Travaux dirigés - 2

1 Les fonctions et passage de paramètres

Exercice 1

Listing 1 – exo8.c

```
1  #include <stdio.h>
2
3  double f1(int k, double x);
4  double f2(int *k, double *x);
5
6  int main(){
7    int k=1;
8    double x=3.,y=0.;
9
10   y=f1(k,x);
11   printf("main: k=%2d, x=%.3lf, y=%.3lf\n",k,x,y);
12   y=f2(&k,&x);
13   printf("main: k=%2d, x=%.3lf, y=%.3lf\n",k,x,y);
14   return 1;
15 }
16
17 double f1(int k, double x){
18   k++;
19   x*=2.;
20   return k*x;
21 }
22
23 double f2(int *k, double *x){
24   (*k)++;
25   (*x)*=2.;
26   return (*k)*(*x);
27 }
```

Q. 1 *A l'aide de diagrammes de représentation mémoire, expliquer le déroulement du programme et donner les resultats affichés.*

Exercice 2

Listing 2 – exo9.c

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

void f1(int i, int *j, double *x, double y);

int main(){
  int i=1,j=2;
  double x=3.,y=M_PI;

  f1(i,&j,&x,y);
  printf("main: i=%6d, j=%6d, x=%.16lf, y=%.16lf\n",i,j,x,y);
  f1(j,&i,&y,x);
  printf("main: i=%6d, j=%6d, x=%.16lf, y=%.16lf\n",i,j,x,y);
  return 1;
}

void f1(int i, int *j, double *x, double y){
  (*j)++;
  i--;
  *x=y/2.;
  y*=2;
  printf("f1  : i=%6d, *j=%6d, *x=%.16lf, y=%.16lf\n",i,*j,*x,y);
}
```

Q. 1 *A l'aide de diagrammes de représentation mémoire, expliquer le déroulement du programme et donner les resultats affichés.*

Q. 2 *Donner les commandes Linux/Unix permettant d'exécuter le code précédent.*

Exercice 3

Listing 3 – exo.c

```c
#include <stdio.h>

void f(double *i, double j){
  *i=j+3./2;
  j=*i-1.;
}

void g(double x,double *y){
  *y=x+1.;
  x=*y-2.;
}

int main(){
  double s=1.,t=-1;
  f(&s,t);
  printf("s=%lf, t=%lf\n",s,t);
  g(s,&t);
  printf("s=%lf, t=%lf\n",s,t);
  return 0;
}
```

Q. 1 *Donner la ou les commandes Linux permettant de créer le programme executable* `exo` *associé au fichier* `exo.c`.

Q. 2 *A l'aide de diagrammes de représentation mémoire, expliquer le déroulement de ce programme et donner l'affichage.*

Exercice 4

Q. 1 *Soit $(x, y, z) \in \mathbb{R}^3$ et $k \in \mathbb{N}$. Nous voulons écrire une fonction modifiant les trois réels x, y, z en kx, ky, kz.*

1. *Donner l'ensemble des prototypes de fonction valides permettant de réaliser cette opération.*
2. *Quels sont les prototypes de fonction invalides ?.*
3. *Choisir un prototype valide et écrire la fonction associée.*

2 Les structures, fonctions et allocation dynamique

Exercice 5

Listing 4 – exo10.c

```c
#include <stdio.h>

typedef struct{
  double Re,Im;
} Complexe;

Complexe sum1(Complexe z1, Complexe z2); /* retoune z1+z2 ?*/
void sum2(Complexe z1, Complexe z2, Complexe z3); /* z3=z1+z2? */
void sum3(Complexe z1, Complexe z2, Complexe *z3); /* *z3=z1+z2? */

int main(){
  Complexe z1={1.0,0.0},z2={0.,1.},z3={0.,0.};
  z3=sum1(z1,z2);
  printf("z1=(%.2lf,%.2lf) - z2=(%.2lf,%.2lf) - z3=(%.2lf,%.2lf)\n",
         z1.Re,z1.Im,z2.Re,z2.Im,z3.Re,z3.Im);
  sum2(z1,z3,z3);
  printf("z1=(%.2lf,%.2lf) - z2=(%.2lf,%.2lf) - z3=(%.2lf,%.2lf)\n",
         z1.Re,z1.Im,z2.Re,z2.Im,z3.Re,z3.Im);
  sum3(z2,z3,&z3);
  printf("z1=(%.2lf,%.2lf) - z2=(%.2lf,%.2lf) - z3=(%.2lf,%.2lf)\n",
         z1.Re,z1.Im,z2.Re,z2.Im,z3.Re,z3.Im);
  return 1;
}

