



---

---

## Atelier de Programmation \*\*\*\*\* TP1

---

---

### Objectifs:

- Pratiquer la décomposition modulaire
- Utiliser les fonctions en C
- Manipuler les pointeurs, les passages par adresse et par valeur
- Utiliser des paramètres formels In, Out et In/Out
- Exploiter les algorithmes de recherche
- Calculer la complexité des algorithmes

### Exercice 1 :

Soient deux tableaux **T1** et **T2** de taille respectivement **N1** et **N2** avec  $5 < N1, N2 < 100$ . Les tableaux contiennent des entiers distincts.

Ecrire un programme en C permettant de calculer l'intersection entre T1 et T2

1. Décomposer le problème en fonctions
2. Implémenter les fonctions en C
3. Proposer une fonction main de test

Exemple :

N1=5

T1

12	-5	71	-10	56
----	----	----	-----	----

NI=2

I

12	-10
----	-----

N2=7

T2

30	53	12	-25	26	-10	47
----	----	----	-----	----	-----	----

### Exercice 2 :

Soient deux tableaux **T1** et **T2** de taille respectivement **N1** et **N2** avec  $5 < N1, N2 < 100$ . Ils contiennent des entiers distincts et triés dans l'ordre croissant. L'objectif est le calcul de l'intersection entre T1 et T2.

1. La solution de l'exercice 1 est-elle une solution pour l'exercice 2? Si oui, est-elle une solution optimale?
2. Quelles sont les fonctions à rectifier par rapport à l'exercice 1 ?
3. Implémenter les fonctions à rectifier en C

Exemple :

N1=5					
T1	-10	-5	12	56	71
NI=2					
I	-10	12			

N2=7

T2	-25	-10	12	26	30	47	53
----	-----	-----	----	----	----	----	----

### Exercice 3 :

Soient les tableaux **T1** contenant **N1** entiers distincts et **T2** contenant **N2** entiers distincts et triés dans l'ordre croissant avec  $5 < N1, N2 < 100$ . L'objectif est de calculer l'intersection entre T1 et T2.

1. Discuter les solutions des deux exercices précédents par rapport à la configuration initiale des tableaux T1 et T2.
2. Implémenter en C la fonction permettant de calculer l'intersection.

Exemple :

N1=5					
T1	12	-5	71	-10	56
NI=2					
I	12	-10			

N2=7							
T2	-25	-10	12	26	30	47	53

### Exercice 4 :

1. Discuter la complexité du traitement de calcul d'intersection des solutions données dans les trois exercices précédents.
2. Calculer le temps d'exécution en utilisant les 3 solutions données pour les trois exercices sur une configuration des deux tableaux **T1** et **T2** comme elle est indiquée dans l'exercice 2. Sachant que les tailles des tableaux données dans les exemples sont petites pour avoir une différence remarquable dans le temps d'exécution des 3 solutions, proposez une démarche permettant d'avoir une telle différence.

## Complément de cours

### Calculer le temps d'exécution d'un code

#### time.h

Le langage C fournit un certain nombre de fonctions permettant de lire l'heure/date courante et de convertir la valeur lue en structure ou en chaîne de caractères. Ces éléments sont définis et déclarés dans le header standard `<time.h>`

#### Fonction clock()

La fonction `clock_t clock(void)` permet de connaître le nombre de fractions de temps machine écoulées depuis le début du programme. La valeur retournée peut être convertie en secondes en la divisant par la constante `CLOCKS_PER_SEC`. Elle peut servir à évaluer l'espace de temps entre deux événements.

### **Exemple 1 :**

```
#include <stdio.h>
#include <time.h>
int main()
{
    printf("Temps ecoule = %d millisecondes\n",clock());
    return 0;
}
```

*Affichage :*

Temps ecoule = 0 millisecondes

### **Exemple 2 :**

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

int main()
{
    long i,t;
    for(i=0;i<500000000;i++);
    t=clock();
    printf("Temps ecoule = %d millisecondes\n",t);
    printf("Temps ecoule = %f secondes\n",((float)t)/CLOCKS_PER_SEC);
    return 0;
}
```

*Affichage :*

Temps ecoule = 1061 millisecondes  
Temps ecoule = 1.061 secondes

### **Exemple 3 :**

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

int main()
{
    long i;
    time_t start, end;
    start = time(NULL);
    for(i=0;i<500000000;i++);
    end = time(NULL);
    printf("Temps ecoule = %.3f secondes\n",difftime(end, start));
    return 0;
}
```

*Affichage :*

Temps ecoule = 1.000 secondes

### **Exemple 4 :**

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

int main()
{
    long i,val,t0,t1,t2;
    t0=clock();
    printf("val= ");
    scanf("%ld",&val);
}
```

```

    t1=clock();
    for(i=0;i<500000000;i++);
    t2=clock();
    printf("t0= %ld millisecondes\n",t0);
    printf("t1= %ld millisecondes\n",t1);
    printf("t2= %ld millisecondes\n",t2);
    printf("Temps ecoule pour la boucle= %d millisecondes\n",t2-t1);
    return 0;
}

```

**Affichage :**

```

t0 = 0 millisecondes
t1 = 1709 millisecondes
t2 = 2790 millisecondes
Temps ecoule pour la boucle = 1081 millisecondes

```

**Exemple 5 :**

```

#include <stdio.h>
#include <time.h>

int main()
{
    struct tm* local;
    time_t t = time(NULL);
    // Get the localtime
    local = localtime(&t);
    printf("Local time and date: %s\n",
        asctime(local));
    return 0;
}

```

**Affichage :**

```

Local time and date : Tue Feb 04 22:39:29 2020

```

**Exemple 6 :**

```

/* Un programme permettant de remplir un tableau T par des entiers
aléatoires dans l'intervalle [0..30000[
*/
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#define taille 10000
void main()
{
    int N=30000,i;
    int T[taille];
    int randomValue;
    srand (time (NULL));
    for(i=0;i<taille;i++)
    {
        randomValue= (int)(rand() / (double)RAND_MAX * (N - 1));
        T[i]=randomValue;
    }
    for(i=0;i<taille;i++)
    {
        printf("T[%d]=%d\n",i,T[i]);
    }
}

```

## Références

- [https://fr.wikiversity.org/wiki/Fonctions\\_de\\_base\\_en\\_langage\\_C/time.h](https://fr.wikiversity.org/wiki/Fonctions_de_base_en_langage_C/time.h)
- <https://www.geeksforgeeks.org/time-h-header-file-in-c-with-examples/>
- <https://www.youtube.com/watch?v=qr5utfOV8Fs> ==> difftime and clock Functions in C Programming Language Video Tutorial
- <https://nicolasj.developpez.com/articles/libc/hasard/> ==> Générateur des entiers aléatoires

FSS-LSI