```
    int i = 0;
    while(chaine[i] != '\0') {
        printf("%c\n", chaine[i]);
        i++;
    }
    printf("La chaine fait %d caracteres\n", i);
    printf("La chaine fait %d caracteres\n", strlen(chaine));
    return 0;
}
```

```
#include <stdio.h>

int main() {
    int var, somme, count;
    float moyen;
    somme = 0;
    count = 0;
    do {
        printf("Veuillez saisir un entier strictement positif...\n");
        scanf("%d", &var);
        if(var >0) {
            printf("Valeur saisie : %d\n", var);
            count++;
            somme+=var;
        } else if(var != -1) {
            printf("Erreur, %d n'est pas un entier strictement positif\n", var)
                ;
        }

    } while(var != -1);
    moyen = (float)somme/count;
    printf("La moyenne est de %f\n", moyen);
    return 0;
}
```

```
#include <stdio.h>

int main() {
    char x;
    int tab,i;
    do {
        printf("Quelle table de multiplication (tapez 0 pour sortir) ?\n");
        x = getchar();
        while(getchar() != '\n'); // On vide le buffer
        if(x > '0' && x <= '9') {
            tab = x - '0'; // recup l'entier correspondant au car
            for(i=0; i<=9; i++) {
                printf("%d * %d = %d\n", i, tab, i*tab);
            }
        } else if(x != '0') {
            printf("Ce n'est pas dans les possibilites du programme\n");
        }
    } while(x != '0');
    return 0;
}
```

Exercice 6 - Pointeurs (tous)

1. Déclarer une variable i de type entier et l'initialiser à 5. Afficher l'adresse de i et la valeur de i.

2. Ajouter une variable j de type pointeur vers un entier et l'initialiser avec l'adresse de i. Afficher l'adresse de j, la valeur de j et la valeur pointée par j.
3. Incrémenter la donnée pointée par j. Afficher la valeur de i et la valeur pointée par j.
4. Multiplier i par 5. Afficher la valeur de i et la valeur pointée par j.
5. Incrémenter j. Afficher la valeur de i, la valeur de j et la valeur pointée par j.

Solution de l'exercice 6

```
#include <stdio.h>

int main() {

    int i;
    int *j;
    i = 5;
    j = &i;
    printf("\nadresse de i : %x, valeur de i : %d\n", &i, i);
    printf("\nadresse de j : %x, valeur de j : %x, valeur pointee par j : %d\n"
        , &j, j, *j);

    return 0;
}
```

```
#include <stdio.h>

int main() {

    int i;
    int *j;
    i = 5;
    j = &i;
    printf("\nadresse de i : %x, valeur de i : %d\n", &i, i);
    printf("\nadresse de j : %x, valeur de j : %x, valeur pointee par j : %d\n"
        , &j, j, *j);
    (*j)++;
    printf("i vaut maintenant %d\n", i);
    i*=5;
    printf("La donnee pointee par j vaut %d\n", *j);
    j++;
    printf("\nadresse de j : %x, valeur de j : %x, valeur pointee par j : %d\n"
        , &j, j, *j);

    return 0;
}
```

Exercice 7 - Tableaux et chaînes de caractères (1 à 7)

1. Déclarer un tableau de 10 entiers. À l'aide d'une boucle, affecter la valeur 7 à tous les indices du tableau.
2. Déclarer un tableau de 5 entiers en l'initialisant avec les valeurs décroissantes allant de 4 à 0.
3. Déclarer une chaîne de caractères en l'initialisant à "bonjour". Afficher tous les caractères de cette chaîne jusqu'au caractère `\0`, puis afficher la chaîne complète.
4. Demander à l'utilisateur de saisir une chaîne de caractères. Remplacer le premier caractère 'o' par le caractère 'a' et le premier caractère 'n' par le caractère 'd' puis afficher la chaîne complète.
5. Déclarer un tableau d'entiers de taille *NMAX*. Demander la saisie de chaque valeur du tableau, calculer la somme des valeurs, stocker le résultat dans une variable *somme* et l'afficher.
6. Reprendre le programme précédent. Ajouter le calcul de la moyenne puis afficher les éléments du tableau case par case en indiquant l'indice, la valeur et un message si celle-ci est plus grande ou égale à la moyenne.