Complexe sum1(Complexe z1, Complexe z2){
  Complexe z3;
  z3=(Complexe){z1.Re+z2.Re,z1.Im+z2.Im};
  return z3;
}

void sum2(Complexe z1, Complexe z2, Complexe z3){
  z3.Re=z1.Re+z2.Re;z3.Im=z1.Im+z2.Im;
}

void sum3(Complexe z1, Complexe z2, Complexe *z3){
  z3->Re=z1.Re+z2.Re;z3->Im=z1.Im+z2.Im;
}
```

Q. 1 *A l'aide de diagrammes de représentation mémoire, expliquer le déroulement du programme et donner les résultats affichés.*

Exercice 6

Q. 1 *Ecrire la fonction `GetData` d'entrée des données a, b et N avec $a \in \mathbb{R}$, $b \in \mathbb{R}$ et $n \in \mathbb{N}^*$. Elles seront saisies au clavier par l'utilisateur.*

Q. 2 *Ecrire une fonction `InitMesh1D` permettant d'initialiser $h = (b-a)/N$ et un tableau alloué dynamiquement contenant l'ensemble des $x_i = a + ih,\ i \in \{0, \ldots, N\}$.*

Q. 3 *Ecrire un programme utilisant les deux fonctions précédantes et affichant le maillage ainsi généré. (désallocation ?)*

Q. 4 *En utilisant une structure Mesh1D, reprendre les questions précédantes.*

Exercice 7

Nous avons les structures suivantes :

```
typedef struct{
    int dim;
    double *val;
} vector;
```

Q. 1 *Nous voulons écrire une fonction* `AllocVector` *permettant d'allouer/initialiser un objet de type* `vector` *à la dimension N.*

1. *Donner l'ensemble des prototypes de fonction permettant de réaliser cette opération.*
2. *Quels sont les prototypes de fonction invalides ?.*
3. *Choisir un prototype et écrire la fonction associée.*

Q. 2 *Ecrire une fonction* `freeVector` *permettant de desallouer le champ* `val` *d'un objet de type* `vector`. *Justifier de ses arguments !*

Q. 3 *Nous voulons écrire une fonction* `FREEVector` *permettant de desallouer le champ* `val` *d'un objet de type* `vector`, *de mettre à zéro le champ* `dim` *et à* `NULL` *le champ* `val`.

1. *Donner l'ensemble des prototypes de fonction permettant de réaliser cette opération.*
2. *Quels sont les prototypes de fonction invalides ?.*
3. *Choisir un prototype et écrire la fonction associée.*

Q. 4 *Nous voulons écrire une fonction* `RandVector` *permettant d'initaliser chacune des composantes par un nombre aléatoire (utiliser la fonction drand48() du langage C :* `double drand48 (void);`*). Le vecteur est supposé déjà alloué.*

1. *Donner l'ensemble des prototypes de fonction permettant de réaliser cette opération.*
2. *Quels sont les prototypes de fonction invalides ?.*
3. *Choisir un prototype et écrire la fonction associée.*

Q. 5 *Soit* $v \in \mathbb{R}^n$, $u \in \mathbb{R}^n$, $\alpha \in \mathbb{R}$ *et* $\beta \in \mathbb{R}$.

1. *Ecrire l'ensemble des prototypes des fonctions valides permettant de calculer :*

   `ScalarProd` : $(u,v) = \sum_{i=1}^{n} u_i v_i$,

   `aUpbV` : $\alpha u + \beta v$

2. *Ecrire l'ensemble des prototypes invalides.*
3. *Pour chaque fonction, choisir un prototype valide et écrire la fonction associée.*

Q. 6 *Ecrire un ch'ti* `main` *utilisant les fonctions* `ScalarProd` *et* `aUpbV`.

Exercice 8

Nous avons les structures suivantes :

