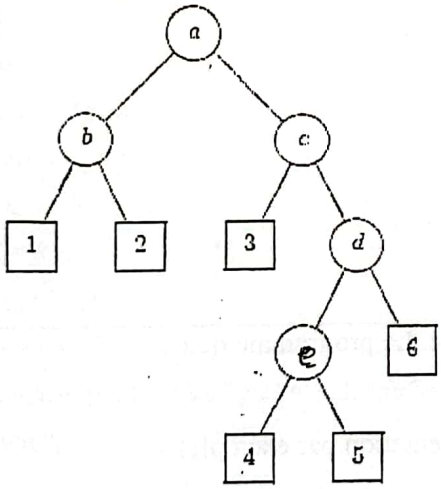
ENIG	République Tunisienne Ministère de l'Enseignement Supérieur et de la Recherche Scientifique Université de Gabès École Nationale d'Ingénieurs de Gabès	Réf : DE-EX-01
		Indice : 3
	EPREUVE D'EVALUATION	Date : 05/02/2021
		Page : 1/9

Année Universitaire : 2022/2023	**Date de l'Examen** : 05/01/2023
Nature : ☐ DC ☑ Examen ☐ DR	**Durée** : ☐ 1h ☐ 1h30min ☑ 2h
Diplôme : ☐ Mastère ☑ Ingénieur	**Nombre de pages** : 04
Section : ☐ GCP ☐ GCV ☐ GEA ☑ GCR ☐ GM	**Enseignant** : I. Azaiez & M-K. Abdmouleh
Niveau d'étude : ☑ 1ère ☐ 2ème ☐ 3ème année	**Documents Autorisés** : ☐ Oui ☑ Non
Matière : Programmation I	**Remarque** :

PARTIE 1 (8 points) :

Exercice (3 points) :

1) **Donner le résultat (Rendu) des parcours :**



a) **préfixé**

.....

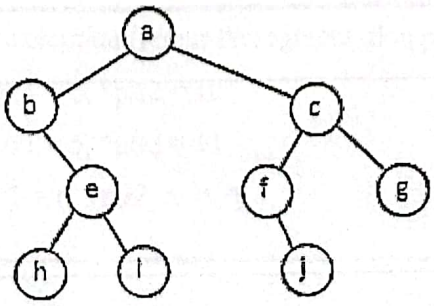
b) **infixé**

.....

c) **postfixé**

.....

2) **Représenter l'arbre dans un tableau.**



CamScanner

QCM (5 points) :

Q 1. Soit q1.c le code suivant :

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

On compile q1.c et si un exécutable est créé on l'exécute. Le programme q1.c :

1. ne compile pas
2. provoque une erreur à l'exécution (erreur de segmentation par exemple)
3. affiche 0 à l'exécution
4. affiche 1 à l'exécution

Q 2. Soit q2.c le code suivant :

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

On compile q2.c et si un exécutable est créé on l'exécute. Le programme q2.c :

1. ne compile pas
2. provoque une erreur à l'exécution (erreur de segmentation par exemple)
3. affiche l'adresse de i en hexadécimal
4. affiche l'adresse de ptr en hexadécimal

Q 3. Soit q3.c le code suivant :

```
#include <stdio.h>
int main() {
   int a, b;
   int *ptr1, *ptr2;
```


CamScanner

```
    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;
}
```

On compile q3.c et si un exécutable est créé on l'exécute. Le programme q3.c :

1. ne compile pas
2. provoque une erreur à l'exécution (erreur de segmentation par exemple)
3. affiche a = 6, b = 5, *ptr1 = 6, *ptr2 = 6
4. affiche a = 5, b = 5, *ptr1 = 5, *ptr2 = 5
5. affiche a = 6, b = 6, *ptr1 = 6, *ptr2 = 6

Q 4. Soit q4.c le code suivant :

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

On compile q4.c et si un exécutable est créé on l'exécute. Le programme q4.c :

1. ne compile pas
2. provoque une erreur à l'exécution (erreur de segmentation par exemple)
3. affiche a = 5, b = 10, *ptr1 = 5, *ptr2 = 10
4. affiche a = 5, b = 11, *ptr1 = 5, *ptr2 = 11
5. affiche a = 6, b = 6, *ptr1 = 6, *ptr2 = 11

