

Rapport sur le mini projet

Gestion des étudiants

Créé par **AZAOUY** et **Bouhdyd** sous la supervision du Professeur

Prof **R.HANNANE**.

Les Axes du rapport :

- **Introduction générale**
- **Fichiers sources**
 - La fonction d'Ajout
 - La fonction de Modification
 - La fonction de consultation
 - La fonction de trier
 - La fonction de recherche
 - La fonction de suppression
 - [La fonction d'utile](#)
- **Fichiers header**
 - Ajouter.h
 - Modifier.h
 - Consulter.h
 - Trier.h
 - Rechercher.h
 - Supprimer.h
 - [Utile.h](#)
- **Programme Principale**
- **Conclusion**

Introduction générale

L'objectif de ce projet est d'organiser une application de gestion des informations des étudiants en utilisant le langage C et en concentrant sur la programmation modulaire. Dans cette application, nous avons organisé les fichiers en deux catégories principales : les fichiers header et les fichiers sources.

Les fichiers header jouent un rôle très important dans la structuration du code. Ils contiennent les déclarations des fonctions utilisées dans le programme, ce qui permet de séparer la déclaration de l'implémentation. Cela rend le code plus modulaire, plus facile à comprendre et à maintenir. Ces fichiers header sont inclus dans le programme principal, généralement nommé main.c, où toutes les fonctionnalités de l'application sont appelées et utilisées.

Quant aux fichiers sources, ils contiennent l'implémentation réelle des fonctions déclarées dans les fichiers header. C'est là que nous définissons le comportement des différentes tâches de l'application, telles que l'ajout, la suppression ou la modification des informations des étudiants.

Dans ce rapport, nous allons examiner de manière détaillée chaque fichier du projet. Nous commencerons par explorer les fichiers sources, où nous trouverons les instructions concrètes pour chaque tâche de l'application. Ensuite, nous passerons en revue les fichiers header, qui définissent les signatures de fonction et les structures de données utilisées dans le projet. Enfin, nous terminerons par une analyse du programme principal et de ses fonctionnalités, mettant en lumière la manière dont il orchestre l'ensemble de l'application.

Fichiers sources

La fonction d'ajout.

Code source :

```
#include "Ajoute.h"
#include "Utile.h"
#include <stdio.h>
#include <stdlib.h>

void ajoutEtudiants() {
    short nbretudiants = 0;
    Etudiant *tabEtudiants = (Etudiant *)malloc(sizeof(Etudiant) * nbretudiants);
    if (tabEtudiants == NULL) {
        printf("Désolé, il y a un problème, essayez plus tard\n");
        exit(EXIT_FAILURE);
    }
    printf("-----\n");
```

```

printf(ANSI_COLOR_BLUE"Combien des etudiant vous voulez ajouter ?\n"ANSI_COLOR_RESET);
printf("-----\n");
printf(": ");
scanf("%hd", &nbretudiants);
clear();

tabEtudiants = realloc(tabEtudiants, nbretudiants * sizeof(Etudiant));
if (tabEtudiants == NULL) {
    printf(ANSI_COLOR_RED"Désolé, il y a un problème, essayez plus tard\n"ANSI_COLOR_RESET);
    exit(EXIT_FAILURE);
}

for (short i = 0; i < nbretudiants; ++i) {
    printf("-----\n");
    printf(ANSI_COLOR_BLUE"Etudiant [%hd]\n"ANSI_COLOR_RESET, i + 1);
    printf("-----\n");
    while (getchar() != '\n');
    printf("Prénom : ");
    fgets(tabEtudiants[i].prenom, sizeof(tabEtudiants[i].prenom), stdin);
    deletenewline(tabEtudiants[i].prenom);
    printf("---\n");
    printf("Nom : ");
    fgets(tabEtudiants[i].nom, sizeof(tabEtudiants[i].nom), stdin);
    deletenewline(tabEtudiants[i].nom);
    printf("---\n");
    printf("Filière :");
    fgets(tabEtudiants[i].filiere.nom, sizeof(tabEtudiants[i].filiere.nom), stdin);
    deletenewline(tabEtudiants[i].filiere.nom);
    printf("---\n");

    do {
        printf("Date d'inscription (jj/mm/aaaa): ");
        fgets(tabEtudiants[i].dateInscri, sizeof(tabEtudiants[i].dateInscri), stdin);
        deletenewline(tabEtudiants[i].dateInscri);
    } while (dateEstValid(tabEtudiants[i].dateInscri) == '0');

    printf("---\n");
    do {
        printf("Numéro apogée:");
        scanf("%d", &tabEtudiants[i].apogee);
    } while (apogeeEstValid(tabEtudiants[i].apogee) == '0');

    printf("---\n");
    printf(ANSI_COLOR_BLUE"Saisi les notes des modules :\n"ANSI_COLOR_RESET);
    printf("-----\n");
    float somme = 0;

    for (short j = 0; j < 6; ++j) {

```

```

while (getchar() != '\n');
printf("Saisi le nom de module %hd: ", j + 1);
fgets(tabEtudiants[i].filiere.tabModules[j].nom, 20, stdin);
printf("---\n");
printf("Saisi la note : ");
scanf("%f", &((tabEtudiants + i)->filiere.tabModules[j].note));

while ((tabEtudiants + i)->filiere.tabModules[j].note < 0 || (tabEtudiants + i)-
>filiere.tabModules[j].note > 20) {
    printf(ANSI_COLOR_RED"La note doit être entre 0 et 20\n"ANSI_COLOR_RESET);
    printf("Saisi la note : ");
    scanf("%f", &((tabEtudiants + i)->filiere.tabModules[j].note));
}

printf("---\n");
somme += (tabEtudiants + i)->filiere.tabModules[j].note;
}

(tabEtudiants + i)->moyenne = somme / 6;
clear();
printf(ANSI_COLOR_GREEN"Les informations ont été ajoutées avec succès.\n"ANSI_COLOR_RESET);
}

ajoutDansFichier(tabEtudiants, nbretudiants);
free(tabEtudiants);
}

void ajoutDansFichier(Etudiant tabEtudiants[], short nbretudiant) {
    FILE *file = NULL;
    file = fopen("./data/Etudiants.bin", "ab");
    if (file != NULL) {
        for (int i = 0; i < nbretudiant; ++i) {
            fwrite(tabEtudiants + i, sizeof(Etudiant), 1, file);
        }
        fclose(file);
    } else {
        ERROR;
    }
}

```

Explication

Ce code permet d'ajouter des étudiants et leurs informations dans une base de données. Il commence par demander à l'utilisateur combien d'étudiants il souhaite ajouter, puis alloue dynamiquement de l'espace mémoire en fonction de ce nombre. Ensuite, il demande les détails de chaque étudiant, tels que le prénom, le nom, la filière, la date d'inscription, le numéro apogée et les notes des modules. Les informations saisies sont ensuite stockées dans un fichier binaire "Etudiants.bin" dans le répertoire "./data". Enfin, l'espace mémoire alloué est libéré.