```
typedef struct{
    int dim;
    double *val;
} vector;

typedef struct{
    int nL; /* Nombre de lignes   */
    int nC; /* Nombre de colonnes */
    double **val;
} matrix;
```

Soit $\mathbb{A}$ une matrice réelle de taille $N \times M$ et `matrix A;` l'objet qui lui est associé en langage C. On souhaite avoir

$$\begin{aligned} \texttt{A.nL} &= N \\ \texttt{A.nC} &= M \\ \texttt{A.val[i-1][j-1]} &= \mathbb{A}_{i,j},\ \forall (i,j) \in \{1,\dots,N\} \times \{1,\dots,M\}. \end{aligned}$$

Q. 1 *Nous voulons écrire une fonction* `AllocMatrix` *permettant d'allouer/initialiser un objet de type* `matrix` *à la taille* $N \times M$.

1. *Donner l'ensemble des prototypes de fonction permettant de réaliser cette opération.*
2. *Quels sont les prototypes de fonction invalides ?.*
3. *Choisir un prototype et écrire la fonction associée.*

Q. 2 *Ecrire une fonction* `freeMatrix` *permettant de desallouer le champ* `val` *d'un objet de type* `matrix`. *Justifier de ses arguments !*

Q. 3 *Nous voulons écrire une fonction* `FREEMatrix` *permettant de desallouer le champ* `val` *d'un objet de type* `matrix`, *de mettre à zéro les champs* `nL` *et* `nV`, *et à* `NULL` *le champ* `val`.

1. *Donner l'ensemble des prototypes de fonction permettant de réaliser cette opération.*
2. *Quels sont les prototypes de fonction invalides ?.*
3. *Choisir un prototype et écrire la fonction associée.*

Q. 4 *Nous voulons écrire une fonction* `RandMatrix` *permettant d'initaliser chacune des composantes par un nombre aléatoire (utiliser la fonction drand48() du langage C :* `double drand48 (void);`*). La matrice est supposé déjà allouée.*

1. *Donner l'ensemble des prototypes de fonction permettant de réaliser cette opération.*
2. *Quels sont les prototypes de fonction invalides ?.*
3. *Choisir un prototype et écrire la fonction associée.*

Q. 5 *Soit* $\mathbb{A} \in \mathcal{M}n, p\mathbb{R}$, $\mathbb{B} \in \mathcal{M}p, m\mathbb{R}$ *et* $\mathbb{D} \in \mathcal{M}n, p\mathbb{R}$.

1. *Ecrire l'ensemble des prototypes des fonctions valides permettant de calculer :*
   `MatrixProd` : $\mathbb{Q} = \mathbb{AB}$,
   `MatrixSum` : $\alpha\mathbb{A} + \beta\mathbb{D}$
2. *Ecrire l'ensemble des prototypes invalides.*
3. *Pour chaque fonction, choisir un prototype valide et écrire la fonction associée.*

Q. 6 *Ecrire un ch'ti* `main` *utilisant les fonctions* `MatrixProd` *et* `MatrixSum`.

Exercice 9

On utilise les structures et fonctions des deux exercices précédants

Q. 1 *Soit* $\mathbb{A} \in \mathcal{M}n, n\mathbb{R}$, $v \in \mathbb{R}^n$ *et* $u \in \mathbb{R}^n$.

1. *Ecrire l'ensemble des prototypes des fonctions valides permettant de calculer :*
   `ProdMatVec` : $v = \mathbb{A}u$
   `ProdScalMat` : $\langle \mathbb{A}u, v \rangle$
2. *Ecrire l'ensemble des prototypes invalides.*
3. *Pour chaque fonction, choisir un prototype valide et écrire la fonction associée.*

Q. 2 *On fait de la compilation séparée.*

1. *Ecrire les fichiers header* `vector.h` *et* `matrix.h` *associés respectivement aux fichiers* `vector.c` *et* `matrix.c`.
2. *Ecrire un main (fichier* `prog.c`*) utilisant les fonctions* `ProdMatVec` *et* `ProdScalMat`.
3. *Expliquer comment créer l'executable (commandes gcc).*
4. *Ecrire un makefile.*