CamScanner

Q 5. Soit q5.c le 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;
}
```

On compile q5.c et si un exécutable est créé on l'exécute. Le programme q5.c :

1. ne compile pas
2. provoque une erreur à l'exécution (erreur de segmentation par exemple)
3. affiche 5 4 3
4. affiche 5 8 13
5. affiche 1 2 3
6. affiche 3 6 9
7. affiche 1 3 6

Q 6. Soit q6.c le 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;
}
```


CamScanner

On compile q6.c et si un exécutable est créé on l'exécute. Le programme q6.c :

1. ne compile pas
2. provoque une erreur fatale à l'exécution (erreur de segmentation par exemple)
3. boucle
4. affiche [5 5 5]
5. affiche [3 5 5]
6. affiche [5 3 5]

Q 7. Soit q7.c le 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;
}
```

On compile q7.c et si un exécutable est créé on l'exécute. Le programme q7.c :

1. ne compile pas
2. provoque une erreur fatale à l'exécution (erreur de segmentation par exemple)
3. boucle
4. affiche [3 5 3]
5. affiche [3 3 5]
6. affiche [5 3 3]
7. affiche [5 3 5]

CamScanner

Q 8. Soit q8.c le code suivant :

```
#include <stdio.h>
struct list_v {
struct list_v *suivant;
};
typedef struct list_v *list;
int main() {
   list m;
   m->suivant = NULL;
   return 0;
}
```

On compile q8.c et si un exécutable est créé on l'exécute. Le programme q8.c :

1. ne compile pas
2. provoque une erreur fatale à l'exécution (erreur de segmentation par exemple)
3. boucle
4. fonctionne normalement mais n'affiche rien

Q 9. Soit q9.c le code suivant :

```
#include <stdio.h>
#include <stdlib.h>
struct list_couple {
   int membre1;
   int membre2;
   struct list_couple *suivant;
};
typedef struct list_couple *list;
int main() {
   list m = malloc(sizeof(struct list_couple));
   list m2 = malloc(sizeof(struct list_couple));
   m->suivant = m2;
   free(m);
   return 0;
}
```


CamScanner

On compile q9.c et si un exécutable est créé on l'exécute. Le programme q9.c :

1. ne compile pas
2. provoque une erreur à l'exécution (erreur de segmentation par exemple)
3. fonctionne et avant le retour de la fonction main() tout l'espace occupé par m et m2 a été libéré.
4. fonctionne mais avant le retour de la fonction main() tout l'espace occupé par m et m2 n'a pas été libéré.

Q 10. Soit q10.c le code suivant :

