ISIMA Première Année

Compte-Rendu de TP

Structures de Données

Gestion de news à partir d'une liste chaînée

Benjamin BARBESANGE
Pierre-Loup PISSAVY
Groupe G21

Enseignant :
Michelle CHABROL

février 2015

ISIMA

1 | Présentation

Le but de ce TP est de concevoir un ensemble de fonctions permettant de gérer des news sous forme de messages, chacun d'entre eux ayant une date de début et de fin de validité. On doit faire usage d'une liste chaînée. Les news sont ordonnées dans la liste chaînée selon l'ordre décroissant de la date de début (de la plus récente à la plus ancienne).

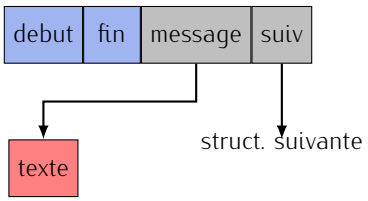
Les messages et informations satellites sont enregistrés dans un fichier, à raison d'une ligne par message. Ce fichier est supposé correct.

Les opérations suivantes sont permises :

- Charger une liste depuis un fichier,
- Sauvegarder une liste dans un fichier,
- Afficher les messages du jour,
- Supprimer les messages obsolètes,
- Modifier une date de début sur tous les messages,
- Afficher tous les messages contenant une chaîne particulière.

1.1 Structure de données employée

(a) Structure utilisée



(b) Code

Code C

```c
typedef struct _news_t {
  int  debut;
  int  fin;
  char *texte;
  struct _news_t *suiv;
} news_t;
```

FIGURE 1.1 – Structure et code correspondant

1.2 Organisation du code source

Nous avons défini deux modules, le premier est dédié à la gestion des listes chaînées (adjonction, recherche, suppression etc.), et le second à la gestion des news (lecture, sauvegarde, modifications etc.). Enfin, le programme principal utilise conjointement ces deux modules pour réaliser le traitement voulu.

1.2.1 Gestion de liste chaînée

- `src/liste_news.h`
- `src/liste_news.c`

1.2.2 Gestion de news

- `src/gestion_news.h`
- `src/gestion_news.c`

1.2.3 Programme principal

- `src/main.c`

2 | Détails du programme

2.1 Gestion de liste chaînée

Code C

```c
/*        liste_news.h
        Header

        -----| GESTION DE NEWS PAR LISTE CHAINEE |-----

        BARBESANGE Benjamin,
        PISSAVY Pierre-Loup

        ISIMA 1ere Annee, 2014-2015
*/

#ifndef __LISTE_NEWS_H__
#define __LISTE_NEWS_H__

  #include <string.h>

  typedef struct _news_t {
    int  debut;
    int  fin;
    char *texte;
    struct _news_t *suiv;
  } news_t;

  typedef struct _news_t cell_t;

  cell_t ** rech_prec(cell_t **, int, short int*);
  void supp_cell(cell_t **);
  void liberer_liste(cell_t **);
  void ins_cell(cell_t **, cell_t *);
  cell_t * creer_cell(int, int, char *);

#endif
```

Code C

```c
/*        liste_news.c
        Fonctions de gestion de la liste chainee

        -----| GESTION DE NEWS PAR LISTE CHAINEE |-----

        BARBESANGE Benjamin,
```

```
        PISSAVY Pierre-Loup

        ISIMA 1ere Annee, 2014-2015
*/

#include <stdio.h>
#include <stdlib.h>
#include "liste_news.h"

void adj_cell(cell_t **prec, cell_t *elt) {
  elt->suiv = (*prec);
  (*prec) = elt;
}

cell_t ** rech_prec(cell_t **liste, int debut, short int *existe) {
  cell_t **prec = liste;
  while ((*prec) && (*prec)->debut > debut) {
    prec = &((*prec)->suiv);
  }
  /* Booleen de presence     */
  /* 1 : present             */
  /* 0 : absent              */
  *existe = (*prec && (*prec)->debut == debut)?1:0;
  return prec;
}

void supp_cell(cell_t **prec) {
  cell_t *elt = *prec;
  *prec = elt->suiv;
  free(elt->texte);
  free(elt);
}

void liberer_liste(cell_t **liste) {
  while (*liste) {
    supp_cell(liste);
  }
  *liste = NULL;
}

void ins_cell(cell_t **liste, cell_t *elt) {
  short int existe;
  cell_t **prec = rech_prec(liste,elt->debut,&existe);
  adj_cell(prec,elt);
}

news_t * creer_cell(int debut, int fin, char *message) {
  news_t *elt = (news_t*) malloc(sizeof(news_t));
  if (elt) {
    elt->debut = debut;
    elt->fin = fin;
    elt->texte = (char*) malloc((strlen(message)+1)*sizeof(char));
    strcpy(elt->texte,message);
  }
  return elt;
```

```
}
```

