III.4 Jeudi 16 avril ♥

1–❑ Écrire la fonction `Distances(X:list,Z:list[list])->list[float]` qui renvoie la liste des distances entre `X` et chaque liste de la liste `Z`. On n'utilisera pas de fonctions auxiliaires.

Solution: sans la fonction précédente

```
def Distances(X,Z):
    LD=[] # liste des distances
    for i in range(len(Z)): # parcours des listes de Z
        # calcul de la distance entre Z[i] et X
        dist_i=0 # initialisation de la distance entre Z[i] et X
        for j in range(len(X)): # ou len(Z[i])
            dist_i=dist_i+(X[j]-Z[i][z])**2 # somme à calculer
        LD.append(sqrt(dist_i))
    return LD
```

2–❑ Écrire une fonction `EstTriee(L)` qui renvoie `True` si la liste de nombre `L` est triée par ordre croissant et `False` sinon.

Solution:

```
def EstTriee(L):
    for i in range(len(L)-1):
        if L[i+1]<L[i]: # suivant plus petit que le précédent
            return False
    return True # on est arrivé au bout de la liste avec toujours L[i+1]>=L[i] :
   elle est triée !
```

3–❑ Quelle est la complexité de cette fonction ?

Solution: On parcourt la liste une fois, le test est en coût constant $O(1)$. La complexité est linéaire en la longueur de la liste.