

TD1 — Programmes et modules C

Exercice 1. Lire le programme C fourni en annexe.

1. Dessiner un schéma de la structure de ce programme (modules et programme principal), en indiquant les dépendances entre modules.
2. Quel est le résultat de l'exécution du programme ?
3. Quelles sont les contraintes sur les valeurs en entrée ?

Exercice 2. Modifier le programme pour qu'il affiche la valeur maximum du tableau en entrée. Les contraintes sur les valeurs d'entrées changent-elles avec cette nouvelle spécification ?

A. Programme principal

Programme principal td1 composé d'un seul fichier td1.c :

```
1 #include "es_tableau.h"
2 #include "operations_tableau.h"
3 #include "type_tableau.h"
4
5 int main() {
6     tableau_entiers tableau;
7
8     /* Lire le tableau dans le fichier "data1.txt" */
9     lire_tableau("data1.txt", &tableau);
10    /* Renverser le tableau */
11    renverser(&tableau);
12    /* Ecrire le tableau renversé dans le fichier "data1_renverse.txt" */
13    ecrire_tableau("data1_renverse.txt", &tableau);
14 }
```

B. Paquetage type_tableau

Paquetage type_tableau composé du seul fichier « spécification » ou « interface » type_tableau.h :

```
1 #ifndef _TYPE_TABLEAU_H_
2 #define _TYPE_TABLEAU_H_
3
4 #define TAILLE_MAX 10000
5
6 /* Définition du type vecteur_entiers : tableau d'entiers de taille TAILLE_MAX */
7 typedef int vecteur_entiers[TAILLE_MAX];
8
9 /* Structure contenant un tableau (de taille TAILLE_MAX) et un entier
10    taille : le nombre d'entiers du tableau */
11 typedef struct {
12     int taille;
13     vecteur_entiers tab;
14 } tableau_entiers;
15
16 #endif /* _TYPE_TABLEAU_H_ */
```

C. Paquetage es_tableau

Paquetage es_tableau composé :

- d'un fichier « spécification » ou « interface » es_tableau.h :

```
1 #ifndef _ES_TABLEAU_H_
2 #define _ES_TABLEAU_H_
3
4 #include "type_tableau.h"
5
6 void lire_tableau(char *nomFichier, tableau_entiers *t);
7
8 void ecrire_tableau(char *nomFichier, tableau_entiers *t);
9
10 #endif /* _ES_TABLEAU_H_ */
```

- d'un fichier « corps » ou « implémentation » es_tableau.c :

```
1 #include "es_tableau.h"
2 #include <stdio.h>
3
4 void lire_tableau(char *nomFichier, tableau_entiers *t) {
5     /* Descripteur du fichier d'entrée */
6     FILE *fichier;
7     int i;
8
9     /* Ouverture du fichier en lecture */
10    fichier = fopen(nomFichier, "r");
11
12    /* Lecture de la taille du tableau */
13    fscanf(fichier, "%d", &(t->taille));
14
15    /* Lecture des valeurs du tableau */
16    for (i = 0; i < t->taille; i++) {
17        fscanf(fichier, "%d", &(t->tab[i]));
18    }
19    fclose(fichier);
20 }
21
22 void ecrire_tableau(char *nomFichier, tableau_entiers *t) {
23     /* Descripteur du fichier de sortie */
24     FILE *fichier;
25     int i;
26
27     /* Créer et ouvrir le fichier en écriture */
28     fichier = fopen(nomFichier, "w");
29
30     /* Écrire la taille du tableau dans le fichier */
31     fprintf(fichier, "%d\n", t->taille);
32
33     /* Écrire les valeurs du tableau */
34     for (i = 0; i < t->taille; i++) {
35         fprintf(fichier, "%d\n", t->tab[i]);
36     }
37     /* Fermeture du fichier */
38     fclose(fichier);
39 }
```

D. Paquetage operations_tableau

Paquetage operations_tableau composé :

- d'un fichier « spécification » ou « interface » operations_tableau.h :

```
1 #ifndef _OPERATIONS_TABLEAU_H_
2 #define _OPERATIONS_TABLEAU_H_
3
4 #include "type_tableau.h"
5
6 void renverser(tableau_entiers *t);
7
8 #endif /* _OPERATIONS_TABLEAU_H_ */
```

- d'un fichier « corps » ou « implémentation » operations_tableau.c :

```
1 #include "operations_tableau.h"
2
3 /* Permute les entiers p et q
4    Données-résultats : p, q : entiers
5    Post-condition : les valeurs de p et q ont été permutées
6 */
7 void permuter(int *p, int *q) {
8     int aux;
9
10    aux = *p;
11    *p = *q;
12    *q = aux;
13 }
14
15 /* renverse l'ordre des elements du tableau T
16    l'algorithme utilisé est le suivant
17    soit n le nombre de valeurs dans le tableau T
18    on permute l'élément d'indice 1 de T avec l'élément d'indice n
19    on permute l'élément d'indice 2 de T avec l'élément d'indice n-1
20    ...
21    on permute l'élément d'indice i1 de T avec l'élément d'indice i2,
22    avec i1<i2 et i1+i2=n+1
23    ...
24    jusqu'à la moitié du tableau c'est à dire lorsque i1>=i2
25 */
26 void renverser(tableau_entiers *t) {
27     /* Indices de parcours du tableau */
28     int i1, i2;
29
30     i1 = 0;
31     i2 = t->taille - 1;
32
33     while (i1 < i2) {
34         /* Effectuer la permutation des éléments d'indices i1 et i2 de t */
35         permuter(&(t->tab[i1]), &(t->tab[i2]));
36
37         /* Incrémenter i1 et décrémenter i2 */
38         i1 = i1 + 1;
39         i2 = i2 - 1;
40     }
41 }
```