Informatique en CPGE (2018-2019)
TD 4 : algorithmes de tri

Exercice 1 : tri par sélection

Le tri par sélection d'un tableau à n éléments $[t_0, \ldots, t_{n-1}]$ se fait comme suit. On cherche la plus petite donnée et on la place en première position, puis on cherche la plus petite donnée parmi les données restantes et on la place en deuxième position, et ainsi de suite.

L'algorithme consiste donc à faire varier i de 0 à $n-2$. Pour chaque itération de i, on recherche dans la tranche $[t_i, \ldots, t_{n-1}]$ le plus petit élément et on l'échange avec t_i.

L'algorithme de tri par sélection est souvent utilisé pour trier des objets, comme des cartes à jouer, des livres, etc. à la main.

Algorithme :

```
pour i variant de 0 à n −2
    indiceMini ← i    (indice du plus petit élément)
    mini ← liste[i]
    pour j variant de i +1 à n −1
        si liste[j] < mini
            indiceMini ← j
            mini ← liste[j]
    échanger liste[i] et liste[indiceMini].
```

1. Ecrire une fonction **tri_selection** qui prend en argument une liste de nombres et renvoie la liste ordonnée suivant l'ordre croissant ; tester cette fonction sur différentes listes.
2. Comparer et commenter le résultat obtenu si l'on interrompt après k étapes l'exécution de l'algorithme du tri par sélection et celle de l'algorithme du tri par insertion étudié en cours.

   Faire le test sur la liste t=[8, 3, 4, 6, 9, 1, 2, 5, 7] avec les deux programmes, en affichant l'état de la liste après chaque passage dans la boucle principale.
3. Quelle est la complexité du tri par sélection dans le pire des cas, dans le meilleur des cas et en moyenne ?

Exercice 2 : tri à bulles

Avec l'algorithme du tri à bulles, on permet seulement l'échange de deux éléments consécutifs du tableau à trier. L'algorithme se résume à :

- on parcourt le tableau et si deux éléments consécutifs sont rangés dans le désordre, on les échange ;
- si un échange a eu lieu pendant le parcours, on recommence l'opération,
- sinon, le tableau est trié, on arrête.

Les éléments les plus grands remontent ainsi dans la liste comme des bulles d'air remontent à la surface d'un liquide. Cet algorithme permet de trier n'importe quel tableau.

1. Effectuer à la main un tri à bulles du tableau : [18, 10, 15, 11, 5]
2. Quel est la complexité du tri à bulles sur un tableau déjà ordonné à n éléments ?
3. Le pire des cas est lorsque le tableau est rangé dans l'ordre décroissant : $[n, n-1, \ldots, 3, 2, 1]$.
   (a) Déterminer le nombre de permutations nécessaires pour amener le nombre n à la position correcte ?
   (b) Ensuite, combien de permutations sont nécessaires pour amener $n-1$ à la bonne place ? Et pour le nombre $n-2$?

(c) Quel est le nombre total de permutations nécessaires pour trier un tableau de taille n rangé initialement en ordre décroissant.

4. Ecrire alors une fonction **tri_bulles** qui prend en argument une liste de nombres et renvoie la liste ordonnée suivant l'ordre croissant. Tester cette fonction sur les listes [18, 10, 5, 11, 19, 15], [1, 2, 3, 4, 5, 6] et [6, 5, 4, 3, 2, 1] en affichant l'état de la liste après chaque passage dans la boucle principale.