Université Internationale de Casablanca LAUREATE INTERNATIONAL UNIVERSITIES Nous innovons pour votre réussite !	**Ecole d'Ingénierie** **Filières :** CPI & MIAGE **Classe :** 2ème année	**Cours : Programmation Structurée 2** **Professeur** : MOUJAHID Abdallah **Date** : 15/01/2018

Examen Final

Durée : 2 heures

Etudiant :	
Note :	

Notes Importantes :

- *Aucun document autorisé. Sont interdits tous les calculatrices, les téléphones, ainsi que tout autre outil de calcul et/ou de communication.*
- *Vous devez aussi remettre à votre professeur cet imprimé, portant votre nom, (Un étudiant qui n'a pas remis l'imprimé n'aura pas de note)*
- *TOUTE sortie est définitive !*
- *La propreté, la clarté et la qualité de rédaction seront pris en considération dans la notation*
- *TOUTE tentative de fraude sera sanctionnée selon la procédure en vigueur*

Partie I : QCM *(10,5 points)*

Répondez en entourant la/les lettre(s) correspondant(s) à la/les bonne(s) réponse(s).
+0,75 pour une bonne réponse, 0 pour absence de réponse, **-0,25** pour une mauvaise réponse.

1. **Quel sera le résultat du code suivant ?**

```
#include <stdio.h>
int main()
{
        int i;
        int *ptr = &i;
        printf("%x\n", &*ptr);
        return 0;
}
```

   a. ne compile pas
   b. provoque une erreur à l'exécution
   c. affiche l'adresse de i
   d. affiche l'adresse de ptr

2. **Quel sera le résultat du code suivant ?**

```
#include <stdio.h>

void trio(int *a, int *b, int *c)
{
        *a=*b+*c;
        *b=*c+*a;
        *c=*a+*b;
}

int main(void)
{
        int a=1, b=2, c=3;
        trio(&a,&b,&c);
        printf("%d %d %d\n",a,b,c);
        return 0;
}
```

   a. ne compile pas
   b. provoque une erreur à l'exécution
   c. affiche 5 4 3
   d. affiche 5 8 13
   e. affiche 1 2 3
   f. affiche 3 6 9

g. affiche 1 3 6

3. **Laquelle de ces valeurs est la plus grande ?**
   a. sizeof(int *)
   b. sizeof(char *)
   c. sizeof(double *)
   d. les 3 ont la même taille.

4. **Si x a été alloué avec :** *int *x = malloc(42 * sizeof(int));* **alors, pour libérer la mémoire pointée par x, on peut :**
   A. Atteindre la fin de la portée de la variable x, la libération sera automatique.
   B. Exécuter for (int i = 0; i < 42; i++) free(x + i);
   C. Exécuter free(x);
   D. Exécuter free(x, 42 * sizeof(int));
   E. Rien, la zone pointée par x sera désalloué toute seule quand elle ne sera plus utilisée.

5. **Quel sera le résultat du code suivant ?**

```
#include <stdio.h>
#define TAB_LENGTH 3

int main() {
        int tab[TAB_LENGTH];
        int j = 0;
        int *ptr = &tab[3];
        for(; j < TAB_LENGTH; j++)
                tab[j] = 5;
        *(tab + 1) = 3;
        *(ptr - 1) = 3;
        printf("[ %d %d %d ]", tab[0], tab[1], tab[2]);
        return 0;
}
```

   a. ne compile pas
   b. provoque une erreur fatale à l'exécution
   c. boucle infinie
   d. affiche [3 5 3]
   e. affiche [3 3 5]
   f. affiche [5 3 3]
   g. affiche [5 3 5]

6. **Si ptr a été déclaré sous la forme int *ptr, un seul des groupes d'affirmations suivantes est vrai, lequel ?**

a.	• l'expression *(ptr+1)==ptr[1] est syntaxiquement invalide • l'expression ptr[2] est une valeur entière • l'expression &ptr est une adresse en mémoire
b.	• l'expression *(ptr+1)==ptr[1] est toujours vraie • l'expression ptr-- est syntaxiquement invalide
c.	• l'expression *(ptr + 1) est une valeur entière • l'expression ptr[1] est une valeur entière • l'expression ptr est une adresse en mémoire

7. **Que ce passe-t-il si l'on inverse les lignes 3 et 4 du code de la fonction f ci-dessous ?**

```
1 void f(type* V) {
2   if (V != NULL) {
3     printf("%d\n",V->val);
4     f(V->next);
5   }
6 }
```

a. la fonction fera une boucle infinie,
b. l'affichage se fera en sens inverse,
c. la fonction n'affichera plus rien,
d. rien ne sera changé.