```
#include <stdio.h>
#include <stdlib.h>
struct list_couple {
   double membre1;
   double membre2;
   struct list_couple *suivant;
};
typedef struct list_couple *list;
int main() {
   list m = malloc(sizeof(struct list_couple));
   list m2 = malloc(sizeof(struct list_couple));
    m->membre1 = 18.5;
   m->membre2 = 23.4;
   m->suivant = m2;
   free(m->membre1);
   free(m->membre2);
   free(m2);
   free(m);
   return 0;
}
```

On compile q10.c et si un exécutable est créé on l'exécute. Le programme q10.c :

1. ne compile pas
2. fonctionne et avant le retour de la fonction main() tout l'espace occupé par m et m2 a été libéré
3. fonctionne mais avant le retour de la fonction main() tout l'espace occupé par m et m2 n'a pas été libéré.

CamScanner

PARTIE 2 : Manipuler un fichier binaire : (12 points) :

On caractérise un étudiant par son matricule (*entier*), son nom (*chaîne de 20 caractères au plus.*), son prénom (*chaîne de 20 caractères au plus*), et sa moyenne générale (*réel*). Un étudiant sera représenté et défini par le type **Etud**. On suppose que toutes les informations relatives à un ensemble d'étudiants sont sauvegardées dans ***le fichier binaire*** « ETUDIANT.DAT »

Ecrire un programme C permettant d'effectuer les opérations suivantes :

1. ***Créer*** la structure **Etud. (1 point)**
2. ***Créer*** le fichier binaire, initialement vide, d'étudiants « *ETUDIANT.DAT* ». **(1.5 point)**
3. ***Ajouter*** un nouvel étudiant (*saisi au clavier*) dans le fichier. **(1.5 point)**
4. ***Afficher*** les enregistrements de ce fichier. **(1.5 point)**
5. ***Recherche séquentielle*** : consulter dans ce fichier, la note d'un étudiant après avoir fourni le nom. **(1.5 point)**
6. ***Recherche directe*** : consulter les informations du nième étudiant du même fichier, après avoir fourni le numéro d'emplacement. **(1.5 point)**
7. ***Modifier*** la note du nième étudiant dans le fichier. **(1.5 point)**
8. Créer un menu de Test qui utilise les fonctions des questions n° : **2, 3, 4, 5, 6, 7 (2 points)**

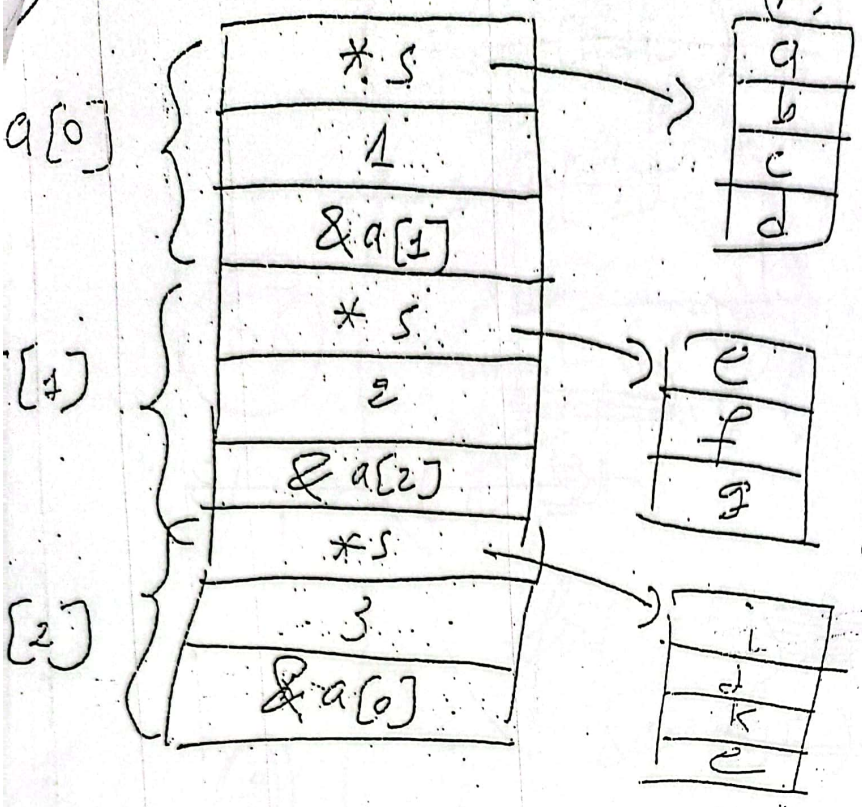
Exemple d'exécution :

---------- Menu ----------

1 : Creer Un Fichier Binaire ;
2 : Ajouter Un Nouvel Etudiant ;
3 : Afficher les enregistrements du fichier ;
4 : Recherche Sequentielle ;
5 : Recherche Directe ;
6 : Modifier la note d'un etudiant ;
0 : Quitter ;

Donnez le choix :

CamScanner



abcd abcd abcd

```
#define NMAX 100
void main()
{
    char t[NMAX], c;
    int n, i = 0;
    while (c = getchar() != '*') && (i < NMAX)
    {
        if ((c + 13 < 'Z') || (c + 13 < 'z'))
            t[i] = c + 13;
        else
            t[i] = c - 13;
        i++;
    }
    ...
}
```

n chose mais

ik

```
if(C+B>'Z')
    T[i] = C+B;
else
    C+B
    T2[i] = C+B;
```

ou

```
while(T[i]
for(i=0; i<N; i++)
    if(T[i]+B>'Z')
```

2 ou

```
    T[i] = T[i] - B;
else
    T[i] = T[i] + B;
```

```
if(T[i]+B>'Z')
{   C = T[i] - B;
    T2[i] = C;
}
else
{   C = T[i] + B;
    T2[i] = C;
}
```

void main() {

~~int~~ int tab[NMAX] = {0};

do {
printf("Donner la taille du tableau");
Scanf("%d", &N);
} while (N > NMAX);

~~for~~ printf("Donner les elements du tableau");

for(i=0; i<N; ++i)
Scanf("%d", &tab[i]);

Afficher(tab, N);

Tri(tab, N) (1)

Afficher(tab, N) (0,5)

(2)

(0,5)

} // fin main

// fin du programme

مكتبة المطالب
تصميم وتنسيق جميع الوثائق تسهيلا
وإصلاح الكتب - زريق
P 97 469 618.

EX2

