Exercices et Solutions Langage C

NOTIONS DE BASE

Exercice

Ecrire un programme C qui permet d'afficher le message suivant: *Bonjour*.

Solution

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
        printf("Bonjour\n");
        system("pause");
        return 0;
}
```

Exercice

Ecrire un programme C qui permet de lire en entrée un entier constitué de trois chiffres et d'afficher celui-ci inversé. Exemple: si l'entrée est *123* on affiche *321*.

Solution

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    int a,b,c;
    printf("donnez un entier de 3 chiffres\n");
    scanf("%d",&a);
    c=a;
    b=(a%10)*100;
    a=a/10;
    b=b+(a%10)*10;
    a/=10;
    b+=a;
    printf("le nombre %d inverse est %d\n",c,b);
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui permet de lire deux nombres réels, et d'afficher ensuite leur produit, avec une précision de trois chiffres après la virgule.

Solution

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    float x,y,p;
    printf("Entrez deux reels:\n");
```

```
7.      scanf("%f%f",&x,&y);
8.      p=x*y;
9.      printf("le produit de %f et %f est: %.3f\n",x,y,p);
10.     system("pause");
11.     return 0;
12. }
```

Exercice

Ecrire un programme C qui permet de permuter le contenu de deux variables entières en passant par une troisième variable auxiliaire. Ceci et en affichant les deux variables avant et après permutation.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.      int a,b,tmp;
6.      printf("Entrez deux entiers a et b:\n");
7.      scanf("%d%d",&a,&b);
8.      printf("Avant permutation: a = %d et b = %d.\n",a,b)
   ;
9.      tmp=a;
10.     a=b;
11.     b=tmp;
12.     printf("Apres permutation: a = %d et b = %d.\n",a,b)
   ;
13.     system("pause");
14.     return 0;
15. }
```

Exercice

Ecrire un programme C qui lit en entrée trois entiers et affiche leur moyenne avec une précision de deux chiffres après la virgule.

Solution

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    int a,b,c;
    float moy;
    printf("Donnez 3 entiers:\n");
    scanf("%d%d%d",&a,&b,&c);
    moy=(float)(a+b+c)/3;
    printf("La moyenne de ces trois entiers est: %.2f\n",moy);
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui lit en entrée un caractère alphabétique entre *a* et *y*, qui peut être soit une majuscule ou une minuscule. Et affiche la lettre qui vient juste après lui dans l'ordre alphabétique.

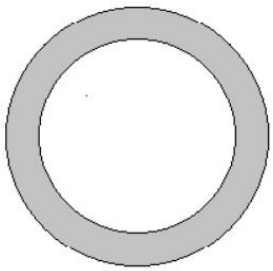
Solution

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    char lettre;
```

```
6.      printf("Donnez une lettre entre 'a' et 'y' ou entre
   'A' et 'Y':\n");
7.      scanf("%c",&lettre);
8.      lettre++;
9.      printf("La lettre suivante est: %c.\n",lettre);
10.     system("pause");
11.     return 0;
12. }
```

Exercice

Ecrire un programme C qui lit deux réels R_1 et R_2 qui représentent les rayons de deux cercles concentriques, et renvoie ensuite l'aire de la surface comprise entre les deux cercles (surface grise). Remarque: R_1 peut être supérieur à R_2, comme il peut lui être inférieur.



Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<math.h>
4. int main()
5. {
6.      double R1,R2,aire1,aire2,aire;
7.      printf("Donnez les rayons des deux cercles:\n");
8.      scanf("%lf%lf",&R1,&R2);
9.      aire1 = 3.14*R1*R1;
```

```
10.     aire2 = 3.14*R2*R2;
11.     aire = fabs(aire1-aire2);
12.     printf("L'aire de la surface entre les deux cercles
    est: %.4lf\n",aire);
13.     system("pause");
14.     return 0;
15. }
```

Exercice

Ecrire un programme C qui lit deux entiers est affiche le plus grand d'entre eux.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<math.h>
4. int main()
5. {
6.     int a,b,max;
7.     printf("Entrez deux entiers:\n");
8.     scanf("%d%d",&a,&b);
9.     max = (a+b+abs(a-b))/2;
10.     printf("Le max de %d et %d est: %d.\n",a,b,max);
11.     system("pause");
12.     return 0;
13. }
```

Exercice

Ecrire un programme C qui trouve pour un réel les deux carrés parfaits les plus proches qui l'encadrent.

On rappelle qu'un carré parfait est un entier dont la racine carrée est aussi un entier. Exemple: *9 = 3x3* et *4 = 2x2* sont des carrés parfaits ; or, *5* ne l'est pas.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<math.h>
4. int main()
5. {
6.     double x;
7.     int a,b;
8.     printf("Entrez un reel:\n");
9.     scanf("%lf",&x);
10.    a = (int)floor(sqrt(x));
11.    a = a*a;
12.    b = (int)ceil(sqrt(x));
13.    b = b*b;
14.    printf("%d <= %lf <= %d\n",a,x,b);
15.    system("pause");
16.    return 0;
17.}
```

Exercice

Ecrire un programme C qui lit une fraction au format *a/b* où *a* et *b* sont deux entiers, et donne son équivalent décimal avec une précision de quatre chiffres après la virgule.

Ex: si l'utilisateur entre *3/2*, le programme doit afficher: *3/2 = 1.5000*

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     int a,b;
6.     float d;
7.     printf("Donnez une fraction:\n");
8.     scanf("%d/%d",&a,&b);
9.     d = (float)a/b;
10.    printf("%d/%d = %.4f\n",a,b,d);
11.    system("pause");
12.    return 0;
13.}
```

CONDITIONS

Exercice

Ecrire un programme C qui permet de dire si un entier est pair ou impair.

