

# TP1 SR01: programmation C

## DOAN Nhat-Minh TC04

\*les codes sont programmés avec Sublime Text 3 (MinGW) sur windows

### Exercice 1

#### 1. Déclarer une structure Etudiant

```
typedef struct Etudiant
{
    int idEtu;
    char nom[30];
    char prenom[30];
    float moyenne;
}Etudiant;
```

#### 2. A partir des données stockées dans le fichier, récupérer l'identifiant, le nom et le prénom de chaque étudiant puis calculer sa moyenne

Dans cet exercice, on considère des cas suivant :

- + On peut choisir quel(s) fichier(s) à ouvrir par exemple on n'a besoin que les moyennes des deux UV parmi 3 de chaque étudiant
- + Si un fichier est vide on va afficher une message d'erreur

#### Ouvrir un fichier

```
FILE* fichier = NULL;
fichier = fopen(nom_fichier, "r");
```

ici - nom\_fichier sont "LO21.txt", "NF16.txt", "SR01.txt"

- r pour lire seulement

S'il n'y a pas encore le fichier, le programme va en créer un.

#### Fermer un fichier

```
fclose(fichier);
```

Lire un fichier

fscanf: lit une chaîne formatée

Ecrire dans un fichier

fprintf: écrit une chaîne « formatée » dans le fichier

### 3. Produire un fichier en sortie (Fichier\_final.txt)

Ici dans le fichier en sortie, on a considéré le cas : la moyenne de 3 UVs de chaque étudiant par l'ordre décroissant

Fichier_final.txt - Bloc-notes			
Fichier Edition Format Affichage ?			
Nb_etudiant=35			
99	Roger	Ebba	17.500
20	Mas	Abella	16.000
34	Navarro	Fannie	15.500
85	Chabba	Dalila	15.000
25	Martin	Callie	14.750
15	Richard	Cadfan	14.500
45	Pastor	Ober	14.500
48	Olive	Walter	14.500
72	Rolland	Daisy	14.500
78	Ribes	Eadwin	14.500
56	Comes	Fanchon	14.000
46	Rodriguez	Kaline	13.750
1	Garcia	Aaricia	13.500
13	Gil	Baptiste	13.500
63	Fabre	Fanélie	13.500
24	Garrigue	Dahlia	13.000
5	Blanc	Aaron	12.750
19	Salvat	Caleb	12.750
7	Sola	Balbine	12.500
12	Torrès	Baldwin	12.500
8	Serra	Abbon	12.250
22	Arnaud	Calixtine	11.750
16	Sanchez	Cadroe	11.500
18	Canal	Caline	11.500
10	Guisset	Abby	11.000
14	Simon	Cainnech	11.000
44	Bernard	Oanell	11.000
6	Pla	Babet	10.750
17	Esteve	Caïssa	10.500
50	Diaz	Zelda	10.500
23	Carrère	Daen	10.000
9	Lopez	Baldassare	9.750
21	Costa	Calimero	9.750
35	Fernandez	Eadbert	7.750
11	Martinez	Abdel	5.500

