Université Internationale de Casablanca LAUREATE INTERNATIONAL UNIVERSITIES Nous innovons pour votre réussite !	**Ecole d'Ingénierie** **Filières :** MIAGE **Classe :** 2ème année – S2	**Cours : Programmation Structurée 2** **Professeur** : MOUJAHID Abdallah **Date** : 12/10/2018

DEVOIR SURVEILLE N° 1

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*

Université Internationale de Casablanca LAUREATE INTERNATIONAL UNIVERSITIES Nous innovons pour votre réussite !	**Ecole d'Ingénierie** **Filières :** MIAGE **Classe :** 2ème année – S2	**Cours : Programmation Structurée 2** **Professeur** : MOUJAHID Abdallah **Date** : 12/10/2018

Partie I : QCM *(11,25 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. Laquelle des fonctions suivantes retourne le nombre de lettres de la chaîne de caractères passée en argument (Cochez la bonne réponse) ?

```
1 int strlen(char* m) {
2   int i;
3   while (m[i] == 0) {
4     i = i + 1;
5   }
6   return i;
7 }
```

```
1 int strlen(char* m) {
2   int i = 1;
3   while (i > 0) {
4     i = m[i];
5   }
6   return i;
7 }
```

```
1 int strlen(char* m) {
2   int i = 1;
3   while (i > 0) {
4     if (m[i] = 0) { i = 0;}
5   }
6   return m[i];
7 }
```

```
1 int strlen(char* m) {
2   int i = 0;
3   while (m[i] != 0) {
4     i++;
5   }
6   return i;
7 }
```

2. 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);

3. Indiquer quel est l'appel correct pour la fonction suivante :

```
void sp(int * i , double j) ;
int a ;
double b ;
```

   a. sp(a, b);
   b. sp(a, &b);
   c. sp(&a, &b);
   d. sp(&a, &b);
   e. sp(∗a, b);

4. Donner le résultat de l'exécution du code suivant :

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

```
int main(){
   int i = 4;
   int j = 2;
   int *ptr1, *ptr2;
   ptr1 = &i;
   ptr2 = &j;
   printf("%d \n", *ptr1/*ptr2);
   return 0;
}
```

a. ne compile pas
b. provoque une erreur à l'exécution
c. affiche 0 à l'exécution
d. affiche 2 à l'exécution
e. Aucune des réponses

5. Donner le résultat de l'exécution du code suivant :

```
#include <stdio.h>
int main() {
   int a, b;
   int *ptr1, *ptr2;
   a = 10;
   b = a+2;
   ptr1 = &b;
   ptr2 = ptr1;
   b = (*ptr2)++;
   printf("a = %d, b = %d,*ptr1 = %d,*ptr2 = %d\n", a, b,*ptr1,*ptr2);
   return 0;
}
```

a. ne compile pas
b. provoque une erreur à l'exécution
c. affiche a = 11, b = 10, *ptr1 = 11, *ptr2 = 11
d. affiche a = 10, b = 12, *ptr1 = 12, *ptr2 = 12
e. affiche a = 12, b = 10, *ptr1 = 10, *ptr2 = 12
f. Aucune des propositions ci-dessus.

6. Ce programme a un défaut. Mais lequel ?

```
main(){
      char ville[100];
      printf("Dans quelle ville habitez-vous ? ");
      scanf("%s", &ville);
      printf("Vous habitez %s, je connais bien cette ville !", ville);
      }
```

   a. Il manque un & devant la variable "ville" dans le printf
   b. Il manque une * devant la variable "ville" dans la déclaration de la variable.
   c. Il y a un & en trop devant "ville" dans le scanf.
   d. Le programme ne contient aucune erreur.

7. Donner le résultat de l'exécution du code suivant :

```
#include <stdio.h>
#define TAB_LENGTH 5
int main() {
    int tab[TAB_LENGTH];
    int j = 1;
    int *ptr = tab + 2;
    for(; j < TAB_LENGTH; j++)
    {
        tab[j] = 4;
        *(ptr - 1) = 2;
    }
    printf("[ %d %d %d ]\n", tab[1], tab[2], tab[3]);
    return 0;
}
```

   a. ne compile pas
   b. provoque une erreur fatale à l'exécution
   c. affiche [2 4 4]
   d. affiche [2 2 4 4]
   e. affiche [4 4 2]
   f. Aucune des propositions ci-dessus.