Résultat

Lors de l'exécution de la fonction d'ajout on rencontre cette question et il faut la répondre pour continuer

```
-----  
Combien des etudiant vous voulez ajouter ?  
-----  
: 1
```

Et après en ajoutant nombre des étudiants qu'on a saisi au début, chaque étudiant a des informations précises

```
-----  
Etudiant [1]  
-----  
Prénom : Youssef  
---  
Nom : Bouhdyd  
---  
Filière : smi  
---  
Date d'inscription (jj/mm/aaaa): 09/09/2022  
---  
Numéro apogée: 2225901  
---
```

et ensuite on remplit les notes

```
Saisi les notes des modules :  
-----  
Saisi le nom de module 1: math  
---  
Saisi la note : 10  
---  
Saisi le nom de module 2: programing  
---  
Saisi la note : 15  
---  
Saisi le nom de module 3: web  
---  
Saisi la note : 14  
---  
Saisi le nom de module 4: elctronique  
---  
Saisi la note : 18  
---  
Saisi le nom de module 5: chime  
---  
Saisi la note : 12  
---  
Saisi le nom de module 6: biologie  
---  
Saisi la note : 13
```

La fin de la saisie (l'ajout des étudiants est réussi)

```
Les informations ont été ajoutées avec succès.  
-----
```

La fonction de modification

Code source :

```
#include "Modifier.h"  
#include "Utile.h"  
#include <stdio.h>  
#include <stdlib.h>  
  
void modifierEtudiant() {  
    // Declaration  
    short exist = 0; // verifier si l'appoge saisi correspond a un etudiant  
    int code; // le numero d'apogee saisi par l'utilisateur  
    int i, j; // les dimension  
    int choix; // la une suite des nombre reel
```

```

printf(ANSI_COLOR_BLUE"Veuillez saisir le numero d'apogee de l'etudiant a modifier:
"ANSI_COLOR_RESET);
scanf("%d", &code);
printf("\n");
FILE *modifeidfile = NULL;
modifeidfile = fopen("./data/modifiedfile.bin", "wb");
if (modifeidfile != NULL) {
    FILE *file = NULL;
    file = fopen("./data/Etudiants.bin", "rb");
    if (file != NULL) {
        Etudiant etudiant;
        while (fread(&etudiant, sizeof(Etudiant), 1, file)) {
            if (etudiant.apogee == code) {
                exist = 1;
                printf("Voici les informations de l'etudiant avant modification:\n");
                afficherEtudiant(etudiant);
                printf("\nQue voulez-vous modifier?\n");
                printf("-----\n");
                printf("1 | Nom\n");
                printf("2 | Prenom\n");
                printf("3 | Date d'inscription\n");
                printf("4 | Numero d'Apogee\n");
                printf("5 | Note d'un module\n");
                printf(ANSI_COLOR_RED"0 | Quitter\n"ANSI_COLOR_RESET);
                printf("-----\n");
                printf("Choix: ");
                scanf("%d", &choix);

                switch (choix) {
                    case 1:
                        printf("Nouveau Nom: ");
                        scanf("%s", etudiant.nom);
                        break;
                    case 2:
                        printf("Nouveau Prenom: ");
                        scanf("%s", etudiant.prenom);
                        break;
                    case 3:
                        while (getchar() != '\n');
                        do {
                            printf("Date d'inscription (jj/mm/aaaa): ");
                            fgets(etudiant.dateInscri, sizeof(etudiant.dateInscri), stdin);
                            deletnewline(etudiant.dateInscri);
                        } while (dateEstValid(etudiant.dateInscri) == '0');
                        break;
                    case 4:
                        do {
                            printf("Nouveau Numero d'Apogee: ");

```



```

        scanf("%d", &etudiant.apogee);
    } while (apogeeEstValid(etudiant.apogee));
    break;
case 5:
    do {
        printf("Veuillez saisir le numero du module a modifier (1-6): ");
        scanf("%d", &j);
    } while (j < 1 || j > 6);
    printf("Nouvelle Note du module %d: ", j);
    scanf("%f", &etudiant.filiere.tabModules[j - 1].note);
    while (etudiant.filiere.tabModules[j - 1].note < 0 || etudiant.filiere.tabModules[j - 1].note
> 20){

        printf(ANSI_COLOR_RED"La note doit entre enter 0 et 20 \n"ANSI_COLOR_RESET);
        printf("Nouvelle Note du module %d: ", j);
        scanf("%f", &etudiant.filiere.tabModules[j - 1].note);
    }
    // Recalcul de la moyenne de la filiere
    float somme = 0;
    for (int k = 0; k < 6; k++) {
        somme += etudiant.filiere.tabModules[k].note; // calculer la somme des note
    }
    etudiant.moyenne = somme / 6; // la mise a jour de la moyenne
    printf("-----\n");
    printf(ANSI_COLOR_GREEN"La moyenne de l'etudiant a ete mise a jour automatiquement.\n"ANSI_COLOR_RESET);
    break;
case 0:
    fwrite(&etudiant, sizeof(Etudiant), 1, modifeidfile);
    clear();
    break;
default:
    printf(ANSI_COLOR_RED"Choix incorrect, veuillez ressayer.\n"ANSI_COLOR_RESET);
}
if (choix >= 1 && choix <= 5){
    printf(ANSI_COLOR_GREEN"L'information d'etudiatat apret la modification \n"ANSI_COLOR_RESET);
    afficherEtudiant(etudiant);
    fwrite(&etudiant, sizeof(Etudiant), 1, modifeidfile);
}
}
else {
    fwrite(&etudiant, sizeof(Etudiant), 1, modifeidfile);
}
}
// Si l'etudiant n'existe pas
fclose(file);
}
else {

