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

    /*-------Déclaration des structures-------*/
    typedef struct Tenreg
    {

        int Eff;
        int cle;
        char *Nom;
        int N_tel;
    } Tenreg;

    typedef struct Tbloc
    {
        Tenreg T[20];
        int Nb;
    } Tbloc;

    typedef struct Entete
    {
        int adrDerBloc;
      // on peut rajouter ici toute information sur le fichier jugée utile
    } Entete;

    typedef struct TObF
    {
        FILE *fichier;
        Entete entete;
    } TObF;

    TObF *f;

/*--------------Fonction d'ouverture d'un fichier--------------*/
int ouvrir(TObF **f,char *chemin,char mode)
{
    *f = malloc(sizeof(TObF));
    char s[3];

    if((mode == 'A') || (mode == 'a')) sprintf(s,"rb+");
    else if ((mode == 'N') || (mode == 'n')) sprintf(s,"wb+");
    else return 0;

    (*f)->fichier = fopen(chemin, s);
    if((*f)->fichier == NULL)   return 0;

    if((mode == 'A') || (mode == 'a'))
    {
        fread(&((*f)->entete), sizeof(Entete), 1, (*f)->fichier);
    }
    else if ((mode == 'N') || (mode == 'n'))
    {
        (*f)->entete.adrDerBloc = 0;

    }
    return 1;
}
/*-------------------------------------------------------------*/

/*--------------Fonction de fermeture d'un fichier--------------*/
void fermer(TObF *f)
{
    //Sauvegarde de l'entête
    rewind(f->fichier);
```

```
    fwrite(&(f->entete), sizeof(Entete), 1, f->fichier);
    fclose(f->fichier);
    free(f);
}
/*-------------------------------------------------------------*/

/*-----------------Fonction de lecture d'un bloc----------------*/
void lireDir(TObF *f,int N_Bloc,Tbloc *buffer)
{
    if(N_Bloc <= (f->entete).adrDerBloc)
    {
        fseek(f->fichier, sizeof(Entete) + (N_Bloc-1) * sizeof(Tbloc),SEEK_SET);
        fread(buffer, 1, sizeof(Tbloc), f->fichier);
    }
}
/*-------------------------------------------------------------*/

/*-----------------Fonction d'écriture d'un bloc----------------*/
void ecrireDir(TObF *f,int N_Bloc,Tbloc *buffer)
{
    if(N_Bloc <= (f->entete).adrDerBloc)
    {
        fseek(f->fichier, sizeof(Entete) + (N_Bloc-1) * sizeof(Tbloc),SEEK_SET);
        fwrite(buffer, 1, sizeof(Tbloc), f->fichier);
    }
}
/*-------------------------------------------------------------*/


int entete(TObF *f,int i)
{
    if(i == 1) return (f->entete).adrDerBloc;
//    else if(i == 2) return (f->entete).nbEnreg;
//    else if(i == 3) return (f->entete).indice_libre;
    else return -1;
}
/*-------------------------------------------------------------*/

/*-------------Fonction de modification de l'entête-------------*/

void aff_entete(TObF *f, int i, int val)
{
    if(i == 1) (f->entete).adrDerBloc = val;

}
/*-------------------------------------------------------------*/

/*--------------------L'allocation d'un bloc--------------------*/
int alloc_bloc(TObF *f)
{
    aff_entete(f,1,entete(f,1)+1);
    return entete(f,1);
}

void Charger(int i)
{
   Tbloc  buf;
   ouvrir(&f,"test.bin",'N');
   int Num=alloc_bloc(f);
   int k;
   for (k=0;k<10;k++)
   {
     buf.T[k].cle=k+100;
     buf.T[k].Eff=0;
```

```
     buf.T[k].Nom="Moh";
     buf.T[k].N_tel=k+2222220;
   }
   buf.Nb=10;
   ecrireDir(f,Num,&buf);
   fermer(f);
printf("fin de charger\n");
}

void Afficher()
{
 Tbloc  buf;
   ouvrir(&f,"test.bin",'A');
   lireDir(f,entete(f,1),&buf);
   printf("Nb = %d\n",buf.Nb);
   int k;
   printf("les enreg : \n");
   for (k=0;k<buf.Nb;k++)
   {
    if (buf.T[k].Eff==0)
     printf("%d %d %s %d\n",buf.T[k].cle, buf.T[k].Eff,buf.T[k].Nom,
buf.T[k].N_tel);
     else printf("enreg efface\n");
   }

   fermer(f);
}

void supprimer(int i,int j)
{ Tbloc  buf;
  ouvrir(&f,"test.bin",'A');
  lireDir(f,entete(f,i),&buf);
  buf.T[j].Eff=1;
  ecrireDir(f,i,&buf);
  fermer(f);

}

int main()
{
    Charger(10);  // charger le fichier avec 10 enreg
    Afficher(10);
    supprimer(1,4); //supprimer le 4eme enregistrement du 1er bloc
    Afficher(10);
    printf("Hello world!\n");
    return 0;
}
```