Année 2021/2022

SAE S1.02

COMPARAISONS D'APPROCHES ALGORITHMIQUES

Projet réalisé par RENAUX. A et YIFRU.A.
Tuteur : E.KERRIEN.

I. Choix et structures utilisé(e)s.

Questions à se poser :
- les noms : sont-ils en majuscules ? Minuscules ?
- les dates : contiennent-elles les jours ? Les mois ? Seulement les années ?
- le genre : doivent-ils apparaître en tant que booléens (1 ou 2) ou sous forme textuelle telle que Homme ou Femme ?

Lors de la première lecture du fichier avec le programme, le temps d'y procéder est relativement long en raison de la quantité d'informations présentent dans le fichier .csv.

Une structure possible est du style :

```
["ALAIN",[0,2,5]]
```

Finalement, on remarque que les noms contenus dans le fichier sont rangés par ordre alphabétique on peut soit utiliser un tableau trié (un tableau de structure qui contient le sexe, le prénom et l'année) et rechercher une donnée par la méthode de dichotomie ou en utilisant une structure de données équivalente : l'arbre binaire de recherche.

Voici la structure finale du programme :

```c
struct person{
    char *name; //Hash key
    int sex[2]; //sex[0]=nb_women & sex[1]=nb_men => nb_total = sex[0] + sex[1]
    int birthdate[4]; // birthdate[0]=min_w, birthdate[1]=max_w, birthdate[2]=min_m & birthdate[3]=max_m
    struct person *next;
};
typedef struct person person;
```

Elle permet de définir le nom de type chaîne de caractères, le genre d'un individu qui sera ici 1 ou 2. 2 si c'est une femme et 1 si c'est un homme, l'année de naissance de l'individu qui possède 4 nombres entiers et un pointeur *next d'enchainer sur la liste s'il ya une collision (équivaut hash(e1) = hash(e2)) Ce pointeur nous permettra donc de parcourir la liste personne et de passer à la personne suivante de la liste.

Nous avons choisi de définir la structure “person” pour nous permettre de positionner les variables au bon endroits avec les bons types.
Toutes les structures de données considérer pour stocker et gérer une recherche sont citées ci-dessous :

	Tableau (non trié)	Liste	Tableau (trié)	Arbre Binaire de Recherche	Table de Hachage
Recherche(x)	O(n)	O(n)	O(logn)	O(logn)	O(1)
Insertion(x)	O(1)	O(1)	O(n)	O(logn)	O(1)
Suppression(x)	O(n)	O(n)	O(n)	O(logn)	O(1)

Après quelque considération, nous avons décider d’essayer la table de hachage.

II. Fonctionnalités.

Ce programme permet d’afficher le nombre de personnes étant nées à une date X, de calculer le nombre de prénoms total de la liste les différentes statistiques d’un prénom c’est-à-dire : le nombre d’individu total enregistrés, l’année de la première apparition d’un prénom ainsi que sa dernière.

Le nombre de naissance il faut se baser sur tous les prénoms enregistrés depuis l’année 1900. Ensuite, nous demandons à l’utilisateur de choisir un genre de prénom : soit 1 soit 2. Si un prénom est masculin ET féminin, il affichera donc les deux.

III. Limites du programme.

Le programme ne pourra pas aller au-delà des informations comprises dans le fichier. Si le fichier doit changer alors il faudra également changer le ficher. Si par exemple, le fichier vient à être changé, que des noms ont ajoutés, des dates ou bien des genre, il faudra alors penser à modifier l’accès au fichier ainsi que certaines valeurs afin de permettre au programme de lire l’entièreté du fichier.

Même si la taille du tableau doit être prédéfini, le tableau alloué dynamiquement pourra contenir plus de données. Mais il y aura plus de liste chaînée, ce qui va ralentir la recherche de donnée, par exemple par nom.

IV. Binôme.

Nous sommes parti sur une base du 50/50.