Compte Rendu TP : Carnet de Notes - Entrées / Sorties

Etudiant : Assane Thiao

Formation : Télécommunications et Réseaux

Niveau : ING1

Année : 2024/2025

1. Objectif

L'objectif de ce TP est de manipuler des structures de stockage permettant de représenter le bulletin scolaire d'un étudiant en utilisant le langage C. Nous devons implémenter des fonctionnalités permettant de sauvegarder et de lire les données contenues dans un fichier texte au format CSV.

2. Structures de Données

Nous avons défini les structures suivantes :

Structure `COURSE`

Elle représente une matière et comprend :

- `title` : le nom de la matière (chaîne de caractères de taille fixe 80)
- `score` : la note obtenue (nombre réel double précision)

Structure `FICHE`

Elle représente une fiche étudiant et comprend :

- `name` : le nom de l'étudiant (chaîne de caractères de taille fixe 80)
- `surname` : le prénom de l'étudiant (chaîne de caractères de taille fixe 80)
- `id` : le numéro étudiant (entier)
- `courses` : un tableau de `COURSE` contenant les notes des matières (défini par une macro `MAX_L_COURS` à 3)

3. Fonctionnalités Implémentées

3.1 Affichage d'une fiche (`printFiche`)

La fonction `printFiche(FICHE fiche)` affiche le contenu d'une fiche de manière lisible sur la console.

3.2 Sauvegarde d'une fiche (`saveFiche`)

La fonction `saveFiche(FILE *file, FICHE fiche)` sauvegarde une fiche dans un fichier au format CSV

3.3 Chargement d'une fiche (`loadFiche`)

La fonction `loadFiche(FICHE *fiche, char *str)` convertit une ligne CSV en structure `FICHE` en utilisant `strtok_s` pour séparer les champs.

