Rapport

–

Projet C – Répertoire



***Programmation en C et C++***

*MENON Valentin*

*MASSINOND Matéo*

*Groupe LSI2*

# Introduction

Le projet "Gestionnaire de Répertoire" est une application en langage C visant à gérer un répertoire de contacts. Il permet aux utilisateurs d'ajouter, de rechercher, de supprimer et d'afficher des contacts. De plus, il prend en charge la sauvegarde et le chargement des données à partir d'un fichier binaire, assurant ainsi la persistance des informations entre les sessions. Ce projet a été réalisé en binôme, ce qui a favorisé la collaboration et l'échange d'idées entre les membres de l'équipe.

# Objectif

L'objectif principal de ce projet est de créer un programme simple et efficace pour gérer des informations de contact telles que :

* Nom
* Prénom
* Numéro de téléphone
* Adresse email

Le projet doit également garantir la validité des données saisies par l'utilisateur et offrir une interface utilisateur intuitive via un menu interactif.

# Structure du Projet

Le projet est organisé comme suit :

1. **Fichiers Source** : Le fichier principal du programme (main.c) et d'autres fichiers peuvent être inclus pour une modularité accrue, comme des fichiers d'en-tête pour les déclarations de fonctions et les définitions de structures.
2. **Makefile** : Un fichier Makefile est utilisé pour automatiser le processus de compilation et d'exécution du programme.
3. **Données** : Les données des contacts sont stockées dans une structure Personne qui contient les champs nécessaires pour chaque contact.

# Structure Personne

typedef struct {

char nom[TAILLE\_MAX];

char prenom[TAILLE\_MAX];

char numero[TAILLE\_MAX];

char email[TAILLE\_MAX];

} Personne;

Chaque instance de la structure Personne représente un contact dans le répertoire.

# Fonctionnalités

Le programme offre plusieurs fonctionnalités clés :

**1. Ajouter une Personne**

L'utilisateur peut saisir les informations d'une nouvelle personne. Des validations sont effectuées pour s'assurer que :

* Le nom et le prénom contiennent uniquement des lettres et des espaces.
* Le numéro de téléphone contient uniquement des chiffres, des espaces et des tirets.
* L'adresse email respecte un format valide grâce à une expression régulière.

**2. Afficher le Répertoire**

Cette fonctionnalité permet à l'utilisateur d'afficher tous les contacts actuellement enregistrés dans le répertoire. Si aucun contact n'est présent, un message approprié est affiché.

**3. Rechercher une Personne**

L'utilisateur peut rechercher un contact par son nom. Une recherche améliorée permet de trouver des contacts par des noms partiels, offrant plus de flexibilité à l'utilisateur.

**4. Supprimer une Personne**

Cette fonctionnalité permet de supprimer un contact en saisissant son nom. Le programme parcourt le répertoire et, si le nom correspond, le contact est retiré de la liste.

**5. Sauvegarder et Charger**

Les données des contacts peuvent être sauvegardées dans un fichier binaire (repertoire.dat) lors de la fermeture du programme. À l'ouverture, les contacts sont chargés à partir de ce fichier pour restaurer l'état précédent.

# Détails de Fonctionnement

## Validation des Entrées

Pour assurer l'intégrité des données, des fonctions spécifiques vérifient les entrées de l'utilisateur :

* **Nom et Prénom** : La fonction est\_alpha vérifie que les chaînes de caractères ne contiennent que des lettres et des espaces.
* **Numéro de Téléphone** : La fonction est\_numero\_valide vérifie que le numéro contient uniquement des chiffres, des espaces ou des tirets.
* **Adresse Email** : La fonction est\_email\_valide utilise des expressions régulières pour valider le format de l'adresse email.

## Sauvegarde et Chargement des Données

Les fonctions Sauvegarder et Charger gèrent respectivement l'écriture et la lecture des données à partir d'un fichier binaire. Cela permet de conserver les informations entre les sessions d'utilisation du programme.

## Tri des Contacts

Le programme utilise la fonction qsort pour trier le répertoire par nom. Cela améliore la recherche et la présentation des contacts.

## Interface Utilisateur

Le programme offre une interface utilisateur simple et intuitive :

* Un menu principal est affiché avec les options disponibles.
* L'utilisateur peut interagir avec le programme en saisissant des choix numériques.
* Les validations d'entrée assurent que l'utilisateur reçoit des instructions claires en cas d'erreur.

# Exécution du Programme

## Installation

Avant de pouvoir exécuter le programme, assurez-vous d'avoir gcc et make installés sur votre système. Vous pouvez les installer sur un système Linux avec la commande suivante :

sudo apt install make gcc

## Compilation et Exécution

Pour compiler et exécuter le programme, utilisez les commandes suivantes :

1. **Compiler** :
2. make
3. **Exécuter** :
4. make run

## Interagir avec le Programme

Une fois le programme lancé, un menu s'affiche :

* Pour ajouter une personne, choisissez l'option 1.
* Pour afficher le répertoire, choisissez l'option 2.
* Pour rechercher, choisissez l'option 3.
* Pour supprimer un contact, choisissez l'option 4.
* Pour sauvegarder et quitter, choisissez l'option 5.

# Conclusion

Ce projet de gestionnaire de répertoire a été un excellent exercice de programmation en C. Il a permis de mettre en pratique des concepts clés tels que la gestion de la mémoire, les structures, les expressions régulières et les fichiers binaires. Les défis rencontrés lors du développement, tels que la validation des entrées et la gestion des fichiers, ont renforcé les compétences techniques des membres de l'équipe.

L'application est prête à être étendue avec des fonctionnalités supplémentaires, comme la modification des contacts, l'importation/exportation de fichiers au format CSV, ou encore une interface graphique. Ces améliorations pourraient rendre le programme encore plus convivial et fonctionnel.

# Remerciements

Nous tenons à remercier notre enseignant pour ses conseils et son soutien tout au long de ce projet. Nous remercions également nos camarades pour leurs retours constructifs qui ont permis d'améliorer notre code et notre approche.

# Auteurs

* **MASSINOND Matéo** [mateo.massinond@efrei.net](mailto:mateo.massinond@efrei.net)
* **MENON Valentin** [valentin.menon@efrei.net](mailto:valentin.menon@efrei.net)