2.2 Gestion de news

Code C

```c
/*        gestion_news.h
        Header

        -----| GESTION DE NEWS PAR LISTE CHAINEE |-----

        BARBESANGE Benjamin,
        PISSAVY Pierre-Loup

        ISIMA 1ere Annee, 2014-2015
*/

#ifndef __GESTION_NEWS_H__
#define __GESTION_NEWS_H__

  #include "liste_news.h"

  int charger(cell_t **, char *);
  int sauver(cell_t *, char *);
  int getDate();
  void afficher_messages_date(cell_t *, int);
  void afficher_message(cell_t *);
  void afficher_liste(cell_t *);
  void afficher_messages_jour(cell_t *);
  void afficher_messages_motif(cell_t *, char *);
  void supprimer_obsoletes(cell_t **);
  void remplacer_date(cell_t **, int, int);

#endif
```

Code C

```c
/*        gestion_news.c
        Fonctions de gestion des news

        -----| GESTION DE NEWS PAR LISTE CHAINEE |-----

        BARBESANGE Benjamin,
        PISSAVY Pierre-Loup

        ISIMA 1ere Annee, 2014-2015
*/

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include "gestion_news.h"

/* taille maximale de la chaine de caractere du message */
#define SIZE_BUF 118
```

```c

char buf[SIZE_BUF+1];

/*        int charger(cell_t **liste, char *nom_fichier)
              Permet de charger les donnees d'un fichier passe en parametre
              dans une liste dont l'adresse du pointeur de tete est
              egalement passe en parametre

        Entrees :
              cell_t **liste: adresse du pointeur de tete de liste dans
              char *nom_fichier : chemin et nom du fichier a partir duquel charger la liste

        Sortie :
              Entier indiquant un code d'erreur
                    0 si aucune erreur
                    1 si le fichier ne s'est pas charge
                    2 si probleme d'allocation d'un element dans la liste
*/
int charger(cell_t **liste, char *nom_fichier) {
  int ret = 1;
  FILE *fichier = fopen(nom_fichier,"r");
  int debut, fin;
  char *scan;
  cell_t *tmp;
  if (fichier) {
    ret = 0;
    while (!feof(fichier) && ret == 0 && fgets(buf,SIZE_BUF+1,fichier)) {
      /* suppression du caractere \n residuel si le texte fait moins de 100 caracteres */
      scan = strchr(buf,'\n');
      if (scan) {
        *scan = '\0';
      }
      sscanf(buf,"%d %d",&debut, &fin);
      tmp = creer_cell(debut,fin,&buf[18]);
      if (tmp) {
        ins_cell(liste,tmp);
      } else {
        /* Code erreur: allocation de cellule */
        ret = 2;
      }
    }
    fclose(fichier);
  }
  return ret;
}

/*         int sauver(cell_t *liste, char *nom_fichier)
        Sauvgarde les donnees de la liste chainee dans un fichier passe
        en parametres

        Entrees :
              cell_t *liste : pointeur sur le premier element de la liste chainee
              char *nom_fichier : chemin et nom du fichier dans lequel sauvegarder
                                                    ce fichier sera cree et effacera
↪ tout autre fichier
```

```
                                                du meme nom dans le repertoire
↪ specifie s'il y en a

        Sortie :
                Entier renvoyant un code d'erreur
                        0 si aucune erreur
                        1 si probleme de creation du fichier
*/
int sauver(cell_t *liste, char *nom_fichier) {
  int ret = 1;
  cell_t *cour = liste;
  FILE *fichier = fopen(nom_fichier,"w+");
  if (fichier) {
    while (cour) {
      fprintf(fichier,"%d %d %s\n",cour->debut,cour->fin,cour->texte);
      cour = cour->suiv;
    }
    fclose(fichier);
    ret = 0;
  }
  return ret;
}

/*        int getDate()
        Retourne la date du jour au format AAAAMMJJ

        Entrees :
                Aucune

        Sortie :
                Entier representant la date du jour de la forme AAAAMMJJ
*/
int getDate() {
  time_t now = time(NULL);
  int datejour;
  struct tm t = *localtime(&now);
  datejour = (t.tm_year+1900)*10000+(t.tm_mon+1)*100+t.tm_mday;
  return datejour;
}

/*        void afficher_message(cell_t *m)
        Affiche le message d'un element de la liste chainee sous la forme
        "Debut : *debut*, Fin: *fin*, Message: *texte*"

        Entrees :
                cel_t *m : pointeur sur un element de la liste chainee

        Sortie :
                Aucune
*/
void afficher_message(cell_t *m) {
  printf("Debut: %d, Fin: %d, Message: %s\n",m->debut,m->fin,m->texte);
}

/*        void afficher_liste(cell_t *l)
        Affiche le contenu integral de la liste suivant le schema de la fonction
```

```
129         afficher_message(cell_t *m)
130
131         Entrees :
