

Devoir surveillé <input type="checkbox"/>	Examen <input checked="" type="checkbox"/>	Session : principale <input checked="" type="checkbox"/> de contrôle <input type="checkbox"/>
Matière : Algorithmique et structures de données I	Semestre: 1	
Enseignant : Aymen Sellaouti et Wided Miled	Date: 06/01/2015 à 15h00	
Filière(s) : MPI	Durée: 1h30	
Nombre de pages : 2	Documents : autorisés <input type="checkbox"/> non autorisés <input checked="" type="checkbox"/>	

Exercice 1 :

- 1- Ecrire en algorithmique et puis traduire en C, la fonction ou procédure **getMinMax** qui permet d'extraire **les indices** des valeurs maximales et minimales d'une partie d'un tableau d'entiers de taille N, comprise entre un indice inférieur b_inf et un indice supérieur b_sup. Si b_inf=1 et b_sup=N, la recherche s'effectuera sur la totalité du tableau.
- 2- Ecrire en algorithmique la fonction ou procédure **Permuter** qui permet d'échanger deux entiers sans utiliser de variables intermédiaires.
- 3- Ecrire en algorithmique la fonction ou procédure **TriMinMax** qui permet d'effectuer le tri MinMax d'un tableau de taille N. Le principe du tri MinMax consiste à détecter à chaque itération les valeurs minimale et maximale de la partie du tableau non encore triée et de les mettre à leur place.

Exemple :

T :

1	4	9	0	67	5
---	---	---	---	----	---

Itération 1 : Min = 0, Max =67, on permute 5 avec Max et 1 avec Min

T :

0	4	9	1	5	67
---	---	---	---	---	----

Itération 2 : Min = 1, Max =9, on permute 4 avec Min et 5 avec Max

T :

0	1	5	4	9	67
---	---	---	---	---	----

.

.

Itération p :

T :

0	1	4	5	9	67
---	---	---	---	---	----

Exercice 2 :

On dispose du fichier texte « **process.txt** » .

Chaque ligne de ce fichier contient les données suivantes :

Numéro	: 5 caractères (entier)
Nom	: 30 caractères
Heure de déclanchement	: HH:MM:SS
	HH : Heures (Entier sur 2 caractères)
	MM : Minutes (Entier sur 2 caractères)
	SS : Secondes (Entier sur 2 caractères)
Temps CPU nécessaire	: 10 caractères (entier)
Temps Entrées Sorties prévu	: 10 caractères (entier)
Code Utilisateur	: 5 caractères (entier)
Taille du processus	: 10 caractères (entier)

On vous demande de :

- 1) Définir la structure **PROCESS** relative au contenu d'une ligne du fichier **process.txt**.
Le champ heure de déclanchement doit lui aussi être défini dans une structure.

- 2) Ecrire la fonction :

```
int modif_taille(int num, int size)
{
}
```

qui permet de changer directement dans le fichier « **process.txt** » la taille relative au processus de numéro **num** avec la taille **size**. La fonction retournera 1 si le numéro est trouvé, 0 sinon.

- 3) Ecrire la fonction :

```
int load_process(PROCESS p[])
{
}
```

qui permet de charger le contenu du fichier « **process.txt** » dans un tableau de structure **p**. La valeur de retour de la fonction est le nombre d'éléments **n** chargés dans le tableau **p**.

- 4) Ecrire la fonction

```
int kill_process(PROCESS p[], int n, int num_process)
{
}
```

qui permet de supprimer du tableau **p** le processus ayant comme numéro **num_process**. La valeur de retour de la fonction est **n-1** si le processus est trouvé ou **n** si le processus n'a pas été trouvé.

- 5) Ecrire une fonction **récursive** qui calcule pour un utilisateur ayant un code donné :

- Le nombre de processus qu'il a lancé
- Le temps total CPU de ses processus

```
Void stat_user(PROCESS p[], int n, int code_user, int *nb_proc, int *temps_cpu)
{
}
```

Bon Travail