UMLV - IUT
DUT - INFO
1'ere Annee (2012-2013)

Programmation C
Corrige du TD#7: Structures

```c
/* Exercice 1.1 Définition de nom de type
Définir un type Date pour des variables formées d’un numéro de jour, d’un nom
de mois et d’un numéro d’année.

Exercice 4.1 Fiche
• Ecrire des fonctions de lecture et d’écriture d’une variable de type Date.
  Dans un premier temps, on ne se préocupera pas de la validité de la date entrée.
*/

#include <stdio.h>

typedef struct {
           int  jour;
           char mois[10];
           int  annee;
        } DATE;

DATE LireDate(void);
void AfficheDate(DATE);

int main(void) {
   DATE date;

   date = LireDate();
   AfficheDate(date);

   return 0;
}

void AfficheDate(DATE date) {
   printf("The date is: %d %s  %d\n", date.jour, date.mois, date.annee);
}

DATE LireDate(void) {
    DATE temp;

    printf("Enter a date (in this form:: Day(1-31) Month(Janvier - Decembre) Year(xxxx)):? ")
    scanf("%d %s %d", &temp.jour, temp.mois, &temp.annee);

    return temp;
}
```

```c
/*
Exercice 1.2 Définition de nom de type
```

```
Ecrire la déclaration d’un type Fiche permettant de mémoriser les informations
sur un étudiant :
• son nom ;
• son prenom ;
• sa date de Naissance, de type Date ;
• sa formation, représentée par deux lettres ;
• s’il est redoublant ou non ;
• son groupe de TD, représenté par un entier ;
• ses notes, représentées par un tableau note d’au plus MAXNOTES rels;
• un entier nbnotes indiquant le nombre de notes valides dans le tableau note.

Exercice 4.2 Fiche
• Ecrire les fonctions LireFiche et EcrireFiche de lecture et d’écriture d’une Fiche.
  Aucune note n’est entr´ee par la fonction LireFiche.
• Ecrire une fonction AjouteNote qui reçoit une Fiche et ajoute une note, si cela
  est possible.
• Ecrire une fonction Moyenne qui reçoit une Fiche et renvoie, si cela est possible,
  la moyenne des notes de l’étudiant.
*/

#include <stdio.h>

#define MAXNOTES 10

typedef struct {
        int  jour;
        char mois[10];
        int  annee;
} DATE;

typedef struct {
           char  nom[10];
           char  prenom[10];
           DATE  date_naissance;
           char  info[3];
           char  redoublant;
           int   groupe;
           float notes[MAXNOTES];
           char  nom_module[MAXNOTES][30];
           int   nbnotes;
        } FICHE;

FICHE  LireFiche(void);
void   EcrireFiche(FICHE);
FICHE  AjouteNote(FICHE);

int main(void) {
   FICHE etd_fiche;

   etd_fiche = LireFiche();
   etd_fiche = AjouteNote(etd_fiche);
   etd_fiche = AjouteNote(etd_fiche);
   EcrireFiche(etd_fiche);

   return 0;
}

FICHE  AjouteNote(FICHE fiche) {

   if (fiche.nbnotes < MAXNOTES) {
       printf("Entrez le nom du module: ? ");
       scanf(" %[^\n]s", fiche.nom_module[fiche.nbnotes]);
       printf("Entrez sa notes pour ce module: ? ");
       scanf("%f", &fiche.notes[fiche.nbnotes]);
```

```
       fiche.nbnotes++;
    }
    else
      printf("\n <---- c'est pas possible d'ajuter une note ---->\n");

    return fiche;
}

FICHE  LireFiche(void) {
   FICHE fiche;

   printf("Entrez le nom de l'etudiant:? ");
   scanf(" %s", fiche.nom);

   printf("Son prenom:? ");
   scanf(" %s", fiche.prenom);

   printf("Sa date de Naissance: (dd mm yyyy) ? ");
   scanf("%d %s %d", &fiche.date_naissance.jour, fiche.date_naissance.mois, &fiche.date_naiss

   printf("Sa formation: ? ");
   scanf(" %s", fiche.info);

   printf("Il est redoublant ou non (O:Oui, N:Non) ? ");
   scanf(" %c", &fiche.redoublant);

   printf("Son groupe: ?");
   scanf("%d", &fiche.groupe);

   fiche.nbnotes=0;
   return fiche;
}

void   EcrireFiche(FICHE fiche) {
   int i;

   printf("La fiche de %s, %s\n", fiche.nom, fiche.prenom);
   printf("\tSa date de Naissance est le %d %s %d\n", fiche.date_naissance.jour,
                                                      fiche.date_naissance.mois,
                                                      fiche.date_naissance.annee);
   printf("\tSa formation: %s\n", fiche.info);
   printf("\tSon groupe: %d\n", fiche.groupe);

   printf("\tIl est redoublant: %s\n", (fiche.redoublant == 'O')? "Oui":"Non");
   printf("\nSes notes: \n");

   printf("\t   Nom de Module          Note\n");
   printf("\t-----------------------------------\n");
   for (i=0; i<fiche.nbnotes; i++)
      printf("\t %17s \t %5.2f\n", fiche.nom_module[i], fiche.notes[i]);
}


---------------------------------------------------------------------------------------------

/*
Exercice 2. Polynomes
• Définir un type Polynomes permettant de manipuler des polynomes à une variable,
  à coefficients réels de degré inférieur à DGMAX.
• Ecrire une fonction LirePolynome effectuant la saisie monome par monome d’un
polynome. Pour chaque monome, on indiquera le degré puis le coefficient.