7. Déclarer un tableau de dimension 2 (3 lignes et 4 colonnes) et l'initialiser avec les valeurs allant de 12 à 23 : la première ligne contiendra les valeurs 12, 13, 14, 15 ; la deuxième 16, 17, 18, 19 ; la troisième 20, 21, 22, 23. Afficher les indices et valeurs du tableau.
8. Reprendre l'exercice précédent avec la notation en pointeurs pour la ligne.
9. Reprendre l'exercice avec une notation en pointeurs uniquement.
10. Déclarer trois matrices d'entiers de taille 3x3, saisir les coefficients de deux matrices et additionner les deux matrices en stockant le résultat dans la troisième.
11. Reprendre l'exercice en multipliant les deux matrices.
12. Déclarer un tableau de 10 chaînes de caractères de 15 caractères chacune, initialiser toutes les chaînes avec le caractère \0.
13. Ajouter la saisie des 10 chaînes au clavier, puis les afficher à l'écran.

Solution de l'exercice 7

```
#define NMAX 10

int main() {
    int i;
    int tab[NMAX];
    for(i=0; i<NMAX; i++) {
        tab[i] = 7;
    }
    return 0;
}
```

```
#include <stdio.h>
#define NMAX 5

int main() {

    char bjr1[] = {'b','o','n','j','o','u','r', '\0'};
    char bjr[] = "bonjour";
    printf("%s\n", bjr);
    int i=0;
    while(bjr[i]!='\0') {
        printf("%c\n", bjr[i]);
        i++;
    }
    bjr[1] = 'a';
    bjr[2] = 'd';
    printf("%s\n", bjr);
    return 0;
}
```

```
#include <stdio.h>
#define NMAX 5

int main() {

    int i,somme;
    int tab[NMAX];
    float moyen;
    somme = 0;
    for(i=0; i<NMAX; i++) {
        printf("Veuillez saisir la valeur %d\n", i);
        scanf("%d", &tab[i]);
        somme += tab[i];
    }
    moyen = (float)somme/NMAX;
```

```
    for(i=0; i<NMAX; i++) {
        printf("La valeur a l'indice %d  du tableau est %d\n", i, tab[i]);
        if(tab[i]>=moyen) {
            printf("La valeur %d est superieure a la moyenne (%f)\n", tab[i],
               moyen);
        }
    }
    printf("La somme vaut %d\n", somme);
    return 0;
}
```

```
#include <stdio.h>
#define NROW 3
#define NCOL 4
int main() {

    int tab[NROW][NCOL] = {12,13,14,15,16,17,18,19,20,21,22,23}
    // ou tab[NROW][NCOL] = {{12,13,14,15},{16,17,18,19},{20,21,22,23}}
    /*int val = 12;
    int i, j;
    for(i=0; i<NROW; i++) {
        for(j=0; j<NCOL; j++) {
            tab[i][j] = val;
            val++;
        }
    }*/
    for(i=0; i<NROW; i++) {
        for(j=0; j<NCOL; j++) {
            printf("tab[%d][%d] = %d\n", i, j, tab[i][j]);
        }
    }
    return 0;
}
```

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

#define NROW 3
#define NCOL 4
int main() {

    int * tab[NROW];
    int val = 12;
    int i, j;
    for(i=0; i<NROW; i++) {
        tab[i] = malloc(sizeof(int)*NCOL);
        for(j=0; j<NCOL; j++) {
            tab[i][j] = val;
            val++;
        }
    }
    for(i=0; i<NROW; i++) {
        for(j=0; j<NCOL; j++) {
            printf("tab[%d][%d] = %d\n", i, j, tab[i][j]);
        }
    }
    for(i=0; i<NROW; i++) {
        free(tab[i]);
    }
    return 0;
}
```

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

#define NROW 3
#define NCOL 4
int main() {

    int i,j;
    int ** tab = malloc(sizeof(int)*NROW);
    int val = 12;
    int ** pnt1 = tab;
    int * pnt2;
    for(i=0; i<NROW; i++) {
        pnt2 = malloc(sizeof(int)*NCOL);
        *pnt1 = pnt2;
        for(j=0; j<NCOL; j++) {
            *pnt2 = val;
            pnt2++;
            val++;
        }
        pnt1++;
    }
    pnt1 = tab;
    for(i=0; i<NROW; i++) {
        pnt2 = *pnt1;
        for(j=0; j<NCOL; j++) {
            printf("tab[%d][%d] = %d\n", i, j, *pnt2);
            pnt2++;
        }
        pnt1++;
    }
    pnt1 = tab;
    for(i=0; i<NROW; i++) {
        free(*pnt1);
        pnt1++;
    }
    free(tab);
    return 0;
}
```

Exercice 8 - Fonctions et passage de paramètres (1 à 9)

