Devoir surveillé

12 Novembre 2021 — Durée 1h15

Document autorisé : **Mémento C** vierge de toute annotation

Suite aux mobilisations pour le climat, le gouvernement français s'est vu obligé d'attribuer une aide aux ménages pour améliorer l'isolation thermique (rénovation énergétique) de leur logement. Une application a été développée pour permettre aux ménages de savoir s'ils peuvent bénéficier de l'aide, en fonction de la taille de leur logement, du nombre d'enfants et du revenu mensuel du ménage. Les données fournies par les utilisateurs sont stockées dans des fichiers, et nous proposons d'étudier une partie de l'application de traitement de ces données.

Le programme principal suivant lit une liste de revenus mensuels ordonnés de manière décroissante stockée dans un fichier `data_revenus.txt` puis lit un fichier `data_menage.txt` contenant les informations saisies par un ménage particulier, insère le revenu mensuel de ce ménage dans la liste des revenus, avant d'afficher la liste des revenus mise à jour :

`main.c` :

```c
#include <stdio.h>
#include <stdlib.h>
#include "traitement.h"
#include "es_tableau.h"
#include "menage.h"
#include "es_menage.h"

int main(int argc, char **argv) {

    tableau_entiers tab_analyse;
    lire_fichier_tableau(argv[1], &tab_analyse);

    menage usager;
    creer_menage(&usager);
    lire_fichier_menage(argv[2], &usager);

    /*** A completer dans l'exo 1 ***/
    /*********************************/
    return 0;
}
```

Un fichier `data_menage.txt` est un fichier composé :

1. d'un entier correspondant au revenu mensuel du ménage
2. d'un entier correspondant à la surface de l'habitation
3. du nombre `N` d'enfants
4. de `N` lignes comportant chacune un unique entier correspondant à l'âge d'un enfant

Exemple de fichier `data_menage.txt` :

```
1875
110
2
12
8
```

Un fichier `data_revenus.txt` est un fichier composé d'un nombre quelconque de lignes. À chaque ligne se trouve un entier correspondant au montant d'un mois de revenus.

Exemple de fichier `data_revenus.txt` :

```
6800
5980
5660
4230
3500
2000
1900
1850
1845
1842
1800
1450
```

Exercice 1. (4 pt)

1. Schématiser la structure globale et les relations entre les paquetages.
2. Écrire un Makefile permettant de compiler le programme `main`.

Exercice 2. (2 pt) On considère le paquetage **traitement**, qui fournit une fonction `insertion`. Cette fonction permet d'insérer un nouveau revenu entré par un utilisateur dans un tableau contenant l'ensemble des revenus triés par ordre décroissant. Voici son implémentation :

Exemple de fichier `traitement.c` :

```c
#include "traitement.h"

void insertion(tableau_entiers * t, int v){
    int i;

    for(i = 0; i < t->taille; i++){

      if(t->tab[i] <= v) {
        break;
      }
    }

    for(int j = t->taille; j >= i; j--){
      t->tab[j] = t->tab[j -1];
    }

    t->tab[i] = v;
    t->taille++;
}
```

Compléter le programme principal (**main.c**) pour :

1. insérer un revenu recueilli dans les données d'un.e usager.e dans le tableau de revenus trié ;
2. afficher le nouveau tableau.

Exercice 3. (4 pt) Décrire un jeu de tests fonctionnels permettant de tester la fonction `insertion`.

NB : il est demandé de *décrire* ce jeu de tests, et donc de *justifier* sa construction, sans nécessairement écrire explicitement les valeurs de chaque test.

Exercice 4. (4 pt) Nous souhaitons ajouter un module **oracle**, permettant de tester la fonction `insertion`.

1. Expliciter les propriétés que votre oracle devrait tester (sans détailler le code).
2. Décrire les fichiers ajoutés (sans détailler l'implémentation des fonctions) et donner les dépendances du module ajouté.

Exercice 5. (4 pt)

1. Écrire trois tests de robustesse permettant de mettre en évidence différentes erreurs de format d'entrée non gérées à la lecture d'une structure de type **menage**.
2. Quelles sont les pré-conditions de la fonction `insertion`? Modifiez cette fonction afin qu'elle renvoie une erreur si ces pré-conditions ne sont pas remplies.

Exercice 6. (2 pt) Pour rendre le programme plus générique, on se propose de modifier le type **menage** pour stocker les âges des enfants dans un tableau de taille 10 plutot que dans 10 variables différentes.

1. Modifier le type abstrait **menage** en conséquence.
2. D'après leur spécification, quelles seront les fonctions qui seront impactées par ce changement et devront être modifiées ?

Annexes

Paquetage `tableau` : contenu du fichier `tableau.h`

```
1  #ifndef _TABLEAU_H_
2  #define _TABLEAU_H_
3
4  #define TAILLE_MAX 10000
5
6  typedef struct {
7    int taille;
8    int tab[TAILLE_MAX];
9  } tableau_entiers;
10
11 #endif
```

Paquetage `es_tableau` : contenu du fichier `es_tableau.h`

```
1  #ifndef _ES_TABLEAU_H_
2  #define _ES_TABLEAU_H_
3  #include "tableau.h"
4
5  /* Lit un fichier contenant une liste triée de revenus puis les stocke
6     dans une structure de type tableau_entiers */
7  void lire_fichier_tableau(char* filename, tableau_entiers* t);
8
9  #endif
```

Paquetage `traitement` : contenu du fichier `traitement.h`

```
1  #ifndef _TRAITEMENT_H_
2  #define _TRAITEMENT_H_
3  #include "tableau.h"
4
5  /* Insere une valeur dans un tableau trié par ordre décroissant */
6  void insertion(tableau_entiers * t, int v);
7
8  #endif
```

Paquetage menage : contenu du fichier menage.h

```c
#ifndef _MENAGE_H_
#define _MENAGE_H_

typedef struct {
  int revenu;
  int surface_logement;
  int nb_enfants;
  int age_enfant1;
  int age_enfant2;
  int age_enfant3;
  int age_enfant4;
  int age_enfant5;
  int age_enfant6;
  int age_enfant7;
  int age_enfant8;
  int age_enfant9;
  int age_enfant10;
} menage;

// Initialise la structure menage
void creer_menage(menage* m);

// Renvoie le revenu mensuel du ménage
int revenu_menage(menage* m);

// Renvoie la surface du logement du ménage
int surface_logement(menage* m);

// Renvoie 1 si le ménage a au moins un enfant, 0 sinon
int a_enfant(menage* m);

// Renvoie le nombre d'enfants d'un menage
int nombre_enfant(menage* m);

/* Renvoie 1 si le ménage est éligible, 0 sinon. Le ménage m est
   éligible si le revenu mensuel moins 200 euros fois le nombre
   d'enfants est inférieur à1500 euros et que la surface est
   inférieure à100 m^2 plus 40 m^2 fois le nombre d'enfants, 0
   sinon (dans ce cas le ménage a les moyens de financer la
   rénovation, ou la surface est trop grande et une premiè re étape est
   de réduire la surface de son logement) */
int est_eligible_prime(menage* m);

#endif
```

Paquetage es_menage : contenu du fichier es_menage.h

```c
#ifndef _ES_MENAGE_H_
#define _ES_MENAGE_H_
#include "menage.h"

/* Lit un fichier contenant les informations sur un
   menage puis les stocke dans une structure de type
   menage */
void lire_fichier_menage(char* nom_fichier, menage* m);

#endif
```