```c
/* ############################################################################
 *                    semaine.c
 *
 *  Fichier contenant les fonctions (le code) relatives a la gestion de la
 *  liste des semaines
 *
 * ############################################################################*/


#include "semaine.h"

/*----------------------------------------------------------------------------
 *                    createSemaine
 *
 * Procédure qui parcours la liste des semaines et ajoute une semaine (si
 * nécessaire) dans l'ordre en fonction de la chaine de caractère complète
 (ex:
 * 201720114Medecin) puis ajoute l'action correspondant avec la fonction
 * createAction()
 *
 * Entrées
 * semaines : pointeur de la liste des semaines
 * chaine : chaine de caractère correspondant aux informations d'une action
 *
 ----------------------------------------------------------------------------*/
void createSemaine(semaines_t * semaines, char * chaine){
  char              semaine[6];         /* Chaine contenant la partie de la
  semaine*/
  char              actions[13];        /* Chaine contenant la partie de
  l'action*/
  int               cmp=0;              /* Compteur parcourant 'chaine' */
  semaines_t        ** ptrSemaine;      /* Contient le résultat d'une recherche
  dans semaines*/

  for(cmp=0;cmp<19;cmp++){              /*Separation de la chaine de
  caractère*/
    if(cmp < 6){
      semaine[cmp]=chaine[cmp];
    }else actions[cmp-6] = chaine[cmp];
  }

  ptrSemaine = recherche(semaines,semaine);
  if ( ptrSemaine != NULL ){
    if (strcmp((*ptrSemaine)->semaine, semaine) != 0){
      allouеSemaine(*ptrSemaine, semaine);
    }
  }else{
      allouеSemaine(*ptrSemaine, semaine);
  }

  CreateAction((*ptrSemaine)->actions, actions);
}

/*----------------------------------------------------------------------------
```

```
 * »» » » » allouеSemaine
 *
 * »» Fonction qui alloue un élément semaine sur le poiteur passé en
 paramètre et
 * ajoute le champ semaine
 *
 * Entrées
 * »» ptrSemaine : Pointeur de semaine sur lequel il faut faire l'allocation
 * »» nomSemaine : Nom de la semaine à allouer
 *
 ------------------------------------------------------------------------*/
void allouеSemaine(semaines_t * ptrSemaine, char * nomSemaine){
» semaines_t » » * nouvSemaine = (semaines_t *) malloc(sizeof(semaines_t));
» /* contient le nouvel élément alloué */
» nouvSemaine->semaine = nomSemaine;
» nouvSemaine->semaineSuivante = ptrSemaine;
» ptrSemaine = nouvSemaine;
}

/*------------------------------------------------------------------------
-
 *» » » » » recherche
 *
 * »» Permet de parcourir la liste des semaines et de trouver la semaine
 * »correspondante à l'élément recherché
 *
 * Entrées
 * »semaines : pointeur de la liste des semaines à parcourir
 *» semaine : chaine de caractère correspondant à la semaine à trouver
 *
 * Sortie
 * »Pointeur de pointeur de l'élément trouvé(peut-être pointeur sur un élément
 * »NIL
 *
 ------------------------------------------------------------------------*/
semaines_t ** recherche(semaines_t * semaines, char * semaine){
» semaines_t» » ** ptrCour = &semaines;
» /* Contient le pointeur permettant de parcourir les semaines */

» while ((*ptrCour != NULL) && (strcmp((*ptrCour)->semaine,semaine)<0)){
» » *ptrCour = (*ptrCour)->semaineSuivante;
» }
» return ptrCour;
}

/*------------------------------------------------------------------------
 * »» » » lireFichier
 *
 * »» Fonction qui lit un fichier donné en paramètre et créer les semaines
 * correspondantes
 *
 * Entrée
 * »» nomFichier : nom du fichier à lire
 *
 * Sortie