```
#include <stdio.h>
#define NMAX
int tab[100];
```

(2)

```
void Afficher (int *t, int N)
{
    int i;
    for(i=0; i<N; i++)
        printf("%d", t[i]);
}
```

(0,5) (2) (1,5)

```
void permute (int *t, int i, int j)
{
    int aux;
    aux = t[i];
    t[i] = t[j];
    t[j] = aux;
}
```

(0,5) (2) (1,5)

```
void Tri (int *t, int N)
{
    int i, j, kmax;
    for (i=0; i<N-1; i++)
    {
        kmax = i;
        for (j=i+1; j<N; j++)
            if (t[j] > t[kmax])
                kmax = j;
        permute (t, i, kmax);
    }
}
```

(1) (2) (2) (1)

ENIG : Ecole Nationale d'Ingénieurs de Gabès

Devoir de Synthèse: *Programmation I* (GCR1)

Session Principale: 10 Janvier 2011

Durée : 02 H 00
Enseignant Responsable : Mohamed Ben Halima

Documents non autorisés

Soyez précis et concis, les copies ne seront pas notées en fonction de la quantité d'encre utilisée !

Exercice N°1: (1.5pts)

Soit le code suivant :

```
#include <stdio.h>
#define MAX = 10
void main() {
      int tab[MAX] ;
      tab[2] = 2 ;
      printf("%d", tab[MAX-8]) ;
      }
```

Ce programme compile-t-il? Pourquoi ? Si oui qu'affiche-t-il ?

Exercice N°2: (1.5pts)

Soit le code suivant :

```
void g(int *p) {
      *p = 12 ;
      }
void main() {
      int *p ;
      g(p) ;
      printf("%d\n", *p) ;
      }
```

Qu'est ce qui s'affiche ?

CamScanner

Exercice N°3: (17pts)

On suppose qu'un salarié est caractérisé par son nom, son prénom, son adresse, sa date de naissance, son grade et son salaire. La date de naissance est structurée en jour, mois et année. Le grade peut être "ouvrier", "secrétaire" ou "directeur". L'adresse est caractérisée par le numéro de local, la nomination de la rue, le code postale, la ville et le pays.

Première partie (07.50pts)

1. Définir les types Date, Grade, Adresse et Salarie. (2pts)
2. Ecrire la fonction *void Saisie (struct Salarie *S)* qui permet de saisir les informations d'un salarié. (1pt)
3. Ecrire la fonction *void AugmenterSalaire (struct Salarie *S)* qui augmente le salaire d'un salarié selon les taux suivants: 4% pour un ouvrier, 5% pour une secrétaire et 7% pour un directeur. (2pts)
4. Ecrire la fonction *void Afficher (struct Salarie E)* qui permet d'afficher les informations d'un salarié. (1pt)
5. Ecrire la fonction principale qui déclare un tableau de 20 salariés, effectue la saisie de leurs informations, augmente leurs salaires puis affiche leurs informations. (1.5pts)

Deuxième partie (09.50pts)

6. Ecrire la fonction *void OrdreAssSalaire (struct Salarie T[], int taille)* qui permet d'ordonner les informations des salariés dans le tableau T selon leurs salaires du moins élevé au plus élevé. (2pts)
7. Ecrire la fonction *struct Salarie RechMeillSalOuv(struct Salarie T[], int taille)* qui retourne une variable de type *struct Salarie* contenant les données de l'ouvrier possédant le meilleur salaire (supposé unique). (2pts)
8. Ecrire la fonction *void OrdreAlphNom (struct Salarie T[], int taille)* qui permet d'ordonner les informations des salariés dans le tableau T selon l'ordre alphabétique de leurs noms. (2pts)
9. Ajouter dans la fonction principale de l'exercice n°1 l'appel aux fonctions *OrdreAssSalaire( ), RechMeillSalOuv( )* et *OrdreAlphNom( )*. Utilisez la fonction *Afficher( )*. (1.5pts)
10. On suppose que plusieurs salariés peuvent avoir le même nom de famille. On veut alors consolider l'ordre alphabétique sur les noms par l'ordre alphabétique sur les prénoms (c-à-d établir l'ordre alphabétique sur les noms et les prénoms en même temps). Que proposer vous comme démarche pour la solution? (2pts)

CamScanner

ENIG	Devoir de contrôle	Groupe : GCR1 Durée : 1h30 Date : 29/10/2007 Nbr pages : 2
	Programmation en C (Aucun document n'est autorisé)	

La qualité de rédaction, la clarté et la précision des réponses et l'utilisation des commentaires seront des éléments d'évaluation très importants. Le barème est donné à titre indicatif.

Exercice 1 (5 pt)