132                 cell_t *l : pointeur sur le premier element de la liste
133
134         Sortie :
135                 Aucune
136 */
137 void afficher_liste(cell_t *l) {
138   cell_t *cour = l;
139   while (cour) {
140     afficher_message(cour);
141     cour = cour->suiv;
142   }
143 }
144
145 /*        void traiter_elt_date_debut(cell_t *liste, int date, void (*fun)(cell_t *))
146         Fonction permettant d'appliquer une fonction a tous les elements ayant
147         une date de debut passe en parametre
148
149         Entrees :
150                 cell_t *liste : pointeur sur le premier element de la liste chainee
151                 int date : date de debut des elements a traiter, de la forme AAAAMMJJ
152                 void (*fun)(cell_t*) : fonction a appliquer aux elements correspondant a la
153                                                                 date passee en parametre
154                                                                 cette fonction prend un pointeur ↩
↪ sur un element de la liste chainee
155
156         Sortie :
157                 Aucune
158 */
159 void traiter_elt_date_debut(cell_t *liste, int date, void (*fun)(cell_t *)) {
160   cell_t *cour = liste;
161   while (cour->debut > date) {
162     cour = cour->suiv;
163   }
164   while (cour->debut == date) {
165     fun(cour);
166     cour = cour->suiv;
167   }
168 }
169
170 /*        void afficher_messages_date(cell_t *liste, int date)
171         Affiche les messages de la liste correspondant a la date passee en parametre
172
173         Entrees :
174                 cell_t *liste : pointeur sur le premier element de la liste chainee
175                 int date : date de debut des elements a traiter, de la forme AAAAMMJJ
176
177         Sortie :
178                 Aucune
179 */
180 void afficher_messages_date(cell_t *liste, int date) {
181   traiter_elt_date_debut(liste,date,&afficher_message);
182 }
183
```

```c
/*        void afficher_messages_jour(cell_t *liste)
        Affiche les messages de la liste correspondant a la date du jour

        Entrees :
                cell_t *liste : pointeur sur le premier element de la liste chainee

        Sortie :
                Aucune
*/
void afficher_messages_jour(cell_t *liste) {
  afficher_messages_date(liste,getDate());
}

/*        void afficher_messages_motif(cell_t *liste, char *motif)
        Affiche les messages de la liste chainee correspondant a un motif
        passe en parametre

        Entrees :
                cell_t *liste : pointeur sur le premier element de la liste chainee
                char *motif : chaine de caractere representant le motif a chercher dans
                                        les messages de la liste

        Sortie :
                Aucune
*/
void afficher_messages_motif(cell_t *liste, char *motif) {
  cell_t *cour = liste;
  while (cour) {
    if (strstr(cour->texte,motif)) {
      afficher_message(cour);
    }
    cour = cour->suiv;
  }
}

/*        void supprimer_obsoletes(cell_t **liste)
        Supprime les messages devenus obsoletes dans la liste chainee
        Les messages sont obsoletes si leur date de fin est anterieure a la date du jour

        Entrees :
                cell_t **liste : adresse du pointeur sur le premier element de la liste chainee

        Sortie :
                Aucune
*/
void supprimer_obsoletes(cell_t **liste) {
  int date = getDate();
  cell_t **prec = liste;
  while (*prec != NULL) {
    if ((*prec)->fin < date) {
      supp_cell(prec);
    } else {
      prec = &((*prec)->suiv);
    }
  }
```

```c
}

/*        void remplacer_date(cell_t **liste, int date, int nvdate)
        Remplace la date de debut des messages ayant une date passee en parametre par une
        autre date aussi passee en parametre
        Cette fonction s'assure egalement que la liste reste triee

        Entrees :
                cell_t **liste : adresse du pointeur sur le premier element de la liste chainee
                int date : date de debut des elements a traiter, de la forme AAAAMMJJ
                int nvdate : nouvelle date de debut a assigner aux elements

        Sortie :
                Aucune
*/
void remplacer_date(cell_t **liste, int date, int nvdate) {
  cell_t *cour = *liste;
  while (cour && cour->debut > date) {
    cour = cour->suiv;
  }
  while (cour && cour->debut == date) {
    if (cour->fin >= nvdate) {
      cour->debut = nvdate;
    }
    cour = cour->suiv;
  }
  sauver(*liste,"/tmp/tmp_tp1_sdd.list");
  liberer_liste(liste);
  charger(liste,"/tmp/tmp_tp1_sdd.list");
}
```