1. Reprendre le problème de l'échange de deux variables entières x et y en créant une fonction *void swap(int * a, int * b)*. Justifier l'utilisation de pointeurs.
2. Écrire une fonction *my_ isascii(char c)* qui retourne 1 si c fait partie de la table ASCII et 0 sinon.
3. Écrire une fonction *my_ strlen* qui retourne la longueur d'une chaîne de caractères.
4. Écrire une fonction qui demande de saisir un nombre entier n plus petit ou égal à 10 à l'utilisateur. n est le nombre de cases qui seront utilisées dans un tableau d'entiers de taille $MAX = 10$ transmis en paramètre de la fonction. Demander ensuite de saisir les n valeurs du tableau. La fonction retournera n.
5. Écrire une fonction *min* qui renvoie la plus petite valeur comprise dans un tableau d'entiers. Quels sont les paramètres nécessaires à cette fonction ?
6. Écrire une fonction *en_ domain(int a, int b, int c)* qui retourne 1 si $a \in [b; c]$ et 0 sinon.
7. Écrire une fonction qui transforme tous les caractères minuscules d'une chaîne en caractères majuscules.
8. Soit le programme suivant :

```c
include <stdio.h>
int main(int argc, char * argv[]) {
```

```
  while(--argc>0) {
    printf("%s",*++argv);
    printf("\n") ;
  }
}
```

Étudier son fonctionnement, ainsi que le passage de paramètres de la fonction principale.

9. Reprendre le calcul de la factorielle sous la forme d'une fonction récursive.
10. Écrire une fonction *my_ isalpha(char c)* qui retourne 1 si *c* est une lettre.
11. Écrire une fonction *my_ isdigit(char c)* qui retourne 1 si *c* est un chiffre
12. Écrire une fonction *somme* (2 paramètres) qui calcule et retourne la somme de deux entiers.
13. Reprendre la fonction *somme* et écrire la fonction *somme2* qui retourne son résultat dans un argument d'appel supplémentaire (donc la fonction ne retourne plus aucune valeur).
14. Écrire une fonction *fact1* (1 paramètre) qui calcule la factorielle de façon itérative.
15. Écrire une fonction *my_ toupper* (1 paramètre) qui transforme un caractère c en la valeur de la majuscule associée (s'il y en a une, sinon ne modifie rien).
16. Reprendre l'exercice précédent et écrire *my_ tolower* (1 paramètre) qui effectue le travail inverse.
17. Écrire la fonction *my_ atoi* (1 paramètre) qui prend une chaîne de caractères en paramètre et retourne l'entier lu depuis celle-ci.
18. Écrire la fonction *my_ ltoa* (2 paramètres) qui transforme un entier *long* en une chaîne de caractères.
19. Écrire une fonction *my_ strcmp* (2 paramètres) qui renvoie vrai si deux chaînes de caractères sont égales, faux sinon.
20. Écrire une fonction *my_ strcat* (2 paramètres) qui concatène une chaîne source à une chaine destination et retourne un pointeur sur le résultat.
21. Écrire une fonction *my_ strcpy* (2 paramètres) qui recopie une chaîne source dans une chaine destination et retourne un pointeur sur le résultat.
22. Écrire une fonction *my_ strnset* (3 paramètres) qui recopie *nbr* fois le caractère *c* dans une chaîne destination et retourne un pointeur sur le résultat.
23. Écrire une fonction *my_ strdup* (1 paramètre) qui recopie une chaîne de caractères vers un emplacement réservé par *malloc* et retourne un pointeur sur le résultat.