```
#include<stdio.h>
void main()
{
        int i, j ;
        int k[3][4]={{1,2,3},{4,5,6,7},{9,1}} ;
        for( i=0 ; i<4 ; i++)
          {for(j=0 ; j<3 ; j++)
                printf("k[%d][%d] \t", i, j ) ;
          }
}
```

Question :

Corrigez le programme pour qu'il affiche la matrice sous cette forme :

1	2	3	0
4	5	6	7
9	1	0	0

Exercice 2 (7 pt)

Ecrire un programme C qui compte le nombre d'occurrence des chiffres compris entre 0 et 9 à partir des caractères taper au clavier. Le nombre d'occurrence des chiffres sera stoker dans un tableau. Utiliser la fonction *getchar()* pour la lecture.

Exercice 3 (8 pt)

Réaliser un programme C de tri par valeur décroissante d'un tableau d'entiers en utilisant l'algorithme dit "par extraction simple" qui se définit de la manière suivante :

- On recherche le plus grand des n éléments du tableau.
- On échange cet élément avec le premier élément du tableau
- Le plus grand élément se trouve alors en première position. On peut alors appliquer les deux opérations précédentes aux (n-1) éléments restants, puis aux (n-2), etc. et cela jusqu'à ce qu'il ne reste plus qu'un seul élément (le dernier qui est alors le plus petit).

CamScanner

ENIG	**République Tunisienne** **Ministère de l'Enseignement Supérieur et de la Recherche Scientifique** **Université de Gabès** **École Nationale d'Ingénieurs de Gabès**	**Réf : DE-EX-01**
		Indice : 3
	EPREUVE D'EVALUATION	**Date : 02/12/2019**
		Page : 1/2

Année Universitaire : 2022/2023	**Date de l'Examen** : 27/10/2022
Nature : ☑ DC ☐ Examen ☐ DR	**Durée :** ☐ 1h ☑ 1h30min ☐ 2h
Diplôme : ☐ Mastère ☑ Ingénieur	**Nombre de pages : 02**
Section : ☐ GCP ☐ GCV ☐ GEA ☑ GCR ☐ GM	**Enseignant : I. Azaïez**
Niveau d'étude : ☑ 1ère ☐ 2ème ☐ 3ème année	**Documents Autorisés :** ☐ Oui ☑ Non
Matière : Programmation I	**Remarque :**

NB. La qualité de rédaction, la clarté et la précision des réponses et l'utilisation des commentaires seront des d'évaluation très importants. **Le barème est donné à titre indicatif.**

Exercice 1 : Définir le type de variables et donnez le résultat d'affichage du programme suivant ? **(5 pts)**

```
#include <stdio.h>
#include <string.h>
int main() {

    voiture v;
    strcpy(v.marque,"Renault");
    voiture w=v;
    strcpy(w.marque,"Kia");
    voiture x=v;

    printf("%s %s %s \n",
v.marque,
       w.marque, x.marque);
```

```
if(w.marque==x.marque){
        strcpy(w.marque,"Ferrari");
        }
    else
        strcpy(x.marque,"Tata");

