BACCALAUREAT

SESSION 2022

Épreuve de l'enseignement de spécialité

NUMERIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°07

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)

Écrire une fonction `conv_bin` qui prend en paramètre un entier positif `n` et renvoie un couple `(b,bit)` où :

- `b` est une liste d'entiers correspondant à la représentation binaire de `n;`
- `bit` correspond aux nombre de bits qui constituent `b`.

Exemple :

```
>>> conv_bin(9)
([1,0,0,1],4)
```

Aide :

- l'opérateur `//` donne le quotient de la division euclidienne : `5//2` donne `2 ;`
- l'opérateur `%` donne le reste de la division euclidienne : `5%2` donne `1 ;`
- `append` est une méthode qui ajoute un élément à une liste existante :

  Soit `T=[5,2,4]`, alors `T.append(10)` ajoute 10 à la liste `T`. Ainsi, `T` devient `[5,2,4,10]`.

- `reverse` est une méthode qui renverse les éléments d'une liste.

  Soit `T=[5,2,4,10]`. Après `T.reverse()`, la liste devient `[10,4,2,5]`.

On remarquera qu'on récupère la représentation binaire d'un entier `n` en partant de la gauche en appliquant successivement les instructions :

```
b = n%2
n = n//2
```

répétées autant que nécessaire.

EXERCICE 2 (4 points)

La fonction `tri_bulles` prend en paramètre une liste `T` d'entiers non triés et renvoie la liste triée par ordre croissant.

Compléter le code Python ci-dessous qui implémente la fonction `tri_bulles`.

```
def tri_bulles(T):
    n = len(T)
    for i in range(...,...,-1):
        for j in range(i):
            if T[j] > T[...]:
                ... = T[j]
                T[j] = T[...]
                T[j+1] = temp
    return T
```

```python
from doctest import testmod


def conv_bin(n):
    """ Convertir un entier en binaire et dénombrer les bits.

    Args:
        n (int): entier positif à convertir

    Returns:
        tuple: couple de la forme (b: list, bit: int) avec
          * b un tableau d'entier correspondant à la rep binaire de n
          * bit le nombre de bits qui constituent b

    Tests et Exemples:
    >>> conv_bin(9)
    ([1, 0, 0, 1], 4)
    """
    b = []
    bits = 0
    while n > 0:
        current_b = n % 2
        n = n // 2
        bits += 1
        b.append(current_b)
    b.reverse()
    return (b, bits)


assert conv_bin(9) == ([1, 0, 0, 1], 4)
testmod()
```

```python
from doctest import testmod


def tri_bulles(T: list) -> list:
    """ Tri d'un tableau suivant l'algo de tri à bulles

    Args:
        T (list): tableau d'entiers

    Returns:
        list: permutation de T triée

    Tests et Exemples:
    >>> tri_bulles([10, 4, 3, 9, -10, 100])
    [-10, 3, 4, 9, 10, 100]
    """
    n = len(T)
    # parcours du tableau
    # en commençant par la dernière valeur (inclue)
    # et se terminant à la case 0 (inclue)
    # et en parcourant à l'envers, ie en allant de -1 en -1
    #
    # il y a deux idées :
    # (1) placer à la dernière case de la partie non triée
    #     la valeur la plus grande
    # (2) permuter la valeur la plus grande case par case
    #     pour l'amener successivement à la dernière case non triée
    for i in range(n-1, -1, -1):
        for j in range(i):
            if T[j] > T[j+1]:
                # lorsque la valeur courante est plus grande
                # que la valeur suivante : on les permute
                temp = T[j]
                T[j] = T[j+1]
                T[j+1] = temp
    return T



assert tri_bulles([10, 4, 3, 9, -10, 100]) == [-10, 3, 4, 9, 10, 100]
testmod()
```