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** : 02/12/2016

Devoir Surveillé 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 :** CPI & MIAGE **Classe :** 2ème année	**Cours : Programmation Structurée 2** **Professeur** : MOUJAHID Abdallah **Date** : 02/12/2016

Partie I : QCM *(9 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) ?

☐
```
int strlen(char* m) {
  int i;
  while (m[i] == 0) {
    i = i + 1;
  }
  return i;
}
```

☐
```
int strlen(char* m) {
  int i = 1;
  while (i > 0) {
    i = m[i];
  }
  return i;
}
```

☐
```
int strlen(char* m) {
  int i = 1;
  while (i > 0) {
    if (m[i] = 0) { i = 0;}
  }
  return m[i];
}
```

☐
```
int strlen(char* m) {
  int i = 0;
  while (m[i] != 0) {
    i++;
  }
  return i;
}
```

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

```
#include <stdio.h>
int main(){
   int j = 0;
   int *ptr;
   ptr = &j;
   printf("%d \n", *ptr+3);
   return 0;
}
```

   a. ne compile pas
   b. provoque une erreur à l'exécution
   c. affiche 0 à l'exécution
   d. affiche 3 à l'exécution

3. L'allocation dynamique de mémoire suit des étapes dans un ordre particulier, lequel ?
   a. malloc, utilisation de la mémoire, vérification allocation réussie, free
   b. malloc, utilisation de la mémoire, free, vérification allocation réussie

c. free, vérification de l'allocation réussie, malloc, utilisation de la mémoire
d. malloc, vérification allocation réussie, utilisation de la mémoire , free

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

```
#include <stdio.h>
int main() {
   int a, b;
   int *ptr1, *ptr2;
   a = 5;
   b = a;
   ptr1 = &a;
   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 = 6, b = 5, *ptr1 = 6, *ptr2 = 6
d. affiche a = 5, b = 5, *ptr1 = 5, *ptr2 = 5
e. affiche a = 6, b = 6, *ptr1 = 6, *ptr2 = 6
f. Aucune des propositions ci-dessus.

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

```
   int i = 0;
   int j = 0;
   for (i = 0; i < 2; i = i + 1)
   {
       for (j = 0; j < 3; j = j +
   1)
       {
           printf("%d ", j);
       }
   }
```

a. 0 1 2 0 1 2

b. 0 0 1 1 2 2 3
c. 0 1 2 0 1 2 3
d. 0 1 2 3 0 1 2
e. Aucune des propositions ci-dessus.

6. 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) ;

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

```
int main() {
int a, b;
int *ptr1, *ptr2;
a = 5;
b = a;
ptr1 = &a;
a++;
ptr2 = &b;
b += a = b;
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 (erreur de segmentation par exemple)
c. affiche a = 5, b = 10, *ptr1 = 5, *ptr2 = 10
d. afficheca = 5, b = 11, *ptr1 = 5, *ptr2 = 11
e. affiche a = 6, b = 6, *ptr1 = 6, *ptr2 = 11

f. Aucune des propositions ci-dessus.

8. 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.

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

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

a. ne compile pas
b. provoque une erreur fatale à l'exécution
c. affiche [5 5 5]
d. affiche [3 5 5]
e. affiche [5 3 5]
f. Aucune des propositions ci-dessus.

10. 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

11. Par laquelle des propositions suivantes faut-il remplacer la ligne .....dans le code ci-dessus pour être certain qu'il s'exécute correctement ?

```
1  int main( ) {
2    int i;
3    int* tab;
4    ........
5    tab[0] = 2;
6    for (i=1; i<=5; i++) {
7      tab[i] = tab[i-1]*2 + 5;
8    }
9    printf("%d\n", tab[5]);
10   free(tab);
11   return 0;
12 }
```

a. tab = (int*) malloc(5*sizeof(int*));
b. tab = (int*) malloc(6*sizeof(int));
c. tab = (int) malloc(sizeof(6*int));
d. tab = (int) calloc(5, sizeof(int));
e. Ce code contient des erreurs.

12. Combien d'int peut-on stocker au maximum dans un tableau tab alloué avec la commande : `int* tab = (int*) malloc(1000);?`

a. 1000,

b. 1000/sizeof(int)
c. 1000*sizeof(int)
d. On ne peut pas savoir

Partie II : Questions directes *(3 points)*

1. Qu'affiche le programme suivant ***(1 pt)*** :

```
#include <stdio.h>

void affiche_chaine(char *T){
   int i=0;
   while(T[i]!='\0')
      {
         printf("%c",T[i]);
         i++;
      }
}

int main()
{
   char chaine[]={'e','x','a','m'};
   char tab[]="chaine";

   affiche_chaine(tab);
   affiche_chaine(chaine);

   return 0;
}
```

……
……

2. Ecrivez le code pour définir une **structure LIVRE** ayant les informations suivantes ***(1 pt)*** :
   - Le titre du livre (200 caractères),
   - Le nom de l'auteur (100 caractères),
   - L'année de son édition.

3. Un programme en C contient la déclaration suivante ***(1 pt)***:

```
char *couleur[6]= {"rouge", "vert", "bleu", "blanc", "noir", "jaune"};
```

   a. Quelle est la valeur de **couleur[1]** ?

……

b. Quelle est la valeur de ***(couleur[3] +3)** ?

……

Partie III : Exercices de programmation *(8 points)*

Exercice 1 (4 points) - Tableaux & Allocations dynamiques de mémoire

Écrire programme qui :

- Créé deux tableaux tab1 de taille N1 et tab2 de taille N2 en optant pour l'allocation dynamique de mémoire (N1 et N2 sont définis par l'utilisateur).
- Rempli tab1 et tab2 avec des valeurs saisies par l'utilisateur.
- Agrandi la taille de tab2 de telle façon à y rajouter les éléments de tab1.
- Rajoute les éléments de le tab1 à la fin de tab2.
- Affiche tab2.
- Libère les mémoires allouées.

Exercice 2 (4 points) – Tableaux de pointeurs & chaines de caractères

Ecrire un programme qui lit 20 mots d'une longueur maximale de 50 caractères au clavier et qui les mémorise dans un tableau de pointeurs sur char en réservant dynamiquement l'emplacement en mémoire pour les mots.
Ensuite, le programme vérifie s'il y a des mots palindromes et les affiche.