Université Internationale de Casablanca LAUREATE INTERNATIONAL UNIVERSITIES Nous innovons pour votre réussite !	**Ecole d'Ingénierie** **Filières :** MIAGE **Classe :** 2ème année/S3	**Cours : Programmation Structurée 2** **Professeur** : MOUJAHID Abdallah

DEVOIR SURVEILLE

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 *(3,75 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. Quelle sera la sortie du code C suivant ?

```
#include <stdio.h>
struct student
{
   char a[5];
};
void main()
{
   struct student s[] = {"hi", "hey"};
   printf("%c", s[0].a[1]);
}
```

a. h
b. i
c. e
d. y

2. Quelle sera la sortie du code C suivant ?

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

int main()
{
   int *p;
   p = (int *)malloc(20);    /* On suppose que p a la valeur 1314 */
   free(p);
   printf("%ld", p);
   return 0;
}
```

a. 1314
b. Valeur aléatoire
c. 1316
d. Erreur d'éxecution

3. Quelle sera la sortie du code C suivant ?

```
#include <stdio.h>
  struct student
  {
    char *name;
  };
  struct student s;
  struct student fun(void)
  {
    s.name = "newton";
    printf("%s\n", s.name);
    s.name = "alan";
    return s;
  }
  void main()
  {
    struct student m = fun();
    printf("%s\n", m.name);
    m.name = "turing";
    printf("%s\n", s.name);
  }
```

a. newton alan alan
b. alan newton alan
c. alan alan newton
d. Erreur de compilation

4. Indiquez l'instruction correcte qui libère correctement la mémoire pointée par's' et'p' dans le programme suivant ?

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

int main()
{
  struct ex
  {
```

```
    int i;
    float j;
    char *s
  };
  struct ex *p;
  p = (struct ex *)malloc(sizeof(struct ex));
  p->s = (char*)malloc(20);
  return 0;
}
```

a. free(p); puis free(p->s);
b. free(p->s); puis free(p);
c. free(p->s);
d. free(p);

5. **Lequel des énoncés suivants assigne correctement 12 à la variable** month **à l'aide de la variable pointeur** *pdt* **?**

```
#include<stdio.h>


  struct date
  {
    int day;
    int month;
    int year;
  };
int main()
{
  struct date d;
  struct date *pdt;
  pdt = &d;
  return 0;
}
```

a. pdt.month = 12
b. &pdt.month = 12
c. d.month = 12
d. pdt->month = 12

Partie II : Questions Directes *(7 points)*

1. ***(1.5 pts)*** Complétez le programme ci-dessous afin de calculer la somme d'un maximum de 10 nombres (les nombres négatifs ne sont pas pris en compte dans le calcul) :

```
# include <stdio.h>
int main()
{
   int i, number, sum = 0;

   for(i=1; …… ; i++)
   {
      printf("saisir un entier : “);
      scanf("%d",&number);

      if(number < 0)
      {
         ……
      }

      ……
   }

   printf("Sum = %d",sum);

   return 0;
}
```

2. ***(2 pts)*** Qu'affiche le programme ci-dessous si l'utilisateur saisie n = 4 (pour le premier scanf) et les valeurs 10 20 30 40 pour le deuxième scanf ? donnez des explications en proposant éventuellement un schéma

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

void main(void)
{
   int n, i ;
   double* note ;
   double moyenne = 0. ;
   printf("\n Entrez le nombre de notes : ") ;
   scanf("%d", &n) ;
```

```
   note = (double *)malloc(n*sizeof(double)); /* affectation de note */

   if (note==NULL)
   {
      printf("\n Problème d'allocation !");
      exit() ;
   }

 for (i=0 ; i<n; i++)
   {
      printf("\n Tapez la note %d : ", i) ;
      scanf("%lf", note+i) ;
      moyenne = moyenne + note[i] ;
   }

    printf("\n moyenne = %lf", moyenne/n) ;
   free( note ) ;
}
```

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

3. ***(1,5 pts)*** Qu'affiche le programme ci-dessous ? donnez des explications en proposant éventuellement un schéma :

```
#include <stdio.h>
#define NB_OPTIONS 4

int main(void)
{
   int i ;
   char* menu[NB_OPTIONS] = {"Quitter", "Sauvegarder", "Save as", "Autre"};
```

```
   for (i=0 ; i<NB_OPTIONS ; i++)
      printf("\t %c", *(menu[i]+i)) ;

   return 0;
}
```

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

4. ***(2 pts)*** Qu'affiche le programme ci-dessous ? donnez des explications en proposant éventuellement un schéma

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

int main(void)
{
   int *p, *tmp, i;
   p = malloc(sizeof(int) * 10);

   if (p == NULL)
   {
      printf("Échec de l'allocation\n");
      exit(1);
   }

   for (i = 0; i < 10; ++i)
      p[i] = i * 10;

   tmp = realloc(p, sizeof(int) * 20);

   if (tmp == NULL)
   {
      printf("Échec de l'allocation\n");
```

```
    exit(1);
  }

  p = tmp;

  for (i = 10; i < 20; ++i)
    p[i] = i * 10;

  for (i = 0; i < 20; ++i)
    printf("p[%d] = %d\n", i, p[i]);

  free(p);
  return 0;
}
```

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Partie III : Exercices de programmation *(9,25 points)*

Exercice 1 – Structures, Fonctions & Tableaux de Structures *(6 pts)*

Soit la structure nommée **Salle** caractérisée par son **identificateur** (entier) et son **libellé** (chaine de caractère de max 20), et le champ **capacité** (entier) qui représente le nombre des étudiants maximal.

1. Créer la structure Salle. ***(0,75 pt)***
2. Définir la fonction Salle ***CreerSalle(int idSalle, char *libelleSalle, int Capacite)*** qui permet de créer une salle et de la retourner ***(1 pt).***
3. Définir la fonction ***void SaisirInfosSalle(Salle Liste[], int n*)** qui permet d'insérer n salles dans un tableau de stockage des salles passé en paramètre (utilisez la question précédente) ***(1 pt).***
4. Définir la fonction ***void AfficherSalle(Salle S)*** qui permet d'afficher les informations de la salle passé en paramètre sous la forme ***(0,75 pt):***

   Id Salle : …….. ; Libellé Salle :…….. ; Capacité Salle :……
5. Définir la procédure ***void AfficherInfosSalles(Salle Liste[], int n)*** permettant d'afficher les informations des salles utilisant la fonction de la question N° 4. ***(1 pt)***
6. Définir la fonction int RechercherSalle(int id, Salle Liste[],int n) qui permet de rechercher une salle par son identificateur passé en paramètre, le parcours doit s'arrêter une fois la salle est trouvée. ***(1,5 pts)***

Exercice 2 – Allocations dynamique de mémoire *(3.25 pts)*

1. ***(2,25 pts)*** Ecrire une fonction **remplirAlea** permettant de remplir aléatoirement un tableau **tab** de **nb** entiers avec des valeurs tirée au hasard dans l'intervalle [**0**, **max**[.
   NB : Pour des raisons d'optimisation, il faudra allouer la mémoire dynamiquement.

2. ***(2,25 pts)*** Ecrire un programme pour tester la fonction rempliAlea ***(1 pt)***