```

```

    printf(ANSI_COLOR_RED"Erreur lors de l'ouverture du fichier Etudiants.bin."ANSI_COLOR_RESET);
    printf("\n");
    modifierEtudiant();
}
fclose(modifiedfile);
}
else {
    printf(ANSI_COLOR_RED"Erreur lors de l'ouverture du fichier modifiedfile.bin.\n"ANSI_COLOR_RESET);
}
if (exist == 0) {
    printf(ANSI_COLOR_RED"L'etudiant avec le numero d'Apogee %d n'existe pas dans la liste.\n"ANSI_COLOR_RED, code);
}
else {
    #ifdef _WIN32
        system("copy modifiedfile.bin Etudiants.bin");
    #else
        system("cp ./data/modifiedfile.bin ./data/Etudiants.bin");
    #endif
}
}
}

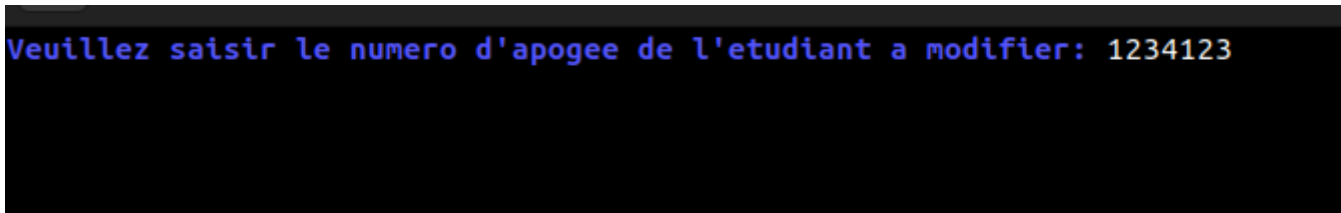
```

Explication

Ce code permet de modifier les informations d'un étudiant existant dans une base de données. Tout d'abord, il demande à l'utilisateur de saisir le numéro apogée de l'étudiant à modifier. Ensuite, il vérifie si l'étudiant existe dans la base de données. Si oui, il affiche les informations de l'étudiant et propose à l'utilisateur de choisir ce qu'il souhaite modifier : nom, prénom, date d'inscription, numéro d'apogée ou note d'un module. Après la modification, il met à jour la moyenne de l'étudiant si une note de module a été modifiée. Les modifications sont enregistrées dans un fichier temporaire "modifiedfile.bin", puis copiées dans le fichier principal "Etudiants.bin". Si l'étudiant n'existe pas, un message d'erreur est affiché.

Résultat

Le premier message qu'on va le rencontrer lors de l'exécution de la fonction de modification, c'est d'avoir si cet étudiant qu'on va le modifier existe dans la liste



```

Veuillez saisir le numero d'apogee de l'etudiant a modifier: 1234123

```

Il faut être sûr que l'étudiant existe dans la liste, et après en cliquant sur vous pouvez cliquer sur n'importe quel numéro existant dans le menu de modification selon vos choix

Voici les informations de l'étudiant avant modification:

Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
Bouhdyd	Youssef	smi	1234123	12/09/2022	16.67

Que voulez-vous modifier?

- 1| Nom
- 2| Prenom
- 3| Date d'inscription
- 4| Numero d'Apogee
- 5| Note d'un module
- 0| Quitter

Choix:

La modification est réussite

Que voulez-vous modifier?

- 1| Nom
- 2| Prenom
- 3| Date d'inscription
- 4| Numero d'Apogee
- 5| Note d'un module
- 0| Quitter

Choix: 1

Nouveau Nom: Bouhdyd

L'information d'etudianat apret la modification

Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
Bouhdyd	Youssef	smi	1234123	12/09/2022	16.67

La fonction de consultation

Code source

```
#include <stdio.h>
#include <string.h>
#include "Affiche.h"
#include "Utile.h"
void consulter(){
    short choix;
    printf(ANSI_COLOR_BLUE"Consultation \n"ANSI_COLOR_RESET);
    printf("-----\n");
    printf("1| La liste de tous les étudiants. \n");
    printf("2| La liste des étudiants admis seulement. \n");
    printf("3| La liste des étudiants d'une filière spécifiée. \n");
    printf(ANSI_COLOR_RED"0| Quitter. \n"ANSI_COLOR_RESET);
    printf("-----\n");
    printf("Choix : ");
    scanf("%hd",&choix);
```

```

switch (choix){
case 1:
    ConsulterTousEtud();
    consulter();
    break;
case 2:
    ConsulterEtudAdmis();
    consulter();
    break;
case 3:
    char filierecherch[10];
    printf("-----\n");
    // Effacer le tampon d'entrée
    while (getchar() != '\n');
    printf("Saisi la filiere : ");
    fgets(filierecherch,10,stdin);
    deletnewline(filierecherch);
    ConsulterEtudFiliere(filierecherch);
    consulter();
    break;
case 0:
    clear();
    break;
default:
    printf(ANSI_COLOR_RED"Votre choix il n'est exit pas dans la liste ! \n"ANSI_COLOR_RESET);
    consulter();
}
}

void ConsulterTousEtud(){
    FILE *file = NULL;
    file = fopen("./data/Etudiants.bin","rb");
    if (file != NULL){
        printf(ANSI_COLOR_BLUE"Liste de tous les etudiants:\n"ANSI_COLOR_RESET);
        Etudiant etudiant;
        while (fread(&etudiant,sizeof(Etudiant),1,file)) afficherEtudiant(etudiant);
        fclose(file);
    }
    else printf(ANSI_COLOR_RED"Les Listes sont vide \n"ANSI_COLOR_RESET);
}

void ConsulterEtudAdmis(){
    FILE *file = NULL;
    short nbradmi = 0 ;
    file = fopen("./data/Etudiants.bin","rb");
    printf("la liste des etudiants admis est:\n");
    if (file != NULL){
        printf(ANSI_COLOR_BLUE"Liste de tous les etudiants:\n"ANSI_COLOR_RESET);
        Etudiant etudiant;
        while (fread(&etudiant,sizeof(Etudiant),1,file)){

```

```

        if(etudiant.moyenne >= 10){
            nbradmi = 1;
            afficherEtudiant(etudiant);
            printf("\n");
        }
    }
    fclose(file);
    if (nbradmi == 0 ) printf(ANSI_COLOR_RED"Il n'y exit pas un etudiant admis ! \n"ANSI_COLOR_RESET);
    printf("\n");
}
else printf(ANSI_COLOR_RED"Les Listes sont vide \n"ANSI_COLOR_RESET);
}

void ConsulterEtudFiliere(char filiere[]){
    FILE *file = NULL;
    file = fopen("./data/Etudiants.bin","rb");
    if (file != NULL){
        printf(ANSI_COLOR_BLUE"la liste des etudiant de la filiere %s:\n"ANSI_COLOR_RESET, filiere);
        Etudiant etudiant;
        short exist = 0 ;
        while (fread(&etudiant,sizeof(Etudiant),1,file)){
            if(strcmp(etudiant.filiere.nom, filiere) == 0){
                exist = 1;
                afficherEtudiant(etudiant);
                printf("\n");
            }
        }
        if (exist == 0) {
            printf(ANSI_COLOR_RED"Cette filière n'exist pas !! \n"ANSI_COLOR_RESET);
            printf("\n");
        }
        fclose(file);
    }
    else printf(ANSI_COLOR_RED"Les Listes sont vide \n"ANSI_COLOR_RESET);
}

```

Explication

Dans ce code, les fichiers sont utilisés pour stocker les informations des étudiants sous forme binaire. Les données sont enregistrées dans un fichier binaire nommé "Etudiants.bin" dans le répertoire "data". Lorsque l'utilisateur choisit de consulter les informations des étudiants, le programme ouvre ce fichier et lit les données stockées à l'intérieur. Les informations des étudiants sont ensuite affichées à l'utilisateur selon l'option sélectionnée. Si le fichier n'existe pas ou s'il est vide, le programme informe l'utilisateur que les listes sont vides.

