D) Somme consécutive …

Écrire la fonction **consecutive_sum(L, S)** qui vérifie s'il existe une suite d'éléments consécutifs dont la somme est **S** (avec S > 0) dans la liste **L**. La liste **L** ne contient que des entiers positifs ou nuls.

Exemples d'applications :

```
>>> consecutive_sum([3, 1, 4, 2, 4], 10)
True

>>> consecutive_sum ([3, 7, 6], 6)
True

>>> consecutive_sum ([], 6)
False

>>> consecutive_sum ([3, 7, 6], 9)
False
```

E) Séparation …

Écrire la fonction **separate(L1, L2, n)** qui prend en paramètre deux listes triées en ordre croissant **L1** et **L2** et un entier **n**, et qui retourne un couple de listes dont :

— la première liste contient les éléments plus petits que **n**
— la seconde liste contient les éléments plus grands ou égaux que **n**

Ces deux listes résultats sont triées en ordre croissant.

Exemples d'applications :

```
>>> separate ([1, 3, 12, 31, 42], [5, 16, 28], 12)
([1, 3, 5], [12, 16, 28, 31, 42])

>>> separate ([1, 3, 12, 31, 42], [5, 16, 28], 42)
([1, 3, 5, 12, 16, 28, 31], [42])

>>> separate ([1, 2, 3], [2, 5, 12, 42], 1)
([], [1, 2, 2, 3, 5, 12, 42])

>>> separate ([], [2, 5, 12, 42], 6)
([2, 5], [12, 42])
```