```

```
 * »» retourne le pointeur de la liste des semaines crée¤¬
 * ------------------------------------------------------------------------*/¤¬
semaines_t * lireFichier(char * nomFichier){¤¬
» semaines_t» » * listeSemaine = NULL; »/* Contient la liste de semaine
  résultante*/¤¬
» FILE» » » » » * file = NULL;» » » » » /* Fichier à lire*/¤¬
» char» » » » » » chaine[MAXLISTE] = "";/* Contient la chaine d'une action
  après chaque lecture*/¤¬
¤¬
» file = fopen(nomFichier, "r");¤¬
¤¬
» if (file != NULL){¤¬
» » while(fgets(chaine, MAXLISTE,file) != NULL){¤¬
» » » createSemaine(listeSemaine,chaine);¤¬
» » }¤¬
» » fclose(file);¤¬
» }¤¬
» return listeSemaine;¤¬
}¤¬
¤¬
/*------------------------------------------------------------------------¤¬
 *» » » » sauvegardeSemaine¤¬
 *¤¬
 *» » Fonction qui permet de sauvegarder la structure en parcourant toutes¤¬
 *» les semaines et appelant la fonction de sauvegarde sur la liste des¤¬
 *» actions¤¬
 *¤¬
 *» Entrées¤¬
 *» » listeSemaine : pointeur sur la liste des semaines à sauvegarder¤¬
 *» » nomFichier : nom du fichier où sauvegarder¤¬
 * ------------------------------------------------------------------------*/¤¬
void sauvegardeSemaine(semaines_t * listeSemaine, char * nomFichier){¤¬
» FILE» » » » » » » * file = NULL;» » » » » /* Fichier où sauvegarder*/¤¬
» semaines_t» » » » * ptrCour = listeSemaine; /* Pointeur de structure
  permettant¤¬
» de parcourir la liste des semaines*/¤¬
¤¬
» file = fopen(nomFichier, "r");¤¬
» if (file != NULL){¤¬
» » while (ptrCour != NULL){¤¬
» » » Sauvegarde(file, ptrCour->actions, ptrCour->semaine);¤¬
» » » ptrCour = ptrCour->semaineSuivante;¤¬
» » }¤¬
» » fclose(file);¤¬
» }¤¬
}¤¬
¤¬
/*------------------------------------------------------------------------¤¬
 *» » » » » createListeJourFromActionsSemaine¤¬
 *¤¬
 * »Fonction qui créer la liste des jours contenant une action donnée en¤¬
 * »parcourant la liste des semaines¤¬
 *¤¬
 *» Entrées¤¬
 *» » listeSemaine : pointeur de la liste des semaines à parcourir¤¬
```

```
 *  action : chaine de caractère à rechercher
 *
 *  Sortie
 *  Chaine de caractère correspondant à la liste des jours résultants
 * ------------------------------------------------------------------------*/
char * createListeJourFromActionsSemaine(semaines_t * listeSemaine, char *
action){
  semaines_t    * ptrCour = listeSemaine; /* Variable de parcours de la liste
    des semaines */
  char          * listeDesJours = ""; /* Variable de retour contenant la
    liste des jours */

  while(ptrCour != NULL){
    strcat(listeDesJours, CreateListeFromActions(ptrCour->actions, action));
    ptrCour = ptrCour->semaineSuivante;
  }

  return listeDesJours;
}

/*------------------------------------------------------------------------
 *          supprimeActionInSemaines
 *
 *  Supprime une action de la sdd à partir de l'année, la semaine, le jour
 *  ainsi que l'heure en recherchant la semaine correspondante et appelant la
 *  fonction de suppression sur une liste d'action
 *
 *  Entrées
 *    listeSemaine : pointeur de la liste des semaines à parcourir
 *    chaine : chaine contenant toutes les informations pour supprimer
   l'élément
 *  ------------------------------------------------------------------------*/
void supprimeActionInSemaines(semaines_t * listeSemaine, char * chaine){
  char          semaine[6];     /* chaine correspondant à la partie semaine */
  char          jourHeure[13];  /* chaine correspondant à la partie jour et
    heure */
  int           cmp=0;          /* Variable de parcour de 'chaine' */
  semaines_t    * temp;         /* Variable temporaire pour effectuer la
```