UNIVERSITÉ M'Hamed BOUGARA – BOUMERDES
FACULTÉ DES SCIENCES
DEPARTEMENT D'INFORMATIQUE

ASD2	1ère année Licence Informatique

Devoir

Objective :

Le but de ce travail est de vous donner l'opportunité de mettre en pratique tout ce que nous avons étudié jusqu'à présent, à travers une application concrète.

Description :

Un système de gestion d'étudiants est un programme qui permet à une école ou une université de suivre les informations de ses étudiants, telles que leurs noms, coordonnées, cours, notes, présence et autres données pertinentes. Le programme doit être capable de lire à partir d'un fichier et d'écrire dans un fichier pour stocker et récupérer les données des étudiants.

Le système de gestion des étudiants doit avoir une interface utilisateur basée sur un menu qui permet aux administrateurs ou aux enseignants d'effectuer diverses tâches, telles que :

1. **Ajouter des étudiants** : Le programme doit permettre aux utilisateurs d'ajouter de nouveaux étudiants au système en entrant leurs noms, leurs coordonnées et d'autres informations pertinentes.
2. **Mettre à jour les informations des étudiants** : Le programme doit permettre aux utilisateurs de mettre à jour les informations d'un étudiant s'il y a des changements, tels qu'un changement d'adresse ou de numéro de téléphone.
3. **Supprimer des étudiants** : Le programme doit permettre aux utilisateurs de supprimer les informations d'un étudiant du système s'ils ont quitté l'école ou l'université.
4. **Afficher des étudiants** : Le programme doit permettre aux utilisateurs de voir les informations d'un étudiant.

Ces tâches doivent être mises en œuvre à l'aide de **Structures C**, **d'Entrées / Sorties de fichiers** et de **Fonctions**.

Structure :

Par exemple, vous pouvez utiliser une structure pour représenter un étudiant est définie avec les champs suivants :

- **`id`**: un entier représentant l'identifiant de l'étudiant.
- **`nom`** : une chaîne de caractères représentant le nom de l'étudiant.
- **`prenom`** : une chaîne de caractères représentant le prénom de l'étudiant.
- **`adresse`** : une chaîne de caractères représentant l'adresse de l'étudiant.
- **`email`** : une chaîne de caractères représentant l'adresse e-mail de l'étudiant.
- **`age`** : un entier représentant l'âge de l'étudiant.
- **`moyenne`** : un nombre réel représentant la moyenne de l'étudiant.
- **`module`** : un tableaux de type module, la structure **`module`** est définie avec les champs suivants :
    - **`nom_module`** : une chaîne de caractères représentant le nom du module.
    - **`coefficient`** : un nombre réel représentant le coefficient du module.
    - **`note`** : un nombre réel représentant la note obtenue par l'étudiant pour ce module.

Fichiers :

Vous pouvez utiliser des Entrées / Sorties de fichiers pour lire les données des étudiants à partir d'un fichier et écrire les données mises à jour dans le fichier, Voici quelques directives à suivre :

- Créer un répertoire **`database`** pour stocker les fichiers de données. Utiliser la fonction **`mkdir()`** pour créer le répertoire.
- Utiliser des fichiers texte pour stocker les informations des étudiants et des modules. Nommer les fichiers texte en utilisant le format **`nom_prenom.txt`.**
- Écrire les informations des étudiants et des modules dans les fichiers texte. Utiliser la fonction **`fprintf()`** pour écrire les données dans les fichiers texte.
- Lire les informations des étudiants et des modules à partir des fichiers texte. Utiliser la fonction **`fscanf()`** pour lire les données à partir des fichiers texte.
- Stocker les fichiers des étudiants dans le répertoire **`database/`**. Pour ce faire, ajouter le préfixe **`database/`** devant le nom de chaque fichier de l'étudiant.

Fonctions :

Vous pouvez également utiliser des fonctions pour effectuer diverses tâches, telles que l'ajout, la suppression ou la mise à jour des informations des étudiants :

- **`ajouter()`:** permet d'ajouter un étudiant dans la liste des étudiants enregistrés. L'utilisateur doit entrer les informations personnelles de l'étudiant (nom, prénom, âge, adresse, etc.) qui seront stockées dans un fichier texte portant le nom de l'étudiant.
- **`afficher()`:** permet d'afficher la liste des étudiants enregistrés avec leurs informations personnelles. Les données sont lues à partir des fichiers texte correspondant à chaque étudiant.
- **`rechercher()`:** permet de rechercher un étudiant enregistré par nom ou prénom. L'utilisateur entre le nom ou prénom de l'étudiant recherché, le programme parcourt la liste des fichiers texte et affiche les informations personnelles de l'étudiant correspondant.

- **`modifier()`:** permet de modifier les informations personnelles d'un étudiant enregistré. L'utilisateur doit entrer le nom de l'étudiant à modifier, puis les nouvelles informations personnelles. Le fichier texte correspondant à l'étudiant est mis à jour avec les nouvelles informations.
- **`supprimer()`:** permet de supprimer un étudiant enregistré de la liste des étudiants. L'utilisateur doit entrer le nom de l'étudiant à supprimer. Le fichier texte correspondant à l'étudiant est supprimé du répertoire "database".
- **`calculer_moyenne()`:** permet de calculer la moyenne d'un étudiant enregistré. L'utilisateur doit entrer le nom de l'étudiant dont il souhaite calculer la moyenne, puis les coefficients et notes de chaque module. Le programme calcule la moyenne et l'affiche.

Fonctionnalités supplémentaires :

Implémentez l'une des fonctionnalités suivantes pour obtenir une excellente note :

- Une interface terminal, voici quelques bibliothèques utiles :
  - **`newt`:** https://pagure.io/newt
  - **`ncurses`:** https://github.com/mirror/ncurses
- La validation des données, par exemple :
  - **`age`** doit être positif et < 150 ans.
  - **`nom`** et **`prénom`** ne doivent contenir que des caractères alphabétiques.
  - **`email`** doit être au format e-mail (<foo>@<bar>.<baz>).
  - **`moyenne`** et **`note de module`** doivent être entre 0 et 20.
  - **`coefficient de module`** ne peut pas être 0.
- Documentation (Utilisation des commentaires pour expliquer chaque partie du code et rendre le programme plus lisible et compréhensible).
- Les documents peuvent être rédigés en **Anglais**.

Remarque : les fonctionnalités de base du programme doivent être implémentées avant l'une de ces fonctionnalités.

Consignes à respecter :

- Un rapport .pdf contenant des explications sur le code et des images d'exécution (documentation), mettre les noms et groupes sur la première page.
- Copier / Coller est interdit.
- L'utilisation de chatGPT est interdite.
- Le travail doit être individuel ou par binôme.
- Utilisez la section objet de l'e-mail comme suit lors de l'envoi du projet : **NomPrenom1_NomPrenom2_GRP**.
- Vous pouvez envoyer soit le code .c avec le rapport .pdf, soit un lien vers le projet sur github (s'il existe), soit un fichier compressé (zip ou rar) de tous les fichiers.
- **La date limite est le 30/05/2023**.

Bon courage.