Algorithmie -Niveau 1ère- Tri selection

1 Spécifications

1.1 Algorithme

Ecrire en pseudo code une fonction nommée **tri_ selection(T)** qui prend en argument un tableau T et qui renvoie le tableau T trié en place.
La méthode utilisée pour trier est la méthode dite du **tri selection** :
On commence avec une liste déjà triée vide. On itère sur la liste et, à chaque tour on range le plus petit élément de la liste non triée à droite de la liste triée.

1.2 Implémentation Python

Ecrire en langage Python une fonction nommée **tri_ selection(T)** qui prend en argument une liste Python T et qui renvoie la liste Python T trié en place.
La méthode utilisée pour trier est la méthode dite du **tri selection** :
On commence avec une liste déjà triée vide. On itère sur la liste et, à chaque tour on range le plus petit élément de la liste non triée à droite de la liste triée.

2 Résolution

Pour vous aider à visualiser le principe du tri selection, voici une petite vidéo :

VIDÉO Un animation sur le tri sélection - Cliquez ici !

2.1 L'algorithme

```
1 FONCTION tri_ selection(T :tableau d'entiers)
2 |  POUR i DE 0 A longueur(T) − 2 FAIRE
3 |  |  min ← i
4 |  |  POUR j DE i+1 À longueur(T)-1 FAIRE
5 |  |  |  SI T[j] < T[min] ALORS
6 |  |  |  |  min ← j
7 |  |  SI min ≠ i ALORS
8 |  |  |  échanger T[i] et T[min]
```

2.2 Implémentation en Python

```
def tri_selection(T):
    for i in range(0,len(T)-1):
        min = i
        for j in range (i+1,len(T)):
```

```
            min = j
        if min != i:
            T[i],T[min] = T[min],T[i]
```

☛ Il faut être capable de :

- Ecrire les pré-conditions
- Ecrire les post-conditions
- Donner un jeu de tests pertinents
- Utiliser des asserts pour vérifier ces tests
- connaître la complexité temporelle
- Démontrer la terminaison de l'algorithme

2.3 Complexité temporelle

Propriété 0.1

La complexité temporelle du tri selection est, dans le pire des cas, en $\mathcal{O}(n^2)$