printf("%s %s %s \n", v.marque,
          w.marque, x.marque);
strcpy(v.marque,"Kia");
if(w.marque!=v.marque){
        strcpy(w.marque,"Cadillac");
        }
    else
        strcpy(x.marque,"Mercedes");
 printf("%s %s %s \n", v.marque,
              w.marque, x.marque);
    return 0;
}
```


CamScanner

Exercice 2 : (15 pts)

Écrire un programme en C qui calcule le montant maximal d'argent que nous pouvons dépenser pour acheter des vêtements de plusieurs modèles sans dépasser le budget alloué.

Étant donné différents modèles (M) pour chaque vêtement (par exemple 3 modèles de chemises, 2 modèles de ceintures, 4 modèles de chaussures, …) et un budget limité (B).

Votre tâche consiste à acheter un modèle de chaque vêtement en dépensant le montant maximal possible sans dépasser le budget (B).

L'entrée se compose de deux nombres entiers $1 \leq B \leq 200$ et $1 \leq M \leq 20$, où B est le budget et M, est le nombre de modèle de chaque vêtement V.

Pour chaque modèle d'un vêtement V, nous définissons un entier K indiquant son prix.

La sortie est un nombre entier qui indique le montant maximal d'argent que nous pouvons dépenser pour acheter un vêtement de chaque modèle sans dépasser le budget. S'il n'y a pas de solution en raison du petit budget qui nous est accordé, écrivez simplement : « ***pas de solution*** ».

Exemple : Supposons que nous ayons le cas de test suivant avec **B = 20, V = 3 :**

- Prix des 3 modèles de vêtement v1 →6 - 4 - 8
- Prix des 2 modèles de vêtement v2 →5 - 10
- Prix des 4 modèles de vêtement v3 →1 – 5 – 3 - 5

Pour ce cas test, la réponse est 19, est le résultat de l'achat des éléments soulignés (8+10+1). Ce n'est pas unique, car les solutions (6+10+3) et (4+10+5) sont également possible.

Cependant, supposons que nous ayons ce cas de test avec **B = 9 (budget limité), V = 3 :**

La réponse n'est alors « *pas de solution* » car même si nous achetons tous les modèles les moins chers pour chaque vêtement, le prix total (4+5+1) = 10 dépasse toujours notre budget donné B = 9.

On y prévoira le déroulement suivant :

1. Saisir le montant du budget *B*
2. Saisir le prix de chaque modèle *Mi* pour chaque vêtement *Vi* à completer
3. Chercher le modèle *Mi* (dont le prix maximal) ssi `B-prix_Mi >= 0` sinon passer au prix des autres modèles du même vêtement *Vi*
4. Répéter la « tâche 3 » pour tous les vêtements *V*
5. Afficher le résultat sous forme d'un message mentionnant montant maximal d'argent que nous pouvons dépenser et le nombre de modèle acheter, sinon « *pas de solution* » pour un budget limité.

CamScanner

Service Kamel
Tirage & Photo Copie

ENIG	Devoir de Synthèse	Groupe : GCR1 Durée : 2h Date : 26/01/2007 Nbr pages : 2
	Programmation en C (Aucun document n'est autorisé)	

La qualité de rédaction, la clarté et la précision des réponses et l'utilisation des commentaires seront des éléments d'évaluation très importants. Le barème est donné à titre indicatif.

Exercice 1 (3 pt)

مكتبة الطالب
نسخ و تنسيق الوثائق
و اصلاح الكتب - زريق
P. 97 ...

Soit le programme suivant :

```
#include <stdio.h>
void main()
{
        struct s1
        {
                char * s;
                int i;
                struct s1 * s1p;
        };

        struct s1 a[3]={{"abcd",1,a+1},{"efg",2,a+2}, {"ijkl",3,a}};