La gestion des fichiers est effectuée à l'aide des fonctions d'entrée/sortie de fichier de la bibliothèque standard de C, telles que fopen, fclose et fread. Ces fonctions permettent au programme d'ouvrir, de lire et de fermer les fichiers de manière appropriée, garantissant ainsi un accès sécurisé aux données stockées.

Résultat

Consulter les étudiant selon trois option

Par exemple Le premier c'est tous les étudiant

Consultation

```

1| La liste de tous les étudiants.
2| La liste des étudiants admis seulement.
3| La liste des étudiants d'une filière spécifiée.
0| Quitter.

```

Choix : 1

Liste de tous les étudiants:

Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
1	1	1	4444444	12/12/2009	9.00
Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
azoui	hassan	smi	5555555	09/09/2011	19.00
Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
1	1	1	6666666	12/12/1920	1.00
Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
Bouhdyd	Youssef	smi	1234123	12/09/2022	16.67

Le deuxième est la liste des étudiants admis seulement

la liste des étudiants admis est:

Liste de tous les étudiants:

Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
azoui	hassan	smi	5555555	09/09/2011	19.00
Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
Bouhdyd	Youssef	smi	1234123	12/09/2022	16.67

La fonction de tri

Code source

```

#include "Trie.h"
#include "Utile.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void trie(){
    short choix;
    printf(ANSI_COLOR_BLUE"Veuillez choisir le cas de tri:"ANSI_COLOR_RESET);
    printf("\n");
    printf("-----\n");

```

```

printf("1 | Numero d'apogee.\n");
printf("2 | Moyenne.\n");
printf("3 | Date d'inscription.\n");
printf(ANSI_COLOR_RED"0 | Quitter. \n"ANSI_COLOR_RESET);
printf("-----\n");
printf("Choix :");
scanf("%hd", &choix);
printf("-----\n");
FILE *file = NULL;
file = fopen("./data/Etudiants.bin", "rb");
if (file != NULL){
    // calculer le nombre d'etudiants dans fichier
    fseek(file, 0, SEEK_END);
    short nbretudiants = ftell(file) / sizeof(Etudiant);
    // remplir le tableau par l'etudiants
    Etudiant tabEtudiants[nbretudiants];
    // retourner la cursur a la debut;
    rewind(file);
    short i = 0;
    while (i < nbretudiants && fread(tabEtudiants + i, sizeof(Etudiant), 1, file)) i++;
    fclose(file);
    switch (choix){
        case 1:
            TriApogee(tabEtudiants, nbretudiants);
            trie();
            break;
        case 2:
            TriMoyenne(tabEtudiants, nbretudiants);
            trie();
            break;
        case 3:
            TriDateInscription(tabEtudiants, nbretudiants);
            trie();
            break;
        case 0:
            clear();
            break;
        default:
            printf(ANSI_COLOR_RED"votre choix est incorrect.\n"ANSI_COLOR_RESET);
            trie();
            break;
    }
}
else {ERROR;}
}

void TriApogee(Etudiant tabEtudiants[], short nbretudiant){
    //declaration
    int i, j; // des indice pour parcourir le tableau de structure

```

```

Etudiant AIDE; // pour nous aider a la permutation
for(i = 0; i < nbretudiant - 1; i++){
    for(j = 0; j < nbretudiant - i - 1; j++){
        if(tabEtudiants[j].apogee > tabEtudiants[j + 1].apogee){
            AIDE = tabEtudiants[j];
            tabEtudiants[j] = tabEtudiants[j + 1];
            tabEtudiants[j + 1] = AIDE;
        }
    }
}
printf(ANSI_COLOR_BLUE"La liste des etudiant trie par numero d'apogee:\n"ANSI_COLOR_RESET);
for(int i = 0; i < nbretudiant; i++){
    afficherEtudiant(tabEtudiants[i]);
    printf("\n");
}
}

void TriMoyenne(Etudiant tabEtudiants[], short nbretudiant){
    int i, j; // des indice pour parcourir le tableau de structure
    Etudiant AIDE; // pour nous aider a la permutation

    for(i = 0; i < nbretudiant - 1; i++){
        for(j = 0; j < nbretudiant - i - 1; j++){
            if(tabEtudiants[j].moyenne > tabEtudiants[j + 1].moyenne){
                AIDE = tabEtudiants[j];
                tabEtudiants[j] = tabEtudiants[j + 1];
                tabEtudiants[j + 1] = AIDE;
            }
        }
    }
    printf(ANSI_COLOR_BLUE"La liste des etudiants trie par la moyenne:\n"ANSI_COLOR_RESET);
    for(int i = 0; i < nbretudiant; i++){
        afficherEtudiant(tabEtudiants[i]);
        printf("\n");
    }
}

void TriDateInscription(Etudiant tabEtudiants[], short nbretudiant){
    int i, j; // des indice pour parcourir le tableau de structure
    Etudiant AIDE; // pour nous aider a la permutation
    for(i = 0; i < nbretudiant - 1; i++){
        for(j = 0; j < nbretudiant - i - 1; j++){
            if(strcmp(nouvDateFromat(tabEtudiants[j].dateInscri), nouvDateFromat(tabEtudiants[j +
1].dateInscri)) > 0){
                AIDE = tabEtudiants[j];
                tabEtudiants[j] = tabEtudiants[j + 1];
                tabEtudiants[j + 1] = AIDE;
            }
        }
    }
}

```



```

printf(ANSI_COLOR_BLUE"La liste des etudiant trie par la date d'inscription:\n"ANSI_COLOR_RESET);
for(int i = 0; i < nbretudiant; i++){
    afficherEtudiant(tabEtudiants[i]);
    printf("\n");
}
}
char* nouvDateFromat(char date[]){
    int jour,mois,annee;
    sscanf(date,"%d/%d/%d",&jour,&mois,&annee);
    char* nouveauForma = (char*)malloc(11);
    sprintf(nouveauForma, "%d/%02d/%02d", annee, mois, jour);
    return nouveauForma;
}

```

Explication

Ce programme est conçu pour trier les données des étudiants stockées dans un fichier en fonction de différents critères : numéro d'apogée, moyenne ou date d'inscription.

