Tri par sélection

I Description

Sur un tableau de n éléments (numérotés de 0 à n-1), le principe du tri par sélection est le suivant :

- rechercher le plus petit élément du tableau, et l'échanger avec l'élément d'indice 0 ;
- rechercher le second plus petit élément du tableau, et l'échanger avec l'élément d'indice 1 ;
- continuer de cette façon jusqu'à ce que le tableau soit entièrement trié.

On dispose par exemple de la liste à 5 valeurs suivante :

indices	**0**	**1**	**2**	**3**	**4**
Valeurs	5	4	7	8	2

On parcours des indices 0 à 4 et le minimum est à l'indice 4.
On échange les valeurs de l'indice 0 et l'indice 4.
L'indice 0 est trié.

	2	4	7	8	5

On parcours des indices 1 à 4 et le minimum est à l'indice 1.
On échange les valeurs de l'indice 1 et l'indice 1.
Les indices de 0 à 1 sont trié.

	2	4	7	8	5

On parcours des indices 2 à 4 et le minimum est à l'indice 4.
On échange les valeurs de l'indice 2 et l'indice 4.
Les indices de 0 à 2 sont trié.

	2	4	5	8	7

On parcours des indices 3 à 4 et le minimum est à l'indice 4.
On échange les valeurs de l'indice 3 et l'indice 4.
Les indices de 0 à 3 sont trié.

	2	4	5	7	8

Le dernier élément est déjà à sa place, puisque l'on a trié tous les autres avant lui.

	2	4	5	7	8

La liste est entièrement triée.

II Exercices

indices	**0**	**1**	**2**	**3**	**4**	**5**	**6**
Valeurs	2	4	5	9	1	3	7
	1	4	5	9	2	3	7
	1	**2**	5	9	4	3	7
	1	**2**	**3**	9	4	5	7
	1	**2**	**3**	**4**	9	5	7
	1	**2**	**3**	**4**	**5**	9	7
	1	**2**	**3**	**4**	**5**	**7**	9
	1	**2**	**3**	**4**	**5**	**7**	**9**

III Pseudo-code, variantes et code python

En pseudo-code, l'algorithme s'écrit ainsi :

```
  procédure tri_selection(tableau t)
      n ← longueur(t)
      pour i de 0 à n - 2
          min ← i
          pour j de i + 1 à n - 1
              si t[j] < t[min], alors min ← j
          fin pour
          si min ≠ i, alors échanger t[i] et t[min]
      fin pour
  fin procédure
```

Une variante consiste à procéder de façon symétrique, en plaçant d'abord le plus grand élément à la fin, puis le second plus grand élément en avant-dernière position, etc.

```
  procédure tri_selection_symetrique(tableau t)
      n ← longueur(t)
      pour i de 0 à n - 2
          max ← i
          pour j de 0 à n - 1 - i
              si tab[max] < tab[j], alors max ← j
          fin pour
          si max ≠ i, alors échanger t[n - 1 – i] et t[max]
      fin pour
  fin procédure
```

Le tri par sélection peut aussi être utilisé sur des listes. Le principe est identique, mais au lieu de déplacer les éléments par échanges, on réalise des suppressions et insertions dans la liste.

En python :

```
def echange(tab, i, j):
    """ Échange les éléments d'indice i et j dans la liste tab. """
    temp = tab[i]
    tab[i] = tab[j]
    tab[j] = temp

def tri_par_selection(tab):
    """
    Tri la liste tab avec un tri par selection

    >>> tri_par_selection([4,3,2,0,1])
    [0, 1, 2, 3, 4]
    """

    for i in range(len(tab) - 1):
        indice_min = i
        for j in range(i, len(tab)):
            if tab[indice_min] > tab[j]:
                indice_min = j
        tab.insert(i, tab.pop(indice_min)) #echange(tab, i, indice_min)
    return tab
```