tableau d'entiers dont les éléments sont 0 ou 1. Par exemple, le tableau `[1, 0, 1, 0, 0, 1, 1]` représente l'écriture binaire de l'entier dont l'écriture décimale est

`2**6 + 2**4 + 2**1 + 2**0 = 83`.

À l'aide d'un parcours séquentiel, écrire la fonction `convertir` répondant aux spécifications suivantes :

```
def convertir(T):
    """
    T est un tableau d'entiers, dont les éléments sont 0 ou 1 et
    représentant un entier écrit en binaire. Renvoie l'écriture
    décimale de l'entier positif dont la représentation binaire
    est donnée par le tableau T
    """
```

Exemple :

```
>>> convertir([1, 0, 1, 0, 0, 1, 1])
83
>>> convertir([1, 0, 0, 0, 0, 0, 1, 0])
130
```

EXERCICE 2 (4 points)

La fonction `tri_insertion` suivante prend en argument une liste `L` et trie cette liste en utilisant la méthode du tri par insertion. Compléter cette fonction pour qu'elle réponde à la spécification demandée.

```
def tri_insertion(L):
    n = len(L)

    # cas du tableau vide
    if ...:
        return L

    for j in range(1,n):
        e = L[j]
        i = j

        # A l'étape j, le sous-tableau L[0,j-1] est trié
        # et on insère L[j] dans ce sous-tableau en déterminant
```

```
        # le plus petit i tel que 0 <= i <= j et L[i-1] > L[j].
        while  i > 0 and L[i-1] > ...:
            i = ...

        # si i != j, on décale le sous tableau L[i,j-1] d'un cran
        # vers la droite et on place L[j] en position i
        if i != j:
            for k in range(j,i,...):
                L[k] = L[...]
            L[i] = ...
    return L
```

Exemples :

```
>>> tri_insertion([2,5,-1,7,0,28])
[-1, 0, 2, 5, 7, 28]
>>> tri_insertion([10,9,8,7,6,5,4,3,2,1,0])
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
"""
Author: Pascal Padilla
Source: correction de l'exercice 1 du sujet 33 des épreuves pratiques NSI 2022

Remarques:
    * calcul à l'aide des valeurs `0` et `1` du tableau :
    tableau  : [1, 0, 1, 0, 0, 1, 1]
    nb_decimal = 1×2^6 + 0×2^5 + 1×2^4 + 0×2^3 + 0×2^2 + 1×2^1 + 1×2^0

    * difficulté: les indices du tableau ne correspondent PAS aux exposants
    tableau  : [1, 0, 1, 0, 0, 1, 1]
    indices  :  0  1  2  3  4  5  6
    exposants:  6  5  4  3  2  1  0

    * 2 méthodes :
        1. trouver une relation entre indice et exposant
        2. trouver une astuce algorithmique

    * Méthode 1.
    Il faut se servir de la taille (ici `7`).
    indices  :      0|     1|     2|     3|     4|     5|     6
    exposants:      6|     5|     4|     3|     2|     1|     0
    exposants:  7-1-0|7-1-1|7-1-2|7-1-3|7-1-4|7-1-5|7-1-6

    comme cela on a la relation: exposant = 7-1 - indice

    * Méthode 2.
    Il faut arriver à récupérer les valeurs du tableau par la fin.
    Ohhhh ! c'est une pile.
    Et c'est exactement ce que fait la méthode .pop() sur les tableaux !
"""



def convertir(T):
    """
    MÉTHODE 1. relation mathématique

    T est un tableau d'entiers, dont les éléments sont 0 ou 1 et
    représentant un entier écrit en binaire. Renvoie l'écriture
    décimale de l'entier positif dont la représentation binaire
    est donnée par le tableau T

    Tests et exemples:
    >>> convertir([1, 0, 1, 0, 0, 1, 1])
    83
    >>> convertir([1, 0, 0, 0, 0, 0, 1, 0])
    130
    """
    nb_decimal = 0
    n = len(T)
    for i in range(n):
        nb_decimal += T[i] * 2**(n - i - 1)
    return nb_decimal



def convertir(T):
    """
    MÉTHODE 2. voir un tableau comme un pile

    T est un tableau d'entiers, dont les éléments sont 0 ou 1 et
    représentant un entier écrit en binaire. Renvoie l'écriture
    décimale de l'entier positif dont la représentation binaire
    est donnée par le tableau T

    Tests et exemples:
    >>> convertir([1, 0, 1, 0, 0, 1, 1])
    83
    >>> convertir([1, 0, 0, 0, 0, 0, 1, 0])
    130
    """
    nb_decimal = 0
```

```
    n = len(T)
    for i in range(n):
        coef = T.pop()
        nb_decimal += coef * 2**i

    return nb_decimal


# vérification avec des assertions
assert convertir([1, 0, 1, 0, 0, 1, 1]) == 83
assert convertir([1, 0, 0, 0, 0, 0, 1, 0]) == 130

# vérification avec des affichages
print(convertir([1, 0, 1, 0, 0, 1, 1]))
print(convertir([1, 0, 0, 0, 0, 0, 1, 0]))

# vérification avec doctest
from doctest import testmod
testmod()
```

```python
"""
Author: Pascal Padilla
Source: correction de l'exercice 2 du sujet 33 des épreuves pratiques NSI 2022

Remarques:
"""

def tri_insertion(L):
    # nombre de valeurs
    n = len(L)

    # cas du tableau vide
    if n == 0:
        return L

    for j in range(1, n):
        e = L[j]
        i = j

    # A l'étape j, le sous-tableau L[0, j-1] est trié
    # et on insère L[j] dans ce sous-tableau en déterminant
    # le plus petit i tel que 0 <= i <= j et L[i-1] > L[j].
        while  i > 0 and L[i-1] > e:
            i = i - 1

        # si i != j, on décale le sous tableau L[i,j-1] d¿un cran
        # vers la droite et on place L[j] en position i
        if i != j:
            for k in range(j,i,-1):
                L[k] = L[k-1]
            L[i] = e
    return L


# vérification avec des assertions
assert tri_insertion([2,5,-1,7,0,28]) == [-1, 0, 2, 5, 7, 28]
assert tri_insertion([10,9,8,7,6,5,4,3,2,1,0]) == [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# vérification avec des affichages
print(tri_insertion([2,5,-1,7,0,28]))
print(tri_insertion([10,9,8,7,6,5,4,3,2,1,0]))
```