Tout d'abord, l'utilisateur choisit le critère de tri parmi les options disponibles. Ensuite, le programme ouvre le fichier "Etudiants.bin" pour lire les données des étudiants.

Il calcule le nombre d'étudiants dans le fichier et stocke ces données dans un tableau de structures d'étudiants.

En fonction du choix de l'utilisateur, le programme appelle l'une des trois fonctions de tri : TriApogee, TriMoyenne ou TriDateInscription. Chaque fonction de tri utilise l'algorithme de tri à bulles pour trier les données en fonction du critère choisi.

Après le tri, le programme affiche la liste des étudiants triés selon le critère sélectionné.

La fonction nouvDateFromat est utilisée pour reformater la date d'inscription dans un format plus convivial avant de la comparer dans la fonction TriDateInscription.

Enfin, le programme ferme le fichier une fois toutes les opérations terminées.

Résultat

Comme vous voyez dans cette illustration on a trois option de trie et nous avons choisi le numéro 1 pour trier selon le numéro d'apogée

```

Veuller choisir le cas de tri:
-----
1| Numero d'apogee.
2| Moyenne.
3| Date d'inscription.
0| Quitter.
-----
Choix :1
-----
La liste des etudiant trie par numero d'apogee:
-----

```

Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
Bouhdyd	Youssef	smi	1234123	12/09/2022	16.67

```

-----

```

Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
1	1	1	4444444	12/12/2009	9.00

```

-----

```

Nom	Prénom	Filière	Apogée	Date d'inscription	Moyenne
azoui	hassan	smi	5555555	09/09/2011	19.00

```

-----

```

La fonction de recherche

Code source

```
#include "Recherche.h"
#include "Utile.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void rechercherEtudiant(){
    short choix,exist = 0;
    printf(ANSI_COLOR_BLUE"Rcherche: \n"ANSI_COLOR_RESET);
    printf("-----\n");
    printf("1 | Rechercher par le nom. \n");
    printf("2 | Rechercher par le prénom. \n");
    printf("3 | Rechercher par le numéro apogée. \n");
    printf(ANSI_COLOR_RED"0 | Quitter.\n"ANSI_COLOR_RESET);
    printf("-----\n");
    printf("Veuillez choisir un numéro parmi cette liste : \n");
    printf("-----\n");
    printf("Choix: ");
    scanf("%hd",&choix);
    FILE *file = NULL;
    char nomrecherch[20];
    char prenomrecherch[20];
    int apogeerecherch;
    Etudiant etudiant;
    file = fopen("./data/Etudiants.bin","rb");
    if (file != NULL){
        switch (choix){
            case 1:
                // Effacer le tampon d'entrée
                while (getchar() != '\n');
                printf("Saisi le nom recherche : ");
                fgets(nomrecherch,20,stdin);
                deletenewline(nomrecherch);
                while (fread(&etudiant,sizeof(Etudiant),1,file)){
                    if (strncmp(etudiant.nom,nomrecherch,20) == 0){
                        afficherEtudiant(etudiant);
                        exist++;
                    }
                }
                printf("-----\n");
                exist != 0 ? printf(ANSI_COLOR_GREEN"Les resultates trouvee (%d)\n"ANSI_COLOR_RESET,exist):printf(ANSI_COLOR_RED"Les resultates trouvee (%d)\n"ANSI_COLOR_RESET,exist);
                rechercherEtudiant();
            
```

```

        break;
    case 2:
        while (getchar() != '\n');
        printf("Saisi le prénom recherche : ");
        fgets(prenomrecherch,20,stdin);
        deletenewline(prenomrecherch);
        while (fread(&etudiant,sizeof(Etudiant),1,file)){
            if (strncmp(etudiant.prenom,prenomrecherch,20) == 0){
                afficherEtudiant(etudiant);
                exist++;
            }
        }
        printf("-----\n");
        exist != 0 ? printf(ANSI_COLOR_GREEN"Les resultates trouvee (%d)\n"
n"ANSI_COLOR_RESET,exist):printf(ANSI_COLOR_RED"Les resultates trouvee (%d)\n"
n"ANSI_COLOR_RESET,exist);
        rechercherEtudiant();
        break;
    case 3:
        while (getchar() != '\n');
        printf("Saisi l'apogée recherche : ");
        scanf("%d",&apogeerecherch);
        while (fread(&etudiant,sizeof(Etudiant),1,file)){
            if (etudiant.apogee == apogeerecherch){
                afficherEtudiant(etudiant);
                exist++;
            }
        }
        exist != 0 ? printf(ANSI_COLOR_GREEN"Les resultates trouvee (%d)\n"
n"ANSI_COLOR_RESET,exist):printf(ANSI_COLOR_RED"Les resultates trouvee (%d)\n"
n"ANSI_COLOR_RESET,exist);
        printf("\n");
        rechercherEtudiant();
        break;
    case 0:
        clear();
        break;
    default:
        printf(ANSI_COLOR_RED"Votre choix n'est pas dans la liste \n"ANSI_COLOR_RESET);
        rechercherEtudiant();
        break;
}
fclose(file);
}
else {ERROR;}
}

```

Explication

Ce code a pour objectif de permettre à l'utilisateur de rechercher des informations sur les étudiants stockées dans un fichier binaire. Les fichiers utilisés sont "Recherche.h" et "Utile.h", qui contiennent des déclarations de fonctions utilisées dans le code.

Le fichier binaire "Etudiants.bin" est ouvert en mode lecture binaire ("rb"). En fonction du choix de l'utilisateur (recherche par nom, prénom ou numéro apogée), le programme lit les enregistrements du fichier à l'aide de la fonction fread().

Les fonctions rechercherEtudiant(), TriApogee(), TriMoyenne(), TriDateInscription() et nouvDateFromat() sont définies pour gérer respectivement la recherche d'étudiant, le tri par numéro apogée, le tri par moyenne, le tri par date d'inscription, et le formatage de la date. Après chaque opération de recherche, le fichier est refermé avec fclose(). Si le fichier ne peut pas être ouvert, une macro ERROR est utilisée pour gérer cette situation.

