IMA 3ème année
Programmation Avancé

TP6 Bibliothèque de listes chaînées de chaînes

1 Objectifs

— Savoir construire une bibliothèque à partir de code existant.
— Utiliser les fonctions de `string.h` (comparaison, copie).
— Connaître les options de `gcc` pour la compilation séparée.
— Savoir construire un `makefile`.

Contexte et préparation : Dans ce TP, nous allons récupérer du code existant pour construire une bibliothèque de listes que nous utiliserons au TP suivant. Cette bibliothèque contiendra les fonctions de base sur les listes chaînées de chaînes de caractères, triées ! Les chaînes de caractères auront une taille maximale `MAXSIZE` fixée à `30`.

IMPORTANT : Ce TP sera réutilisé. Il servira d'inspiration pour le *TP7 Arbres* et votre librairie sera entièrement repris pour le *TP8 Tables de Hachage*

2 Questions du TP (à faire impérativement)

Avant de commencer, récupérer les fichiers `~wrudamet/public/IMA3/TP6/listechaines.h` et `~wrudamet/public/IMA3/TP6/accompttp.c`

2.1 Copier/coller rapides et ajustements

À partir des codes dont vous disposez, créer un fichier `listechaines.c` conforme au `.h` fourni en récupérant et adaptant les fonctions déjà écrites. Les fonctions de chargement à partir d'un fichier et d'insertion dans l'ordre alphabétique sont données dans le fichier `accompttp.c`.
Écrire dans `listechaines.c` la fonction main permettant de tester toutes les fonctions.

2.2 Compilation séparée, librairie statique

1. Déplacer le main de `listechaines.c` dans un fichier nommé `tp6.c`
2. Compiler les deux `.c` précédents (avec l'option `-c`) et lier les deux fichiers.o obtenus pour produire un exécutable nommé `tp6` (cf cours).
3. Si vous modifiez `tp6.c` (et uniquement ce fichier là), quelles lignes de compilation/liaison devez-vous faire ?
4. Créer une librairie statique de nom `liblistechaines.a` à partir de `listechaines.o` (commande `ar`)
5. Quelle commande permet de lier `tp6.o` et la librairie `liblistechaines.a` ?
6. Lancer le programme et tester.
7. Si vous modifiez `tp6.c`, quelles commandes de compilation/liaison devez-vous faire ?

3 Questions s'il vous reste du temps

1. Vérifier que vous pouvez utiliser la librairie du voisin (copier son `.a` dans le répertoire courant)
2. Observer le contenu des binaires `tp6.o` et `listechaines.o` à l'aide de la commande `nm`
3. Construire le `Makefile` pour la compilation séparée avec création de librairie statique

4 Annexes

```c
#include<stdio.h>
#include<stdbool.h>
#include<stdlib.h>
#include<string.h>

#define MAXSIZE 30

//Declaration de liste chainee de chaines de caracteres
struct cell {
  char val[MAXSIZE];
  struct cell * suiv;
};

//OPTIONNEL: vous pouvez utiliser ces typedefs si vous voulez !
typedef struct cell   Cellule;
typedef struct cell * Liste;
typedef struct cell * ptCellule;

//Affichage de la liste en ligne
void afficher_liste(struct cell *);

//Ajout d'un mot en tete de la liste
void ajout_tete(struct cell **, char *);

//Suppression du mot en tete de la liste
void supp_tete(struct cell **);

//Ajout un mot dans une liste supposee
// triee dans l'ordre alphabetique
void ajout_alphab(struct cell **, char *);

//Dit si un mot donne est dans la liste
//pas forcement triee
bool appartient(struct cell *, char *);

//Donne la taille de la liste.
int taille(struct cell *);

//construit une liste triee a partir d'un fichier
void charge_fichier(FILE *, struct cell **);

//Destruction de Liste.
void detruire_liste(struct cell **);
```

`listechaines.h`

```c
/*
 * Acompagnement du TP, code exemple pour vous aider.
*/

//ajout d'un mot dans une liste chainee triée
void ajout_alphab(struct cell ** pl, char * mot)
{
    // liste vide ou mot < mot prochain => ajout en tête
    if ( (*pl == NULL) || (strcmp(mot, (*pl)->val) < 0) )
    {
        ajout_tete(pl,mot);
    }
    else
    {
        // mot > mot prochain => ajouter après dans la liste
        if (strcmp(mot, (*pl)->val) > 0)
        {
            ajout_alphab(&(*pl)->suiv,mot);
        }
        //else => mot déjà dans la liste, ne rien faire?
    }
}


//construit une liste triee a partir d'un fichier
void charge_fichier(FILE * fp, struct cell ** pl)
{
    char mot[MAXSIZE];

    //Nb d'elements à lire dans chaque fscanf
    const int NB_A_LIRE = 1;

    //`man fscanf` pour comprendre les valeurs de retour!!!
    while (fscanf(fp, "%s", mot) == NB_A_LIRE)
    {
        ajout_alphab(pl, mot);
    }

    //On peut tester la bonne ou mauvaise terminaison de la lecture
    if(feof(fp))    printf("Fin normal de lecture\n");
    if(ferror(fp))  printf("ERREUR de lecture\n");

}
```

Fichier d'accompagnement `accompttp.c`