VÉRIFICATION DE L'ALGORITHME

- Initialisation
  - ✓ Quand i vaut 1, le sous-ensemble T[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. On peut voir que l'on a le sous-tableau T[2…n] et l'on fait une comparaison qui ne déborde pas en restant sur T[1…n]. On place donc le plus petit élément au début. En repassant sur la boucle extérieure on itére sur i l'on parcourt donc T[3…n] en ayant trié T[1…2]. Ce qui signifie que le tri est conservé.

- 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

bubbleSort(T)

1 pour $i \leftarrow 1$ à $longueur[T]$
2 faire pour $j \leftarrow longueur[T]$ à $i + 1$ par décrémentation
3 faire si $T[j] < T[j-1]$
4 alors permuter $T[j] \leftrightarrow T[j-1]$

coût	fois
c_1	n
c_2	n-1
c_3	$\sum_{j=2}^{j=n} (t_j - 1)$

$$f(n) = c_1 n + c_2(n-1) + \sum_{j=2}^{j=n} (t_j - 1)$$