Résultat

La recherche selon le nom de l'étudiant

```
Rcherche:
-----
1| Rechercher par le nom.
2| Rechercher par le prénom.
3| Rechercher par le numéro apogée.
0| Quitter.
-----
Veuillez choisir un numéro parmi cette liste :
-----
Choix: 1
Saisi le nom recherche : Bouhdyd
-----
Nom          | Prénom      | Filière     | Apogée      | Date d'inscription | Moyenne
-----
Bouhdyd      | Youssef     | smi         | 1234123     | 12/09/2022        | 16.67
-----
Les resultates trouvee (1)
```

La recherche selon le numéro d'apogée

```
Rcherche:
-----
1| Rechercher par le nom.
2| Rechercher par le prénom.
3| Rechercher par le numéro apogée.
0| Quitter.
-----
Veuillez choisir un numéro parmi cette liste :
-----
Choix: 3
Saisi l'apogée recherche : 1234123
-----
Nom          | Prénom      | Filière     | Apogée      | Date d'inscription | Moyenne
-----
Bouhdyd      | Youssef     | smi         | 1234123     | 12/09/2022        | 16.67
-----
Les resultates trouvee (1)
```

La fonction de suppression

Code source

```
#include "Supprime.h"
```

```

#include "Utile.h"
#include <stdio.h>
#include <stdlib.h>
void supprimeEtudiant(){
    int deletedapogee;
    int exist = 0;
    printf("-----\n");
    printf(ANSI_COLOR_RED"Saisi l'apogée de l'étudiant qui voudrait supprimer \n"ANSI_COLOR_RESET);
    printf("-----\n");
    printf("Apogée: ");
    scanf("%d",&deletedapogee);
    FILE *modifeidfile = NULL;
    modifeidfile = fopen("./data/modifiedfile.bin", "wb");
    if (modifeidfile != NULL) {
        FILE *file = NULL;
        file = fopen("./data/Etudiants.bin", "rb");
        if (file != NULL) {
            Etudiant etudiant;
            while (fread(&etudiant, sizeof(Etudiant), 1, file)) {
                if (etudiant.apogee == deletedapogee) {
                    exist = 1 ;
                    continue;
                }
                else {
                    fwrite(&etudiant, sizeof(Etudiant), 1, modifeidfile);
                }
            }
            fclose(file);
        }
        else {
            printf(ANSI_COLOR_RED"Erreur lors de l'ouverture du fichier Etudiants.bin.\n"ANSI_COLOR_RESET);
        }
        fclose(modifeidfile);
    }
    else {
        printf(ANSI_COLOR_RED"Erreur lors de l'ouverture du fichier modifiedfile.bin.\n"ANSI_COLOR_RESET);
    }
    // verifier si l'etudiant exist
    if (exist == 0) {
        printf(ANSI_COLOR_RED"L'etudiant avec le numero d'Apogee %d n'existe pas dans la liste.\n"ANSI_COLOR_RESET, deletedapogee);
    }
    else {
        printf(ANSI_COLOR_GREEN"l'étudiant a été supprimé avec succès \n"ANSI_COLOR_RESET);
#ifdef _WIN32
        system("copy modifiedfile.bin Etudiants.bin");
#else
        system("cp ./data/modifiedfile.bin ./data/Etudiants.bin");
#endif
    }
}

```

```
#endif
}
}
```

Explication

Ce code a pour but de supprimer un étudiant de la liste en fonction de son numéro d'apogée saisi par l'utilisateur. Les fichiers utilisés sont "Supprime.h" et "Utile.h", qui contiennent des déclarations de fonctions utilisées dans le code.

L'utilisateur saisit le numéro d'apogée de l'étudiant à supprimer. Ensuite, le code ouvre les fichiers binaire "Etudiants.bin" en mode lecture et "modifiedfile.bin" en mode écriture binaire ("wb"). Il lit chaque enregistrement d'étudiant dans le fichier source. Si l'apogée de l'étudiant correspond à celui saisi par l'utilisateur, il est ignoré et le programme passe à l'enregistrement suivant. Sinon, l'enregistrement est écrit dans le fichier de destination.

Après le traitement de tous les enregistrements, le fichier source est refermé. Si le fichier source n'a pas pu être ouvert, un message d'erreur est affiché. Enfin, le fichier de destination est refermé. Si l'étudiant a été trouvé et supprimé avec succès, le fichier de destination est renommé pour écraser le fichier source, mettant ainsi à jour la liste des étudiants.

Résultat

L'étudiant qui avait le numéro d'apogée identique à ce qui était saisi sera supprimé dans la liste des étudiants directement

```
-----
Saisi l'apogée de l'étudiant qui voudrait supprimer
-----
Apogée: 4444444
l'étudiant a été supprimé avec succès
-----
```

Voilà le cas où il ne trouve pas aucun étudiant ayant le même apogée

```
Rcherche:
-----
1| Rechercher par le nom.
2| Rechercher par le prénom.
3| Rechercher par le numéro apogée.
0| Quitter.
-----
Veuillez choisir un numéro parmi cette liste :
-----
Choix: 3
Saisi l'apogée recherche : 4444444
Les resultats trouvee (0)
```

Utile.c

Code source

```
#include "Utile.h"
#include "head.h"
#include <stdio.h>
```

```

#include <stdlib.h>
void clear(){
    #ifdef _WIN32
        system("cls");
    #else
        system("clear");
    #endif
}
void deletnewline(char *string){
    for (char *p = string ; *p != '\0' ; p++){
        if (*p == '\n') *p = '\0' ;
    }
}
char dateEstValid(char *date){
    // define trois variable pour l'annee et mois,jour
    short j,m,a;
    if (sscanf(date,"%hd/%hd/%hd",&j,&m,&a) == 3){
        if (j >= 1 && j <= 31 &&
            m >= 1 && m <= 12 &&
            a >= 1900 && a <= 3000) {
            return '1';
        }
        else {
            printf(ANSI_COLOR_RED"La date saisie '%s' est invalide ou ne correspond pas au format
(jj/mm/aaaa)."ANSI_COLOR_RESET, date);
            printf("\n");
            return '0';
        }
    }
    else {
        printf(ANSI_COLOR_RED"La date saisie '%s' est invalide ou ne correspond pas au format
(jj/mm/aaaa)."ANSI_COLOR_RESET, date);
        printf("\n");
        return '0';
    }
}
char apogeeEstValid(int apogee){
    // verifier si l'apogee contient 7 nombres
    if ((int) (apogee / 1000000) >= 1 && (int) (apogee / 1000000) < 10 ){
        FILE *file = NULL ;
        file = fopen("./data/Etudiants.bin","rb+");
        Etudiant etudiant;
        if (file != NULL){
            while(fread(&etudiant,sizeof(Etudiant),1,file))
                if (etudiant.apogee == apogee) {
                    printf(ANSI_COLOR_RED"Cet apogee est deja exist ! \n"ANSI_COLOR_RESET);
                    return '0';
                }
        }
    }
}