*/

#include <stdio.h>
```

```c
#define MAXDEGREE 101
#define MAX(x, y) (x > y ? x : y)

typedef struct {
    int degree;
    int coef[MAXDEGREE];
} polynomial;

typedef struct {
     int coef;
     int expo;
} term;

polynomial add_poly(polynomial, polynomial);
polynomial mul_poly(polynomial, polynomial);
polynomial mul_poly_by_term(polynomial, term);
void       display_poly(polynomial, char);
polynomial read_poly();

int main(void) {
   polynomial a, b, c;
   int i;

   printf("Enter the terms of the first polynomial:\n");
   a = read_poly();

   printf("Enter the terms of the second polynomial:\n");
   b = read_poly();

   display_poly(a, 'a');
   display_poly(b, 'b');

  c = add_poly(a, b);

   printf("\nPolynomial c = a + b\n");
   display_poly(c, 'c');

   c = mul_poly(a, b);

   printf("\nPolynomial c = a * b\n");
   display_poly(c, 'c');
   return 0;
}


polynomial mul_poly_by_term(polynomial a, term t) {
   int i;

   for (i=a.degree; i >= 0; i--) {
       a.coef[i + t.expo] = a.coef[i] * t.coef;
       if (t.expo) a.coef[i] = 0;
   }
   a.degree = a.degree + t.expo;
   return a;
}


polynomial mul_poly(polynomial a, polynomial b) {
     polynomial c = {0,0}; term t;
     int i;

     for (i=a.degree; i >= 0; i--) {
         t.coef = a.coef[i]; t.expo=i;
         c = add_poly(c, mul_poly_by_term(b, t));
     }
```

```c
     return c;
}

polynomial add_poly(polynomial a, polynomial b) {
    polynomial c;
    int i;

    c.degree = MAX(a.degree, b.degree);
    for (i=c.degree; i >= 0; i--)
        c.coef[i] = a.coef[i] + b.coef[i];

    return c;
}

polynomial read_poly() {
   polynomial a={0,0};  /* Initialize the polynomial to zero */
   int i, nb_terms, expo, coef, max_expo=0;

   do {
     printf("Enter number of terms in your polynomial (a positive integer): ");
     scanf("%d", &nb_terms);
     if (nb_terms <= 0) printf("Invalide value assigend to nb_terms, another trail\n");
   } while (nb_terms <= 0);

   for (i=0; i < nb_terms; i++) {
      printf("Enter the exponent (positive integer) :"); scanf("%d", &expo);
      if (expo < 0) {
         printf("Invalid exponent ...Enter a valid exponent\n");
         i--;
         continue;
      }
      if (expo > max_expo) max_expo = expo;
      printf("Enter the coefficient: "); scanf("%d", &coef);
      a.coef[expo] += coef;  /* Accumulate similar terms */
   }

   a.degree = max_expo;
   return a;
}


void   display_poly(polynomial a, char poly_name) {
    int i;

    printf("\nPolynomail %c = ", poly_name);

    for(i = a.degree; i > 0; i--)
       if (a.coef[i]) printf("%dx^%d + ", a.coef[i], i);

    if (a.coef[i]) printf("%d", a.coef[i]); else printf("0");
    printf("\n");
}


---------------------------------------------------------------------------------------------

/*
Exercice 3. Rationnel
• Définir un type Rationnel composé de deux entiers: un numérateur et un
  dénominateur.
• Ecrire une fonction LireRationnel qui effectue la lecture d’un rationnel
  valide. Le rationnel mémorisé aura été simplifié.
• Ecrire une fonction SommeRationnel qui retourne la somme des deux rationnels
  valides passés en argument. Le rationnel retourné aura été simplifié.
```

```
*/

#include <stdio.h>

typedef struct {
           int  num;
           int  denom;
        } Rationnel;

Rationnel LireRationnel(char *str);
Rationnel Simplifie(Rationnel);
Rationnel SommeRationnel(Rationnel, Rationnel);
void      AfficheRationnel(Rationnel);

int main(void) {
    Rationnel a = LireRationnel("1iere"), b = LireRationnel("2ieme");
    Rationnel c = SommeRationnel(a, b);

    printf("\nLa somme des deux rationnels est:\n\t\t");
    AfficheRationnel(a); printf("+ "); AfficheRationnel(b); printf("= "); AfficheRationnel(c)

    printf("\n\n");
    return 0;
}

Rationnel SommeRationnel(Rationnel x, Rationnel y) {
      Rationnel z = {x.num * y.denom + x.denom * y.num, x.denom * y.denom};
      return Simplifie(z);
}

void AfficheRationnel(Rationnel x) {
     printf("%d/%d ", x.num, x.denom);
}

Rationnel LireRationnel(char *str) {
    Rationnel x;

    printf("\nLire le %s rationnel :\n", str);

    printf("\tEntrez le numerateur (une entier) :"); scanf("%d", &x.num);

    do {
      printf("\tEntrez le denominateur (une entier != 0) :");
      scanf("%d", &x.denom);
      if (x.denom == 0)
        printf("\t\t <---- Le denominateur ( = %d) n'est pas valide ----> \n", x.denom);
    } while(x.denom == 0);

    return Simplifie(x);
}

Rationnel Simplifie(Rationnel x) {
    int diviseur=2;

    while ((x.num >= diviseur) && (x.denom >= diviseur)) {
        if ((x.num % diviseur == 0) && (x.denom % diviseur == 0)) {
           x.num /= diviseur;
           x.denom /= diviseur;
        }
        else
          diviseur++;
    }

    return  x;
}
```