ATELIER 2: LES FICHIERS

Exercice 1:

Ecrire une fonction qui calcule et retourne le nombre de caractères dans un fichier texte dont le nom est passer en paramètre (Utiliser n'importe quel fichier du répertoire).

Ecrire un programme permettant de tester votre fonction.

Exercice 2:

Voici un fichier texte, « nombre_enfants.txt » qui contient les prénoms des employés d'une société avec le nombre d'enfants correspondant:

```
Ali 4
Ahmed 5
Mariem 3
Salem 0
Nader 1
```

Ouvrez un éditeur de texte (par exemple Bloc Note), copiez ces informations et enregistrez le fichier avec le nom « nombre_enfants.txt ».

1. Déclarer un type enregistrement ***Employe*** contenant les informations suivantes: ***prenom***(chaine de caractère), ***nbEnfant***(entier).
2. Ecrire une fonction qui calcule et retourne la moyenne de nombre d'enfants par employé. La fonction va prendre en paramètre le nom du fichier à ouvrir.
3. Ecrire une fonction qui cherche si le prénom d'un employé existe dans le fichier. La fonction retourne le nombre d'enfant de cet employé s'il existe, -1 si non. La fonction va prendre en paramètre le nom du fichier à ouvrir et le nom de l'employé à chercher.
4. Ecrire une fonction qui affiche tout les employés qui on n enfants ou plus. n est une valeur passer en paramètre avec le nom du fichier.

Exercice 3:

On définit des étudiants par un nom, un prénom et un numéro d'inscription(deux étudiants différents ne peuvent pas avoir le même code). Ecrire en C les fonctions suivantes et donner la fonction ***main*** permettant de les exploiter :

1. ***creerFichier*** qui permet de saisir n étudiants et les enregistre dans un fichier. Le nom du fichier et n sont passer en paramètres.
2. ***ajouterEtudiant*** qui, étant donné un fichier et toutes les informations relatives à un étudiant, permet de rajouter l'étudiant au fichier.
3. ***copierFichier*** qui, étant donnés deux noms de fichiers, crée le deuxième fichier copie du premier (le premier fichier est correspond à un fichier existant).

4. ***afficherListeEtud*** qui liste le contenu d'un fichier dont le nom est donné en paramètre.
5. ***rechercherEtud*** qui permet de retrouver un étudiant dans un fichier donné connaissant son numéro d'inscription. La fonction retourne 1 s'il existe, 0 si non.
6. ***supprimerCode*** qui supprime dans un fichier donné l'étudiant ayant un code donné.
7. ***supprimerPrenom*** qui supprime dans un fichier donné le premier étudiant ayant un prénom donné.

Exercice 4:

Une pharmacie désire gérer son stock de médicament et ses clients. Un client est caractérisé par un nom (chaîne de 20 caractères) et un total crédit (réel). Le total crédit représente la somme de tous les achats que le client a effectué à la pharmacie. Un médicament est caractérisé par un code (entier), un nom (chaîne de 20 caractères), une famille (chaine de 30 caractères), un prix unitaire (réel) et une quantité (qt). Les noms des clients et les codes des médicaments sont uniques.
Les médicaments de la pharmacie sont stockés dans un fichier texte dont chaque ligne représente un médicament.
Les clients sont stockés dans un deuxième fichier texte dont chaque ligne représente un client.

Définir les deux structures **Client** et **Medicament**.

Écrire les fonctions suivantes :

1. ***float prixTotal (char FichClient[])*** , qui permet de calculer la somme des crédits de tous les clients de la pharmacie.
2. ***void afficherMed (char FichMed[], int qt),*** qui permet d'afficher tous les médicaments de la pharmacie dont la quantité est inferieur ou égale à **qt**
3. ***void MaxCredit(char FichClient[])*** , qui affiche le client qui possède le plus grand total crédit.
4. ***void approvisionner (char FichMed [], Medicament M)*** qui permet d'ajouter un médicament M au stock. Si le code correspondant existe déjà, alors on ajoute juste la quantité de M à celle enregistrée dans le fichier, sinon on ajoute le nouveau médicament à la fin du fichier.
5. ***void supprimer (char FichMed [], char famille[]) ,*** qui permet de supprimer tous les médicaments de la famille passée en paramètre.
6. ***int prixMedicament (char FichMed[], int code, float *prix),*** qui permet d'affecter dans le paramètre prix, le prix d'un médicament en fonction de son code, cette fonction retourne 0 ou bien 1 suivant l'existence du médicament demandé dans le fichier.
7. ***int Vendre(char FichMed[], char FichClient[], int code, int qt, char NomClient[]),*** qui permet de vendre une quantité qt d'un médicament en fonction de son code au client NomClient, cette opération de vente aura pour effet de déduire la quantité achetée du stock du médicament et d'augmenter le total crédit du client avec le prix payer.