```

```

        fclose(file);
    }
    else return '1';
}
else {
    printf( ANSI_COLOR_RED "L'apogee doit contient 7 chiffre \n" ANSI_COLOR_RESET);
    return '0';
}
return '1';
}

void afficherEtudiant(Etudiant etudiant){
    printf("-----\n");
    printf(ANSI_COLOR_YELLOW " Nom\t\t| Prénom\t| Filière\t| Apogée\t| Date d'inscription\t| Moyenne\t\n" ANSI_COLOR_RESET);
    printf("-----\n");
    printf(" %-10s\t| %-10s\t| %-10s\t| %-10d\t| %-20s\t| %.2f\t\n",
        etudiant.nom, etudiant.prenom, etudiant.filiere.nom, etudiant.apogee, etudiant.dateInscri,
        etudiant.moyenne);
    printf("-----\n");
}

```

Explication

Ce code fournit des fonctions utilitaires pour manipuler des données d'étudiants. Les fichiers utilisés sont "Utile.h" et "head.h". Les fonctions incluses sont :

1. clear() : Efface l'écran de la console.
2. deletnewline(char * string) : Supprime le caractère de nouvelle ligne (\n) à la fin d'une chaîne.
3. dateEstValid(char * date) : Vérifie si une date est valide au format jj/mm/aaaa.
4. apogeeEstValid(int apogee) : Vérifie la validité d'un numéro d'apogée.
5. afficherEtudiant(Etudiant etudiant) : Affiche les informations d'un étudiant.

Ces fonctions sont utilisées pour des opérations de gestion des étudiants, telles que la validation des données entrées et l'affichage des informations

Résultat

voici une validation d'entrer la date

```

Date d'inscription (jj/mm/aaaa): 12/30/2004
La date saisie '12/30/2004' est invalide ou ne correspond pas au format (jj/mm/aaaa).
Date d'inscription (jj/mm/aaaa): 40/02/2004
La date saisie '40/02/2004' est invalide ou ne correspond pas au format (jj/mm/aaaa).
Date d'inscription (jj/mm/aaaa): 09/03/2023

```

voici une validation d'apogée

```

Numéro apogée:1234123
Cet apogee est deja exist !
Numéro apogée:12341234
L'apogee doit contient 7 chiffre
Numéro apogée:123123
L'apogee doit contient 7 chiffre
Numéro apogée:2229342

```


Fichiers header

Ajout.h

Code source :

```
#ifndef AFFICHE_H
#define AFFICHE_H
#include "head.h"
#include "Utile.h"
void consulter();
void ConsulterTousEtud();
void ConsulterEtudAdmis();
void ConsulterEtudFiliere(char filiere[]);
#endif
```

Explication :

Ce code déclare un fichier d'en-tête "Ajoute.h" qui contient des déclarations de fonctions pour gérer l'ajout d'étudiants. Il inclut les fichiers d'en-tête "head.h" et "Utile.h". La fonction principale déclarée est ajoutEtudiants(), qui gère le processus d'ajout d'étudiants. De plus, il y a une fonction de manipulation de la base de données déclarée ajoutDansFichier(), qui prend un tableau d'étudiants et le nombre d'étudiants à ajouter, et les insère dans un fichier.

Modifier.h

Code source :

```
#ifndef MODIFIER_H
#define MODIFIER_H
#include "head.h"
void modifierEtudiant();
#endif
```

Explication :

Ce code déclare un fichier d'en-tête "Modifier.h" qui contient la déclaration d'une fonction modifierEtudiant(). Cette fonction gère la modification des données d'un étudiant. Le fichier inclut l'en-tête "head.h" pour les définitions nécessaires. L'instruction préprocesseur #ifndef garantit que le contenu de l'en-tête est inclus une seule fois dans un fichier source lors de la compilation.

Consulter.h

Code source:

```
#ifndef AFFICHE_H
#define AFFICHE_H
#include "head.h"
#include "Utile.h"
void consulter();
void ConsulterTousEtud();
void ConsulterEtudAdmis();
void ConsulterEtudFiliere(char filiere[]);
#endif
```

Explication :

Ce code définit un fichier d'en-tête "Affiche.h" qui contient les déclarations de trois fonctions : consulter(), ConsulterTousEtud(), ConsulterEtudAdmis(), et ConsulterEtudFiliere(). Ces fonctions sont utilisées pour consulter différentes informations sur les étudiants. Le fichier inclut les en-têtes "head.h" et "Utile.h" pour les dépendances nécessaires. L'utilisation de l'instruction préprocesseur #ifndef garantit que le contenu de l'en-tête est inclus une seule fois dans un fichier source lors de la compilation.

Trier.h

Code source :

```
#ifndef TRIE_H
#define TRIE_H
#include "head.h"
void trie();
void TriApogee(Etudiant tabEtudiants[], short nbretudiant);
void TriMoyenne(Etudiant tabEtudiants[], short nbretudiant);
void TriDateInscription(Etudiant tabEtudiants[], short nbretudiant);
// on utilise cette fonction pour transformer la date de form jj/mm/aaaa a aaaa/mm/jj pour trie par date d'inscription
char* nouvDateFromat(char date[]);
#endif
```

Explication :

Ce fichier d'en-tête "Trie.h" définit plusieurs fonctions utilisées pour trier les étudiants selon différents critères. Les fonctions déclarées sont trie(), TriApogee(), TriMoyenne(), TriDateInscription(), et

nouvDateFromat()). L'utilisation de l'instruction préprocesseur `#ifndef` garantit que le contenu de l'en-tête est inclus une seule fois dans un fichier source lors de la compilation.

Rechercher.h

Code source :

```
#ifndef RECHERCHE_H
#define RECHERCHE_H
#include "head.h"
void rechercherEtudiant();
#endif
```

Explication :

Ce fichier d'en-tête "Recherche.h" définit une fonction `rechercherEtudiant()` utilisée pour rechercher des étudiants dans une base de données. L'utilisation de l'instruction préprocesseur `#ifndef` garantit que le contenu de l'en-tête est inclus une seule fois dans un fichier source lors de la compilation.

Supprimer.h

Code source :

```
#ifndef SUPPRIME_H
#define SUPPRIME_H
#include "head.h"
void supprimerEtudiant();
#endif
```

Explication :

Ce fichier d'en-tête "Supprime.h" contient la déclaration de la fonction `supprimerEtudiant()`, utilisée pour supprimer un étudiant de la base de données. L'utilisation de l'instruction préprocesseur `#ifndef` permet de s'assurer que le contenu de l'en-tête est inclus une seule fois dans un fichier source lors de la compilation.

