


Institut Supérieur d'Informatique et de Mathématiques 	Année Universitaire : 2023-2024
	<p style="text-align: center;">Examen</p> Matière : ASD 1 Filière : L1 Info Enseignant : Sakka Rouis Taoufik

Exercice 1 : (5+3 Points)

L'algorithme du **Tri-Idiot** peut être formulé comme suit : tirez deux indices différents au hasard, compris entre le premier et le dernier indice, échangez éventuellement les deux éléments associés, et vérifiez si le tableau est trié. Si le tableau est trié, on s'arrête ; sinon, on recommence.

Indication : vous pouvez utiliser la fonction rand de la bibliothèque stdlib.h comme suit :

```
int x = rand ( ) % 20; /*x prend une valeur entre 0 et 19 */
```

1. Proposer une implémentation pour cet algorithme de tri.
2. Comparez cet algorithme avec les trois autres solutions vues en cours. (Écrivez avec des phrases **claires** sinon 0)

Exercice 2 : (3+3 Points)

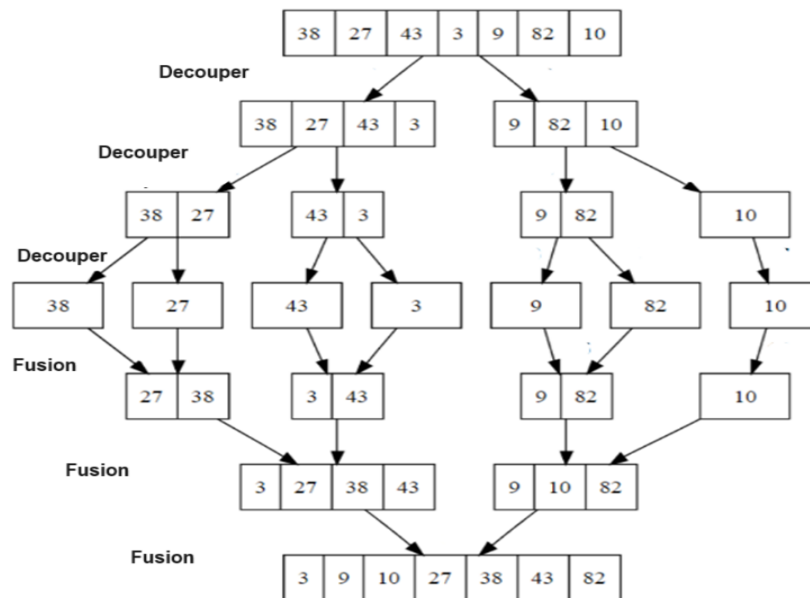
<pre>#include <stdio.h> #include <stdlib.h> #define n 10 int a[n] = {5, 2, 52, 3, 44, 4, 12, 52, 34, 7}; int info; unsigned Controllo (unsigned x) { return a[x] == info; } int Ricercare () { unsigned x, y; unsigned p, q; p = 0; q = n; x = p; y = q;</pre>	<pre>while (x != y) { if (Controllo (x)) y = x; else x++; } return (x==n ? -1 : x); } void main () { printf("donner un entiere \n"); scanf("%d", &info); printf("%d \n", Ricercare ()); }</pre>
---	---

1. Expliquer l'organisation générale du programme cité ci-dessus.
2. Identifier le problème censé être résolu par la fonction « Ricercare » précédente et la comparer **aux** algorithmes résolvant le même type de problème.

Exercice 3 : (6 Points)

Le principe du tri par fusion peut être résumé comme suit :

1. Découper le tableau $T[g..d]$ à trier en deux sous-tableaux $T[g..(g+d)/2]$ et $T[(g+d)/2+1..d]$.
2. Trier récursivement les deux sous-tableaux $T[g .. (g+d)/2]$ et $T[(g+d)/2+1 .. d]$, ou ne rien faire s'ils sont de taille 1 (condition d'arrêt de la récursion).
3. Fusionner les deux sous-tableaux triés $T[g .. (g+d)/2]$ et $T[(g+d)/2+1 .. d]$ de manière à ce que le tableau final soit trié.



```
#include <stdio.h>
```

```
void Fusion (int T [ ], int g, int m, int d);
```

```
void TriFusion (int T [ ], int g, int d){
```

```
    int m;
```

```
    if (g<d){
```

```
        m=(g+d)/2;
```

```
        TriFusion (T, g, m) ;
```

```
        TriFusion (T, m+1, d) ;
```

```
        Fusion (T, g, m, d) ;
```

```
    }
```

```
}
```

```
void Tri ( int T [ ], int n){
```

```
    TriFusion (T, 0, n-1) ;
```

```
}
```

```
void main (){
```

```
    int N=7 ;
```

```
    int i, T [ ]={38, 27, 43, 3, 9, 82,10} ;
```

```
    Tri (T, N) ;
```

```
    for (i=0 ; i<N ; i++)
```

```
        printf ("%d \t ", T [i] ) ;
```

```
}
```

1. On vous demande de proposer une implémentation pour la fonction "Fusion" permettant de fusionner les deux sous-tableaux (supposé triés) $T[g .. m]$ et $T[m+1 .. d]$ de sorte à ce que le tableau final soit trié.