2.3 Programme principal

Code C

```c
/*        main.c
        Fichier principal permettant les tests

        -----| GESTION DE NEWS PAR LISTE CHAINEE |-----

        BARBESANGE Benjamin,
        PISSAVY Pierre-Loup

        ISIMA 1ere Annee, 2014-2015
*/

#include <stdio.h>
#include "gestion_news.h"

int main(int argc, char *argv[]) {
  int old_deb = 20150217;
  int new_deb = 20150226;
  cell_t *liste = NULL;
  if (argc > 1) {
```

```
    charger(&liste,argv[1]);
  }
  if (liste) {
    printf("Affichage de la liste apres recuperation\n");
    afficher_liste(liste);

    printf("\nAffichage des messages du jour\n");
    afficher_messages_jour(liste);

    printf("\nSuppression des messages obsoletes\n");
    supprimer_obsoletes(&liste);
    afficher_liste(liste);

    printf("\nModification des dates de debut: %d -> %d\n",old_deb,new_deb);
    remplacer_date(&liste,old_deb,new_deb);
    afficher_liste(liste);

    liberer_liste(&liste);
  }
  return 0;
}
```

3 | Compte rendu d'exécution

3.1 Makefile

Makefile

```
#Compilateur et options de compilation
CC=gcc
CFLAGS=-Wall -ansi -pedantic -Wextra -g

#Fichiers du projet
SOURCES=main.c gestion_news.c liste_news.c
OBJECTS=$(SOURCES:.c=.o)

#Nom du programme
EXEC=programme

all: $(OBJECTS)
	$(CC) $(CFLAGS) $^ -o $(EXEC)

.c.o:
	$(CC) -c $(CFLAGS) $*.c
clean:
	rm $(OBJECTS) $(EXEC)
```

3.2 Jeux de tests