8. Considérons le prototype de la fonction suivante: ***void Fiche(float *X, float *Y,int i,char Z,char R)*** ainsi que les déclarations suivantes :

   float A, C ;
   int J ;
   char B, H ;

   Quels sont les appels justes de la fonction Fiche?

   a. Fiche (A,C ; J ; B, H) ;
   b. Fiche (&A, &B, C, J, H) ;
   c. Fiche (&A,&C, 3, 'b', B) ;
   d. Fiche (&A, &C, J, B, H) ;
   e. Fiche (A ; J ; B, H) ;

9. Considérons le fragment de code ci-dessous :

```
int tab[]={ 10, 20, 30, 40 };
int *ptr1=&tab[1];
int *ptr2=&tab[3];
```

   Dans ce code, une seule des affirmations suivantes est vraie, laquelle ?

   a. les expressions *(ptr1-1) et *(ptr2-3) retournent toutes les deux la même valeur, 10
   b. l'expression ptr2-ptr1 vaut 20
   c. l'expression *(ptr2-ptr1) retourne la valeur 20
   d. Ce code génère une erreur

10. Donner le résultat de l'exécution du code suivant :

```
#include <stdio.h>
int x = 6;
void ecrire(int x) {
      printf("%d ", x += 2);
}
int main() {
      ecrire(x++);
      printf("%d\n", x);
      return 0;
}
```

    a. affiche 9 8 à l'exécution
    b. provoque une erreur à l'exécution
    c. affiche 7 8 à l'exécution

d. affiche 8 7 à l'exécution
e. affiche 7 7 à l'exécution

11. Donner le résultat de l'exécution du code suivant :

```
void fonction(int a[]) {
   a[1] = 10;
}
int main(void) {
      int T[]={1,2,3};
      fonction(T);
      printf("%d", T[1] );
      return 0;
}
```

a. 10
b. 1
c. 2
d. Le programme contient une erreur

12. Quel est l'effet du programme suivant ?

```
#include <stdio.h>
void f(int *p, int *q, int *r){
      int a, *b;
      b = q; q = r; r = b;
      a = *p; *p = *q; *q = a;
}
int main(){
      int x = 3, y = 5, z = 7;
      f(&x, &y, &z);
      printf("%d %d %d", x, y, z);
}
```

a. erreur à la compilation
b. affiche 735
c. erreur à l'exécution
d. affiche 753
e. affiche 537

13. Soit la déclaration suivante : **`char *p="ma chaîne";`** **Laquelle ou lesquelles** de ces propositions sont correctes (sélectionner toutes les bonnes réponses):
    a. p pointe sur un tableau de 10 char.
    b. Cette instruction provoque une erreur d'exécution.
    c. free(p) permet de libérer la mémoire allouée à p.
    d. p pointe sur le caractère m.
    e. On ne peut pas modifier la chaîne de caractères pointée par p.
    f. On peut modifier la chaîne de caractères pointée par p.

14. Après les instructions :

```
int tab[] = {0, 10, 100, 1000};
int *t = &tab[1] ;
int *p;
```

que donnera les instructions:

```
p = &(t[1] + 1);
printf("%d", *p);
```

    a. affiche 10
    b. affiche 18
    c. affiche 19
    d. erreur de compilation
    e. boucle infinie

15. Soit les déclarations suivantes :
    int i=3;
    int * pi=&i;
    Laquelle ou lesquelles de ces propositions sont **TOUJOURS** correctes :

    a. La deuxième ligne provoque une erreur de segmentation.
    b. *pi vaut 3.
    c. si on réalise l'instruction suivante : *(pi)=5 alors on ne modifie pas i.
    d. si on réalise l'instruction suivante : pi[0]=5 alors la valeur de i change.
    e. si on réalise l'instruction suivante : pi=5 alors on ne modifie pas i.

Partie II : Exercices de programmation *(8,75 points)*

Exercice 1 – Les fonctions – Nombre amis *(3,25 pts)*

1. ***(2 pts)*** Ecrire la fonction ***float indiceVal(float val, float tab[], int nb)*** qui recherche la valeur val dans le tableau tab. Cette fonction retourne l'indice du premier élément du tableau égal à val, ou -1 si cette valeur n'existe pas dans le tableau.

2. ***(1,25 pts)*** Tester la fonction ***indiceVal*** dans une fonction main.

Exercice 2 – Les fonctions – Nombre amis *(5,5 pts)*

Deux nombres M et N sont appelés **nombres amis** si la somme des diviseurs de M est égale à N et la somme des diviseurs de N est égale à M.

1. ***(2 pts)*** Ecrire une fonction sommeDiv qui retourne la somme des diviseurs d'un nombre passé en paramètre.
2. ***(2 pts)*** Ecrire une fonction verifierAmis qui vérifie si deux entiers x et y sont des nombres amis. Cette fonction utilise la fonction sommeDiv de la première question.
3. ***(1,5 pt)*** Ecrire un programme principal qui permet de tester