3.4 Lecture d'un fichier contenant plusieurs fiches

Nous avons modifié exo5 pour lire plusieurs lignes de `database.csv` et afficher chaque fiche successivement.

```c
/* Assane Thiao 28/02/2025 */
#include <stdio.h>  /* Pour les entrées/sorties */
#include <stdlib.h> /* Pour les conversions et allocations mémoire */
#include <string.h> /* Pour la manipulation des chaînes de caractères */

/* Définition des constantes */
#define MAX_L_CARAC 80  /* Taille max des noms et prénoms */
#define MAX_L_COURS 3   /* Nombre de matières */

/* Structure représentant une matière avec son nom et sa note */
typedef struct {
    char title[MAX_L_CARAC];
    double score;
} COURSE;

/* Structure représentant une fiche d'étudiant avec nom, prénom, ID et notes */
typedef struct {
    char name[MAX_L_CARAC];
    char surname[MAX_L_CARAC];
    int id;
    COURSE courses[MAX_L_COURS];
} FICHE;

/* Déclaration des fonctions */
void printFiche(FICHE fiche);
void saveFiche(FILE* file, FICHE fiche);
void loadFiche(FICHE* fiche, const char* str);

/* Fonction principale */
int main() {
    /* Initialisation d'une fiche étudiant */
    FICHE etudiant1 = { "Doe", "John", 11300055, {{"Maths", 12.0}, {"Info", 8.5}, {"Physics", 10.0}} };
    printFiche(etudiant1); /* Affichage de la fiche */
    printf("\n");
    /* Téléchargement d'une fiche depuis une chaîne */
    const char* etu = "Thiao;Assane;12410656;Maths;16.71;Info;19.03;Physics;13.23";

    /* Allocation dynamique pour éviter le pointeur NULL */
    FICHE* file1 = (FICHE*)malloc(sizeof(FICHE));
    if (file1 == NULL) {
        printf("Erreur d'allocation mémoire.\n");
        return 1;
    }

    loadFiche(file1, etu);
    printFiche(*file1);

    /* Sauvegarde des fiches dans un fichier */
    FILE* file;
    if (fopen_s(&file, "database.csv", "w") == 0) {
        saveFiche(file, etudiant1);
        fclose(file);
    }
    else {
        printf("Erreur d'ouverture du fichier\n");
```

```c
    }
    else {
        printf("Erreur d'ouverture du fichier\n");
    }

    /* Libération de la mémoire allouée */
    free(file1);

    return 0;
}

/* Fonction pour afficher une fiche étudiant */
void printFiche(FICHE fiche) {
    printf("Nom: %s\n", fiche.name);
    printf("Prénom: %s\n", fiche.surname);
    printf("Numéro étudiant: %d\n", fiche.id);
    for (int i = 0; i < MAX_L_COURS; i++) {
        printf("%s: %.2f\n", fiche.courses[i].title, fiche.courses[i].score);
    }
}

/* Fonction pour sauvegarder une fiche dans un fichier CSV */
void saveFiche(FILE* file, FICHE fiche) {
    if (file == NULL) return; // Vérification avant d'écrire dans le fichier
    fprintf(file, "%s;%s;%d", fiche.name, fiche.surname, fiche.id);
    for (int i = 0; i < MAX_L_COURS; i++) {
        fprintf(file, ";%s;%.2f", fiche.courses[i].title, fiche.courses[i].score);
    }
    fprintf(file, "\n");
}

/* Fonction pour charger une fiche à partir d'une chaîne formatée CSV */
void loadFiche(FICHE* fiche, const char* str) {
    if (fiche == NULL || str == NULL) return; // Vérification des pointeurs

    char buffer[256];
    strcpy_s(buffer, sizeof(buffer), str);  // Copie sécurisée de la chaîne pour strtok_s

    char* nextValeurRecupere = NULL;
    char* valeurRecupere = strtok_s(buffer, ";", &nextValeurRecupere);
    strcpy_s(fiche->name, MAX_L_CARAC, valeurRecupere);

    valeurRecupere = strtok_s(NULL, ";", &nextValeurRecupere);
    strcpy_s(fiche->surname, MAX_L_CARAC, valeurRecupere);

    valeurRecupere = strtok_s(NULL, ";", &nextValeurRecupere);
    fiche->id = atoi(valeurRecupere);

    /* Extraction des matières et notes */
    for (int i = 0; i < MAX_L_COURS; i++) {
        valeurRecupere = strtok_s(NULL, ";", &nextValeurRecupere);
        strcpy_s(fiche->courses[i].title, MAX_L_CARAC, valeurRecupere);

        valeurRecupere = strtok_s(NULL, ";", &nextValeurRecupere);
```

```c
/* Fonction pour sauvegarder une fiche dans un fichier CSV */
void saveFiche(FILE* file, FICHE fiche) {
    if (file == NULL) return; // Vérification avant d'écrire dans le fichier
    fprintf(file, "%s;%s;%d", fiche.name, fiche.surname, fiche.id);
    for (int i = 0; i < MAX_L_COURS; i++) {
        fprintf(file, ";%s;%.2f", fiche.courses[i].title, fiche.courses[i].score);
    }
    fprintf(file, "\n");
}

/* Fonction pour charger une fiche à partir d'une chaîne formatée CSV */
void loadFiche(FICHE* fiche, const char* str) {
    if (fiche == NULL || str == NULL) return; // Vérification des pointeurs

    char buffer[256];
    strcpy_s(buffer, sizeof(buffer), str);  // Copie sécurisée de la chaîne pour strtok_s

    char* nextValeurRecupere = NULL;
    char* valeurRecupere = strtok_s(buffer, ";", &nextValeurRecupere);
    strcpy_s(fiche->name, MAX_L_CARAC, valeurRecupere);

    valeurRecupere = strtok_s(NULL, ";", &nextValeurRecupere);
    strcpy_s(fiche->surname, MAX_L_CARAC, valeurRecupere);

    valeurRecupere = strtok_s(NULL, ";", &nextValeurRecupere);
    fiche->id = atoi(valeurRecupere);

    /* Extraction des matières et notes */
    for (int i = 0; i < MAX_L_COURS; i++) {
        valeurRecupere = strtok_s(NULL, ";", &nextValeurRecupere);
        strcpy_s(fiche->courses[i].title, MAX_L_CARAC, valeurRecupere);

        valeurRecupere = strtok_s(NULL, ";", &nextValeurRecupere);
        fiche->courses[i].score = atof(valeurRecupere);
    }
}
```

Test

```
Console de débogage Microsoft Visual Studio
Nom: Doe
PrÚnom: John
NumÚro Útudiant: 11300055
Maths: 12.00
Info: 8.50
Physics: 10.00

Nom: Thiao
PrÚnom: Assane
NumÚro Útudiant: 12410656
Maths: 16.71
Info: 19.03
Physics: 13.23

Sortie de C:\Users\e12410656\source\repos\partieTest\Debug\partie2.exe (processus 21860). Code : 0.
Pour fermer automatiquement la console quand le débogage s'arrête, activez Outils->Options->Débogage->Fermer automatique
ment la console à l'arrêt du débogage.
Appuyez sur une touche pour fermer cette fenêtre. . .
```

4. Résultats et Observations

- Les structures et macros facilitent la manipulation des données.
- `strtok_s` permet une lecture sécurisée des champs.
- L'allocation dynamique a permis de gérer des chaînes de taille variable.

- L'utilisation des fichiers CSV rend les données lisibles et facilement modifiables.

5. Améliorations Possibles

1. **Meilleure présentation de l'affichage** : Améliorer `printFiche` en utilisant des tabulations et un formatage avancé (`printf`).
2. **Lecture de fichiers de grande taille** : Gérer des chaînes de longueur arbitraire via l'allocation dynamique.
3. **Ajout de nouvelles matières** : Modifier `MAX_L_COURS` pour gérer un nombre de matières plus flexible.

6. Conclusion

Ce TP nous a permis de renforcer nos compétences en manipulation de fichiers, en gestion de structures et en manipulation de chaînes de caractères en C. Nous avons également appris à implémenter un format d'échange de données simple et efficace via le CSV.