## Exercice 2

### 1. Affichage:

- Initialisation de la matrice

```
char m[LIGNE][COLONNE]={0},{0};
```

m[0][0]=' ';	m[1][0]=' ';	m[2][0]=' ';	m[3][0]=' ';	m[4][0]=' ';
m[0][1]='+';	m[1][1]='+';	m[2][1]='+';	m[3][1]=' ';	m[4][1]='+';
m[0][2]='+';	m[1][2]=' ';	m[2][2]='+';	m[3][2]=' ';	m[4][2]='+';
m[0][3]='+';	m[1][3]=' ';	m[2][3]='+';	m[3][3]='+';	m[4][3]='+';
m[0][4]=' ';	m[1][4]=' ';	m[2][4]=' ';	m[3][4]=' ';	m[4][4]=' ';
m[0][5]='+';	m[1][5]='+';	m[2][5]='+';	m[3][5]='+';	m[4][5]='+';
m[0][6]='+';	m[1][6]=' ';	m[2][6]='+';	m[3][6]='+';	m[4][6]=' ';
m[0][7]='+';	m[1][7]='+';	m[2][7]='+';	m[3][7]=' ';	m[4][7]='+';
m[0][8]=' ';	m[1][8]=' ';	m[2][8]=' ';	m[3][8]=' ';	m[4][8]=' ';
m[0][9]=' ';	m[1][9]='+';	m[2][9]='+';	m[3][9]='+';	m[4][9]=' ';
m[0][10]='+';	m[1][10]=' ';	m[2][10]=' ';	m[3][10]=' ';	m[4][10]='+';
m[0][11]=' ';	m[1][11]='+';	m[2][11]='+';	m[3][11]='+';	m[4][11]=' ';
m[0][12]=' ';	m[1][12]=' ';	m[2][12]=' ';	m[3][12]=' ';	m[4][12]=' ';
m[0][13]=' ';	m[1][13]='+';	m[2][13]=' ';	m[3][13]=' ';	m[4][13]=' ';
m[0][14]='+';	m[1][14]='+';	m[2][14]='+';	m[3][14]='+';	m[4][14]='+';
m[0][15]=' ';	m[1][15]=' ';	m[2][15]=' ';	m[3][15]=' ';	m[4][15]=' ';

- Affichage

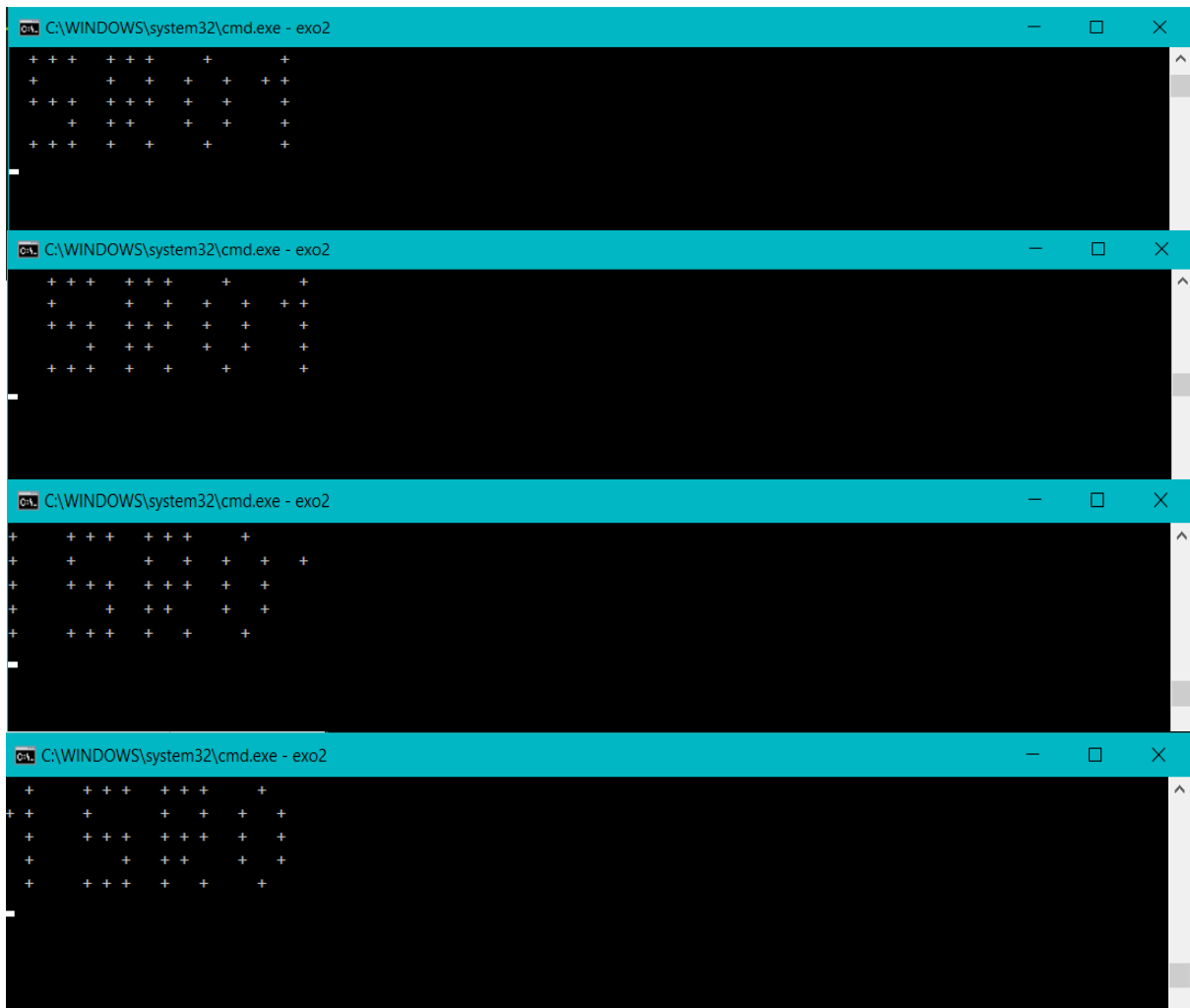
```
void affichage(char m[LIGNE][COLONNE])
{
    for (int i = 0; i < LIGNE; ++i)
    {
        for (int j = 0; j < COLONNE; ++j)
            printf("%c ",m[i][j]);
        printf("\n");
    }
}
```