Utile.h

Code source:

```
#ifndef UTILE_H
#define UTILE_H
```

```
#include "head.h"
// effacer le terminal
void clear();
// cette fonction supprime les \n
void deletnewline(char *string);
// cette fonction verifier si data d'inscription donnee est valid ou no
char dateEstValid(char *date);
// cette fonction verifier si l'appoge donnee est valid ou no
char apogeeEstValid(int apogee);
// affiche un etudiant
void afficherEtudiant(Etudiant etudiant);
#endif
```

Explication :

Ce fichier d'en-tête "Utile.h" contient plusieurs fonctions utilitaires utilisées dans le programme. Voici ce qu'elles font :

clear(): Efface le contenu du terminal.

deletnewline(char *string): Supprime les caractères de nouvelle ligne ('\n') d'une chaîne de caractères.

dateEstValid(char *date): Vérifie si la date d'inscription donnée est valide au format "jj/mm/aaaa".

apogeeEstValid(int apogee): Vérifie si l'apogée donné est valide, c'est-à-dire s'il contient exactement 7 chiffres et n'existe pas déjà dans la base de données.

afficherEtudiant(Etudiant etudiant): Affiche les informations d'un étudiant.

Programme Principale

Code source:

```
#include <stdio.h>
#include "Affiche.h"
#include "Ajoute.h"
#include "Modifier.h"
#include "Recherche.h"
#include "Supprime.h"
#include "Trie.h"
#include "Utile.h"
int main(){
    int choix;
    // affiche les taches principale de programme.
    printf("-----\n");
```

```
printf(ANSI_COLOR_BLUE"Center De Gestion \n" ANSI_COLOR_RESET);
printf("-----\n");
printf("1| Ajouter un ou plusieurs étudiants.\n");
printf("2| Modifier les informations sur l'étudiant.\n");
printf("3| Consulter.\n");
printf("4| Trier les étudiants.\n");
printf("5| Rechercher.\n");
printf("6| Supprimer.\n");
printf(ANSI_COLOR_RED "0| Quitter.\n" ANSI_COLOR_RESET);
printf("-----\n");
printf("Veuillez choisir un numéro parmi cette liste : \n");
printf("-----\n");
printf("Choix: ");
scanf("%d",&choix);
switch (choix){
case 1:
    clear();
    ajoutEtudiants();
    break;
case 2:
    clear();
    modifierEtudiant();
    break;
case 3:
    clear();
    consulter();
    break;
case 4:
    clear();
    trie();
    break;
case 5:
    clear();
    rechercherEtudiant();
    break;
case 6:
    clear();
    supprimeEtudiant();
    break;
case 0:
```

```

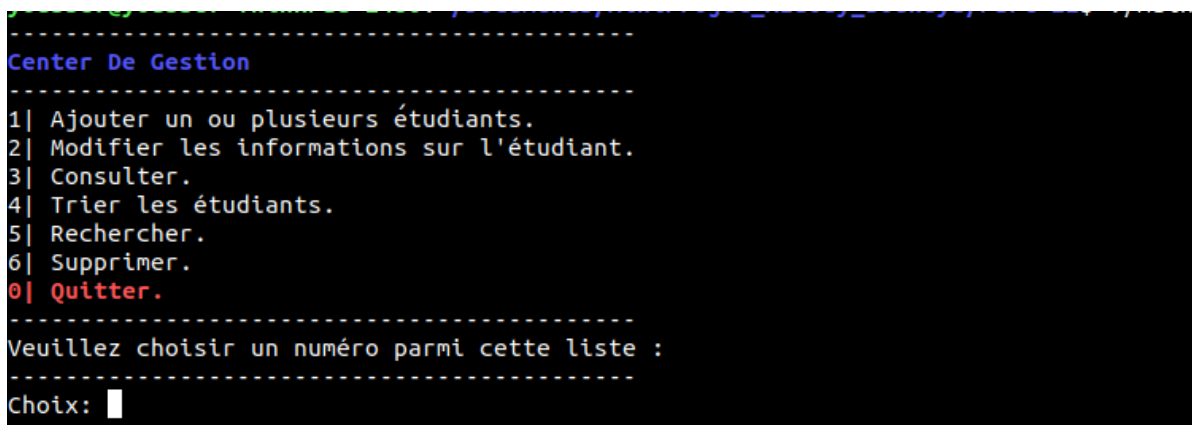
clear();
printf("*****\n");
printf("*");
printf("\033[1;34m Au revoir \033[0m");
printf("*\n");
printf("*****\n");
return 0;
default:
clear();
printf(ANSI_COLOR_RED"Votre choix n'est pas dans la liste \n"ANSI_COLOR_RESET);
}
main();
}

```

Explication :

Ce code est un programme de gestion d'informations étudiantes. Voici un aperçu de ses fonctionnalités :

- 1. Ajouter un ou plusieurs étudiants** : Permet à l'utilisateur d'ajouter de nouveaux étudiants à la base de données.
- 2. Modifier les informations sur l'étudiant** : Permet de mettre à jour les informations d'un étudiant existant.
- 3. Consulter** : Permet de consulter les informations des étudiants déjà enregistrés.
- 4. Trier les étudiants** : Permet de trier les étudiants selon certains critères (non spécifiés dans le code).
- 5. Rechercher** : Permet à l'utilisateur de rechercher un étudiant par son nom ou un autre critère.
- 6. Supprimer** : Permet de supprimer un étudiant de la base de données.
- 7. Quitter** : Permet à l'utilisateur de quitter le programme.



```

-----
Center De Gestion
-----
1| Ajouter un ou plusieurs étudiants.
2| Modifier les informations sur l'étudiant.
3| Consulter.
4| Trier les étudiants.
5| Rechercher.
6| Supprimer.
0| Quitter.
-----
Veuillez choisir un numéro parmi cette liste :
-----
Choix: 1

```

Chaque option du menu est associée à une fonction définie dans des fichiers externes inclus dans le code (Affiche.h, Ajoute.h, Modifier.h, Recherche.h, Supprime.h, Trie.h, Utile.h), ce qui permet une modularité et une organisation du code. Le programme utilise également des couleurs pour améliorer la lisibilité de l'interface utilisateur.

Conclusion

En conclusion, ce projet d'application de gestion des informations des étudiants en langage C illustre l'importance de la programmation modulaire dans le développement logiciel. En divisant les fichiers en deux catégories principales - les fichiers header et les fichiers sources - nous avons pu organiser le code de manière structurée et compréhensible. Les fichiers header fournissent les déclarations des fonctions, tandis que les fichiers sources contiennent l'implémentation réelle de ces fonctions. Dans les sections suivantes, nous examinerons en détail chaque composant de l'application, en mettant en lumière son rôle et son importance dans le fonctionnement global du système.