Licence S.T.A. - semestre 2
Informatique - A.P.I.1
TD n°5 tris (1)

Dans toute cette feuille on suppose que le type `TABLEAU` a été déclaré à l'aide des déclarations suivantes :

```
type INDICE =  ..... ;    // un intervalle d'entiers
     ELEMENT = ........ ; // un type auquel on peut appliquer la relation d'ordre <=
     TABLEAU = array[INDICE] of ELEMENT ;
```

Exercice 1 : *Dérouler l'algorithme du tri sélection*

Étant donné le tableau `t`

4	3	7	1	2	5	3

donnez les valeurs successives que prend ce tableau `t` lors du tri sélection.

Exercice 2 : *Implémenter le tri sélection en Pascal*

Q 1 . Écrivez la fonction `ind_min` qui calcule l'indice du minimum dans le sous-tableau `t[a..b]` du tableau `t`.

Q 2 . Écrivez la procédure `trier_select` qui trie le tableau `t` en utilisant l'algorithme du tri sélection.

Exercice 3 : *Une amélioration (?) du tri sélection*

Pourquoi lorsqu'on cherche l'indice du minimum, ne profiterions-nous pas de chercher aussi celui du maximum, ainsi avec un seul parcours on pourrait positionner le plus petit et le plus grand élément du tableau, on avancerait ainsi deux fois plus vite.

Q 1 . Peut-on écrire une fonction qui calcule ces deux indices en un seul parcours ? Que proposez-vous comme sous-programme pour ce calcul (toujours en un seul parcours) ?

Q 2 . Écrivez cette nouvelle version du tri sélection.

Q 3 . A-t-on vraiment amélioré le tri en nombre de comparaisons et d'échanges ?

Exercice 4 : *Dérouler l'algorithme du tri insertion*

Étant donné le tableau `t`

4	3	7	1	2	5	3

donnez les valeurs successives que prend ce tableau `t` lors du tri insertion.

Exercice 5 : *Implémenter le tri insertion en Pascal*

Q 1 . Écrivez la procédure `inserer` qui insére `t[k]` dans `t[a..k]` (le sous-tableau `t[a..k-1]` étant supposé trié).

Q 2 . Écrivez la procédure `trier_insert` qui trie le tableau `t` en utilisant l'algorithme du tri insertion.

Exercice 6 : Un troisième tri : le tri à bulle

Q 1 . Voici une procédure

```
procedure faireUneEtape(const a,k : INDICE ; var t : TABLEAU) ;
// C.U. : t[k+1..b] trié et t[a..k]<=t[k]
var i : INDICE ;
begin
  for i := a to k-1 do begin
    if t[i]>t[i+1] then begin
      echanger(t[i],t[i+1]) ;
    end {if} ;
  end {for} ;
end {faireUneEtape} ;
```

Déroulez l'instruction `faireUneEtape(1,5,t)` sur le tableau (indicé de 1 à 7)

4	3	7	1	3	7	9

Q 2 . Que pouvez-vous dire de `t[k]` après l'instruction `faireUneEtape(a,k,t) ;` ?

Q 3 . Combien fait-on d'échanges dans l'instruction `faireUneEtape(a,k,t) ;` dans le pire des cas ? Dans le meilleur des cas ? Que peut-on en conclure dans ce dernier cas pour `t` ?

Q 4 . En déduire un algorithme de tri qui utilise cette procédure `faireUneEtape(a,k,t) ;`

Q 5 . Comparez ce nouveau tri avec le tri sélection.