        struct s1 * p = a;
        printf("%s %s %s \n" a[0].s, p->s, a[2].s1p->s);
}
```

Service Kamel
Tirage & Photo Copie
Tél:53.756.027 - Fax:75 392 936

1) Donnez la représentation des données dans la mémoire centrale.

2) Quels résultats affichés sur l'écran.

CamScanner

Exercice 3 (10 pt)

Réaliser un programme de tri par valeur décroissante d'un tableau d'entiers en utilisant l'algorithme dit "par extraction simple" qui se définit de la manière suivante :

- On recherche le plus grand des n éléments du tableau.
- On échange cet élément avec le prémier élément du tableau
- Le plus grand élément se trouve alors en première position. On peut alors appliquer les deux opérations précédentes aux (n-1) éléments restants, puis aux (n-2), etc. et cela jusqu'à ce qu'il ne reste plus qu'un seul élément (le dernier-qui est alors le plus petit).

On y prévoira les fonctions suivantes :

- Une fonction *Affichage (...)* qui affiche les éléments du tableau.
- Une fonction *Permute (...)* qui permute deux éléments du tableau.
- Une fonction *Tri (...)* qui fait le tri du tableau.
- Le programme principal main () pour le test.

CamScanner

ENIG	Devoir de synthèse Programmation C (Aucun document n'est autorisé)	Groupe : GCR1 Durée : 2h Date : 09/01/2016 Nr pages : 2

NB. La qualité de rédaction, la clarté et la précision des réponses seront des éléments d'évaluation très importants. Vous devez traiter chaque partie sur des feuilles séparées.

Partie I : (12 pts)

On se propose de créer un programme pour gérer un groupe d'étudiants. Chaque étudiant sera caractérisé par :

- Son nom (chaine de caractère),
- Son prénom (chaine de caractère),
- Son âge (un entier),
- Son groupe (chaine de caractère),
- Ses notes (un tableau de 5 éléments de type double).

1. Donnez la définition de la structure correspondant aux critères énoncés ci-dessus. Utilisez ensuite un typedef pour définir votre nouveau type « etudiant » utilisable dans la suite de votre programme. 2 pts
2. On se propose maintenant d'écrire une fonction permettant de remplir une structure de ce nouveau type ainsi qu'une autre fonction permettant d'afficher les informations sur un étudiant. Écrire ces deux fonctions, en suivant la fonction main() et les prototypes suivants : 3 pts

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

void affichageEtudiant (etudiant);
void saisirInformationEtudiant(etudiant * e);

void main (void){
        etudiant e;
        puts("Entrez les informations sur un étudiant : ");
        saisirInformationEtudiant(…);
        affichageEtudiant(…);
}
```

3. Écrivez une fonction permettant de demander à l'utilisateur les notes d'un étudiant. Modifiez ensuite la fonction main() pour tester votre nouvelle fonction. 2 pts
4. On veut maintenant pouvoir gérer un groupe d'étudiants sans pour autant fixer la taille a priori 3 pts
   - Écrivez une fonction qui alloue dynamiquement un tableau de structures nécessaire au stockage d'un groupe d'étudiants en mémoire. Elle renvoie le nombre d'étudiants créés.
   - Saisir les informations et les notes pour chaque étudiant en utilisant les fonctions *saisirInformationsEtudiant()* et *saisirNotesEtudiant()*, définis précédemment.
   - Modifiez votre main() pour qu'il appelle cette fonction afin de créer le tableau d'étudiants correspondant à un groupe et il affiche ensuite les étudiants un à un.

   N'oubliez pas de libérer la mémoire à la fin du programme.
5. Écrivez la fonction affichageBulletin() affichant pour chaque étudiants ses informations, puis une te par ligne sous la forme « Note i : xx/20 » et enfin une ligne Moyenne : xx/20. Pour cette moyenne, ajoutez un champ "moyenne" à la structure de départ, puis écrivez une fonction permettant le remplissage de ce champ pour une structure donnée. 2 pts

CamScanner

Partie 2 :

Exercice 1 : (4 pts)

Ecrire un programme de création séquentielle d'un fichier comportant pour un certain nombre de personnes, les informations suivantes, fournies au clavier :

- Nom (au maximum 20 caractères) ;
- Age ;
- Nombre d'enfants ;
- Age de chacun des différents enfants ; on ne demandera (et donc on n'enregistrera) que l'âge des 15 premiers enfants. On peut utiliser un tableau pour enregistrer les âges des enfants.

L'utilisateur fournira un nom « vide » pour signaler qu'il n'a plus de personnes à enregistrer.

Exemple

```
Donner le nom du fichier à créer : personne
---Pour terminer la saisie, donner un nom vide---
Nom                 : Mhanna
Age                 : 32
Nombre enfants      : 1
Age enfant no 1     : 7

Nom                 : Ben ali
Age                 : 29
Nombre enfants      : 0

Nom                 : Kammoun
Age                 : 45
Nombre enfants      : 3
Age enfant no 1     : 21
Age enfant no 2     : 18
Age enfant no 3     : 17
```

Exercice 2 : (4 pts)

Une pile est une structure de données telle que:

- L'ajout d'un élément se fait au sommet de la file,
- La suppression d'un élément se fait également au sommet de la file.

Ecrire les fonctions suivantes :

- InitialiserPile ( ) : qui initialise un liste chaînées vide
- FileVide ( ) : qui retourne vrai si la liste des vide
- Empiler ( ) : qui permet d'ajouter un élément en tête de la liste
- Depiler ( ) : qui enlève un élément en tête de la liste

Créer un menu qui permet de faire appel à chacune de ces fonctions.

CamScanner