VÉRIFICATION DE L'ALGORITHME

- Initialisation
  - ✓ Quand i vaut 2, le sous-ensemble T[1 … i-1] est en fait un singleton T[1] et étant unique, il est par définition, déjà trié. Vérifié.

- Conservation
  - ✓ Sur la boucle extérieure, la propriété est vérifiée car les éléments sont triés, mais il faudrait montrer que la boucle intérieure vérifie formellement cela avec la condition tant que. Nous admettons la validité.

- Terminaison
  - ✓ La boucle prend fin quand i dépasse la taille du tableau, soit i = n+1. Ce qui signifie qu'en voulant faire l'itération pour i = n+1, on se retrouve avec un tableau T[1 … n]. Le tableau conserve les mêmes éléments et il reste entier. De plus, il est trié, donc l'algorithme est correct.

ANALYSE DES RESSOURCES

TriParInsertion(tableau T)

pour $i \leftarrow 2$ à $longueur[T]$
    faire clé $\leftarrow T[i]$
        $j \leftarrow (i-1)$
        tant que $j > 0$ et $T[j] >$ clé
            faire $T[j+1] \leftarrow T[j]$
                $j \leftarrow (j-1)$
        $T[j+1] \leftarrow$ clé

coût	fois
c_1	n
c_2	n-1
c_3	n-1
c_4	$\sum_{i=2}^{i=n} t_i$
c_5	$\sum_{i=2}^{i=n} (t_i - 1)$
c_6	$\sum_{i=2}^{i=n} (t_i - 1)$
c_7	n-1

$$f(n) = c_1 n + c_2(n-1) + c_3(n-1) + c_4 \sum_{i=2}^{i=n} t_i + c_5 \sum_{i=2}^{i=n} (t_i - 1) + c_6 \sum_{i=2}^{i=n} (t_i - 1) + c_7(n-1)$$