Solution de l'exercice 8

```
#include <stdio.h>

// Declaration de la fonction swap
void swap(int *a, int *b);

int main() {

    int x, y;
    printf("Veuillez renseigner 2 entiers...\n");
    scanf("%d %d", &x, &y);
    printf("Avant swap, x vaut %d et y vaut %d\n", x, y);
    swap(&x, &y);
    printf("Apres swap, x vaut %d et y vaut %d\n", x, y);
    return 0;
}

// Definition de la fonction swap
void swap(int *a, int *b) {
    int buffer;
    buffer = *a;
    *a = *b;
    *b = buffer;
}
```

```c
#include <stdio.h>

int my_isascii(char c) {
    return (c >= 0 && c<= 127);
}

int main() {
    char x;
    scanf("%c", &x);
    int result = my_isascii(x);
    printf("Result vaut %d\n", result);
    return 0;
}
```

```c
#include <stdio.h>

int my_strlen(char * pt) {
    int i=0;
    while(*(pt+i) != '\0') {
        i++;
    }
    return i;
}

int main() {

    char bjr[] = "Bonjour";
    printf("%s est de longueur %d\n", bjr, my_strlen(bjr));
    return 0;
}
```

```c
#include <stdio.h>
#define NMAX 10

int compte(int * tab);

int main() {

   int tab1[NMAX];
   int resultat = compte(tab1);
   printf("%d valeurs ont ete saisies\n", resultat);
   return 0;
}

int compte(int * tab) {
    int nb,i;
    printf("Veuillez donner le nombre d'elements a lire\n");
    scanf("%d", &nb);
    for(i=0; i<nb; i++) {
        printf("Valeur de l'element %d : \n", i);
        scanf("%d", &tab[i]);
    }
    return nb;
}
```

```c
#include <stdio.h>
#define NMAX 6

int min(int * tab, int n);
int max(int * tab, int n);
```

```
int main() {
    int ex1[NMAX] = { 45, 75, 23, 149, 856, 130 };
    int minVal = min(ex1, NMAX);
    int maxVal = max(ex1, NMAX);
    printf("La plus petite valeur est %d\n", minVal);
    printf("La plus grande valeur est %d\n", maxVal);
    return 0;
}

int min(int * tab, int n) {
    int i, res;
    res = tab[0]; // res = *tab
    for(i=1;i<n;i++) {
        if(tab[i]<res) { // if(*(tab+i)<res) {
            res = tab[i]; // res = *(tab+i);
        }
    }
    return res;

}
int max(int * tab, int n) {
    int i, res;
    res = tab[0]; // res = *tab
    for(i=1;i<n;i++) {
        if(tab[i]>res) { // if(*(tab+i)<res) {
            res = tab[i]; // res = *(tab+i);
        }
    }
    return res;
}
```

```
#include <stdio.h>
#define NMAX 7

int in_domain(int n, int lb, int ub);

int main() {

    int tab[NMAX] = {23, 76, 4, -9, 146, 867, 4};
    int inf, sup,i;
    printf("Veuillez saisir une borne inf...\n");
    scanf("%d", &inf);
    printf("Veuillez saisir une borne sup...\n");
    scanf("%d", &sup);
    printf("Valeurs dans l'intervalle [%d;%d]\n", inf, sup);
    for(i=0;i<NMAX;i++) {
        if(in_domain(tab[i], inf, sup) == 1) {
            printf("%d ", tab[i]);
        }
    }
    printf("\n");
    return 0;
}

int in_domain(int n, int lb, int ub) {
    return (n>=lb && n <= ub);
}
```

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

```c
#define NMAX 50

void maj(char * chaine);

int main() {

    char str[NMAX];
    printf("Veuillez entrer une chaine\n");
    scanf("%s", str);
    maj(str);
    printf("La chaine en majuscules est %s\n", str);

    return 0;
}

void maj(char * chaine) {
    int i;
    for(i=0; i<strlen(chaine); i++) { // while (chaine[i] != '\0') {
        if(chaine[i] >= 'a' && chaine[i] <= 'z') {
            chaine[i]-='a'-'A';
        }

    }

}
```

```c
#include <stdio.h>

int fact(int n) {
    if(n==1 || n==0) {
        return 1;
    } else {
        return n*fact(n-1);
    }
}

int main() {
    int x;
    printf("Quelle factorielle voulez-vous calculer ?\n");
    scanf("%d", &x);
    printf("%d! = %d\n", x, fact(x));
    return 0;
}
```

Exercice 9 - Structures (1 à 7)

1. Détailler la déclaration d'une structure ainsi que le mode d'accès direct et en utilisant des pointeurs.
2. Déclarer la structure *Un_ Tableau_ Entier* contenant un tableau d'entiers de taille $NMAX$ et une variable entière nommée *ncase* donnant le nombre de cases utilisées dans le tableau.
3. Déclarer la structure *menu* contenant un tableau (d'au plus 20 items) de chaîne de caractères (d'au plus 60 caractères). La structure contiendra aussi un entier n représentant le nombre d'items du menu.
4. Ajouter une fonction qui retourne la longueur de la plus longue chaîne parmi celles d'un menu.
5. Ajouter une fonction qui affiche les items d'un menu en les centrant par rapport à la plus longue chaîne. Chaque item est précédé de son numéro d'indice (entre 1 et n).
6. Ajouter une fonction qui affiche le menu et demande à l'utilisateur "Veuillez choisir votre menu (q ou Q pour quitter)". Si la saisie est valide, retourner l'indice du menu saisi (entre 1 et n). Si l'utilisateur tape 'q' ou 'Q', retourner 0. Si l'utilisateur tape autre chose, effacer l'écran et afficher de nouveau le menu.