8. **Comment initialiser un pointeur sur un tableau de 10 char ?**
   a. p = malloc (10);
   b. p = (char *) malloc(10,1);
   c. p = (char *) malloc (sizeof(10*char));
   d. p = (char *) malloc(10 * sizeof(char));

9. **Lequel de ses prototypes de fonction ne permet pas de faire passer un tableau ?**
   a. void Mafonction (int tableau[], int taille);
   b. void Mafonction (int tableau, int taille);
   c. void Mafonction (int * tableau, int taille);
   d. Aucune des propositions ci-dessus.

10. **Comment affiche-t-on le champ 'nom' de la variable 'test' dans l'exemple suivant:** ***struct { char* nom; } test;***

a. printf("%s", test.nom);
b. printf("%s", test->nom);
c. printf("%s", *test.nom);

d. printf("%s", &test->nom);

11. Quel sera le résultat du code suivant ?

```
int test(int x)
{
    ++x;
    printf("%d ",x);
}
main()
{
    int a=1;
    test(a++);
    printf("%d",a);
}
```

a. 2 2
b. 2 1
c. 1 2
d. 1 1
e. Erreur de compilation

12. Que manque-t-il à la définition de structure suivante pour qu'elle soit syntaxiquement correcte ?

```
struct node{
    int val;
    node *left, *right;
};
```

a. un typedef
b. un ;
c. un struct
d. rien, elle est correcte

13. Quel est le code qui alloue dynamiquement de la mémoire correctement pour un tableau à deux dimensions (le code sera mis dans la ligne indiquée par le commentaire)?

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

int main()
{
```

```
int *p, i, j;
/* Ajouter le code ici*/
for(i=0; i<3; i++)
{
   for(j=0; j<4; j++)
   {
      p[i*4+j] = i;
      printf("%d", p[i*4+j]);
   }
}
return 0;
}
```

a. p = (int*) malloc(3, 4);
b. p = (int*) malloc(3*sizeof(int));
c. p = malloc(3*4*sizeof(int));
d. p = (int*) malloc(3*4*sizeof(int));

14. Quel sera le résultat de l'exécution de ce programme :

```
Void main ( ) {
Char a[] = "INFO" ;
a + +;
printf (" \n %s", a); }
```

a. Le code génère une erreur
b. Le code affiche « INFO »
c. Le code affiche « NFO »
d. Le code affiche « INF »
e. Aucune des propositions ci-dessous.

Partie II : Questions Directes *(3,5 points)*

1. ***(1,5 pts)*** Quelles sont les valeurs des entiers a,b et c à la fin de l'exécution du programme ?

```
#include <stdio.h>
int a=4;
int f(int * p)
{
  int a=5;
  *p=12*a;
  return *p*g(a);
```

```
}
int b=6;

int g(int a)
{
  int b;
  b=3*a;
  return b;
}
int main()
{
  int b=2;
  int c=3;
  c=f(&b);
  printf("%d %d %d\n",a,b,c);
}
```

……
……
……
……

2. ***(2 pts)*** Il y a 4 erreurs dans ce programme. Indiquez quelles sont les lignes qui sont incorrectes et proposez une correction.

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

main() {
  int i;
  int* p=NULL;
  int t1[20], t2[20], t3[20], t4[];

  for(i=0; i<20; i++) t1[i]=i;
  for(i=0; i<20; i++) p[i]=i*i;
  t3 = t2;
  if (t2 == t1)
    printf("les valeurs des tableaux sont identiques");
else
   printf("les valeurs des tableaux sont differents");
p = t2;
for (p=t2; p<t2+20; p++)
   printf("%d ",*p);
}
```

……
……
……
……
……
……

Partie III : Exercice de programmation *(6 points)*

On désire créer un programme pour gérer notre collection de disques. On suppose que l'on a, au maximum, 250 CD.

Voici les informations qui se rapportent à chaque disque :

- un entier qui est le numéro de l'album.
- une chaîne de 30 caractères qui représente le titre de l'album.
- une chaîne de 20 caractères qui représente l'artiste de l'album.
- un entier qui indique le nombre de chansons sur l'album.
- un tableau de 20 entiers contenant le numéro de chaque chanson sur l'album.
- un réel qui représente la durée totale en minutes de l'album.

1. ***(1 pt)*** Proposer une Structure « **CD** » pouvant **représenter** les informations ci-dessous.

2. ***(1 pt)*** Ecrire le code de la fonction SaisirTabCD permettant de **remplir** et **renvoyer** un tableau de **N** « CD ».

3. ***(1,5 pts)*** Proposer une fonction getAlbumsByArtist permettant d'**afficher** les albums d'un artiste donné (il sera question de passer un tableau de N « CD » et le nom de l'artiste entant que paramètres).

4. ***(1,5 pts)*** Ecrire le code de la fonction getLongestAlbum qui **renvoie** l'album le plus long (durée maximale) à partir d'une collection de « CD ».

5. ***(1 pt)*** Proposer une fonction getAverageSongs qui **renvoie** le nombre moyen des chansons contenues dans une collection de « CD ».