## 2. Animation:

Dans la partie de l'animation, on utilise une nouvelle matrice. Deux matrices présentent une image de SR01 à tour de rôle

```
void animation(char a[LIGNE][COLONNE],char m[LIGNE][COLONNE])
{
    for (int i = 0; i < LIGNE; ++i)
    {
        for (int j = 0; j < COLONNE; ++j)
            if (j-1<0)
                a[i][j]=m[i][15];
            else
                a[i][j]=m[i][j-1];
    }
}
```

```
int run=1;
while (run==1)
{
    animation(a,m);
    affichage(a);
    usleep(500000);
    system("clear");
    animation(m,a);
    affichage(m);
    usleep(500000);
    system("clear");
}
```



## Exercise 3

1.

```
1  #include <stdio.h>
2  int main () {
3      int A=20 , B=5;
4      int C=!--A/++!B;
5      printf ("A=%d B=%d c=%d \n", A,B,C) ;
6  }
7  |
```

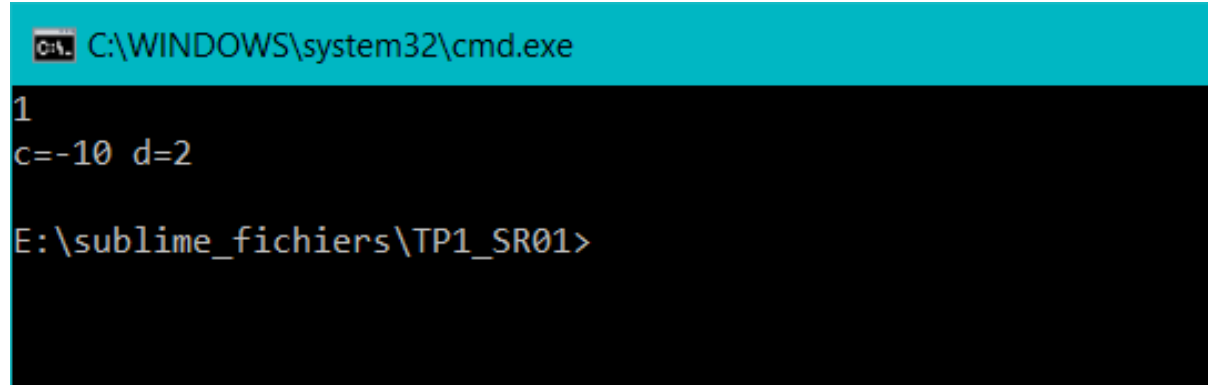
```
exo3.c: In function 'main':
exo3.c:4:13: error: lvalue required as increment operand
    int C=!--A/++!B;
              ^~
[Finished in 0.2s with exit code 1]
```

## Résultat

« lvalue » c'est-à-dire il n'y a pas la variable pour que l'opérateur change sa valeur. Ici on ne peut pas changer la valeur de !B qui vaut 0.

2.

```
8  #include <stdio.h>
9  int main () {
10     int A=20 , B=5 , C= -10 , D =2;
11     printf ("%d \n", A&&B||!0&&C++&&!D++) ;
12
13     printf ("c=%d d=%d \n", C, D) ;
14 }
15
```



## Résultat

Car A&&B est vrai, il n'est donc pas besoin de vérifier la suite après ||, le système ne calcule pas C++ et D++.

Donc le résultat de la 1<sup>ère</sup> printf donne 1.

Quand on sort de la 1<sup>ère</sup>, C et D restent leur valeurs au début -10 et 2

3.

```
19 # include <stdio.h>
20 int main () {
21     int p[4]={1 , -2 ,3 ,4};
22     int *q=p;
23     printf ("c=%d\n", *++q**q++) ;
24     printf ("c=%d \n" ,*q);
25 }
26
```

C:\WINDOWS\system32\cmd.exe

c=4  
c=3

E:\sublime\_fichiers\TP1\_SR01>\_

### Résultat

- d'abord on calcule ++q, ici on prend (q+1), q++, on garde (q+1) et retourne q+2 après le calcul
- donc on a  $*(q+1)**(q+1)=-2*-2=4$  et  $*(q+2)=3$

4.

```
27 # include <stdio.h>
28 int main () {
29     int p [4]={1 , -2 ,3 ,4};
30     int *q=p;
31     int d=*q&*q++|*q++;
32     printf ("d=%d\n", d) ;
33     printf ("q=%d \n" ,*q);
34 }
```

```
C:\WINDOWS\system32\cmd.exe
d=-1
q=3
E:\sublime_fichiers\TP1_SR01>
```

### Résultat

- $*q \& *q++ | *q++ = p[0] \& p[0] | p[1]$  et  $*q = *(q+2) = p[2]$  après le calcul  
 $= 1 \& 1 | -2$   
 $= 1 | -2$

Dans 8 bits :

1	=	00000001 <sub>2</sub>
2	=	00000010 <sub>2</sub>
-2	=	11111110 <sub>2</sub>

Donc  $1 | -2$  vaut  $11111111 = -1$

- $*q = p[2] = 3$



5.

```
36 #include <stdio.h>
37 int main () {
38     int a= -8 ,b =3;
39     int c= ++a&&--b?b--:a++;
40     printf ("a=%d b=%d c=%d\n",a,b,c) ;
41 }
42
```

C:\WINDOWS\system32\cmd.exe

a=-7 b=1 c=2

E:\sublime\_fichiers\TP1\_SR01>

### Résultat

c= ++a&&--b?b--:a++

- c vaut b--si (++a&&--b) est vraie vaut a-- sinon

- ici

++a : on prend a+1 =-7

--b : on prend b-1=2

-7&&2 est vrai donc c vaut b--

- On garde b=2 (c=2) et rend b=b-1=1 après le calcul

- a vaut -7

6.

```
43 #include <stdio.h>
44 int main () {
45     int a= -8 ,b =3;
46     a >>=2^b;
47     printf ("a=%d\n",a) ;
48 }
49
```

C:\WINDOWS\system32\cmd.exe

a=-4

E:\sublime\_fichiers\TP1\_SR01>

### Résultat

- le système calcule  $2^3$  :

00000010^00000011 (bitwise xor)

=>00000001<sub>2</sub>=1

- a = a >> 1

8 = 00001000

-8 = 11111000

-8>>1 = 11111100 (toujours négatif) = -4

Complément à 2 : 00000011

Plus un : 00000100 = 4<sub>10</sub>