Solution

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

```
3. int main()
4. {
5.     int a;
6.     printf("Donnez un entier:\n");
7.     scanf("%d",&a);
8.     if(a%2==1)
9.     {
10.       printf("%d est impair\n",a);
11.    }
12.    else
13.    {
14.       printf("%d est pair\n",a);
15.    }
16.    system("pause");
17.    return 0;
18.}
```

Exercice

Ecrire un programme C qui permet de comparer deux entiers *a* et *b*, et d'afficher selon le cas l'un des messages suivants: *a=b*, *a>b* ou*a<b*.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     int a,b;
6.     printf("Donnez les deux entiers a comparer:\n");
```

```
7.      scanf("%d%d",&a,&b);
8.      if(a==b)
9.      {
10.        printf("%d=%d\n",a,b);
11.     }
12.     else
13.     {
14.        if(a<b)
15.        {
16.           printf("%d<%d\n",a,b);
17.        }
18.        else
19.        {
20.           printf("%d>%d\n",a,b);
21.        }
22.     }
23.     system("pause");
24.     return 0;
25.}
```

Exercice

Ecrire un programme C qui lit trois entiers pour les afficher ensuite dans un ordre croissant.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
```

```
4. {
5.      int a,b,c;
6.      printf("Donnez 3 entiers:\n");
7.      scanf("%d%d%d",&a,&b,&c);
8.      if(a>=b && a>=c)
9.      {
10.        if(b>=c) printf("%d %d %d\n",c,b,a);
11.        else printf("%d %d %d\n",b,c,a);
12.     }
13.
14.     if(b>=a && b>=c)
15.     {
16.        if(a>=c) printf("%d %d %d\n",c,a,b);
17.        else  printf("%d %d %d\n",a,c,b);
18.     }
19.
20.     if(c>=a && c>=b)
21.     {
22.        if(a>=b) printf("%d %d %d\n",b,a,c);
23.        else printf("%d %d %d\n",a,b,c);
24.     }
25.     system("pause");
26.     return 0;
27. }
```

Exercice

Ecrire un programme C qui lit un nombre réel et détermine s'il est entier ou non.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     float x;
6.     printf("Donnez un reel:\n");
7.     scanf("%f",&x);
8.     if(x==(int)x) printf("%f est un entier.\n",x);
9.     else printf("%f n'est pas un entier.\n",x);
10.    system("pause");
11.    return 0;
12.}
```

Exercice

Ecrire un programme C qui lit un caractère et détermine s'il fait partie des alphabets ou non. Et s'il l'est, dire en plus s'il est une minuscule ou une majuscule.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     char c;
6.     printf("Donnez un caractere:\n");
7.     scanf("%c",&c);
8.     if(c>='a' && c<='z')
```

```
9.          printf("%c est une alphabet en minuscule.\n",c);
10.     if(c>='A' && c<='Z')
11.         printf("%c est une alphabet en majuscule.\n",c);
12.     if( !(c>='a' && c<='z') && !(c>='A' && c<='Z'))
13.         printf("%c n'est pas une alphabet.\n",c);
14.     system("pause");
15.     return 0;
16. }
```

Exercice

Ecrire un programme C qui lit une date au format *15/09/2012* et l'affiche sous le format suivant: *15-Septembre-2012*.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     int jour,mois,annee;
6.     printf("Inscrivez une date (Ex: 15/9/2012):\n");
7.     scanf("%d/%d/%d",&jour,&mois,&annee);
8.     if(jour<10) printf("0%d-",jour);
9.     else printf("%d-",jour);
10.    switch(mois)
11.    {
12.        case 1: printf("Janvier");
13.             break;
14.        case 2: printf("Fevrier");
```

```
15.             break;
16.        case 3: printf("Mars");
17.             break;
18.        case 4: printf("Avril");
19.             break;
20.        case 5: printf("Mai");
21.             break;
22.        case 6: printf("Juin");
23.             break;
24.        case 7: printf("Juillet");
25.             break;
26.        case 8: printf("Aout");
27.             break;
28.        case 9: printf("Septembre");
29.             break;
30.        case 10: printf("Octobre");
31.             break;
32.        case 11: printf("Novembre");
33.             break;
34.        case 12: printf("Decembre");
35.             break;
36.     }
37.     printf("-%d\n",annee);
38.     system("pause");
39.     return 0;
40. }
```

Exercice

Ecrire un programme C qui lit un entier et dit s'il est un carré parfait ou non.

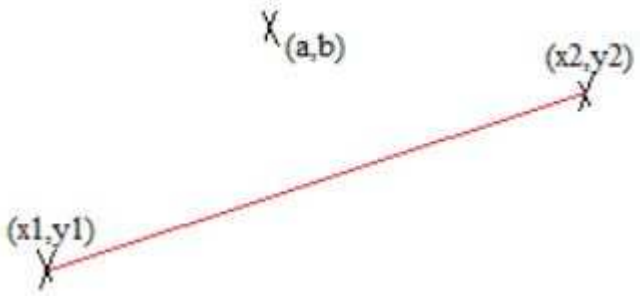
On rappelle qu'un entier est carré parfait, si sa racine carrée est entière. Ex: les nombre *1* (*1x1*), *4* (*2x2*) et *9* (*3x3*) sont tous des carrés parfaits.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<math.h>
4. int main()
5. {
6.     float n;
7.     printf("Entrez un entier:\n");
8.     scanf("%f",&n);
9.     if( sqrt(n) == (int)sqrt(n) ) printf("%.0f est un ca
   rre parfait.\n",n);
10.    else printf("%.0f n'est pas un carre parfait.\n",n);
11.    system("pause");
12.    return 0;
13.}
```

Exercice

Ecrire un programme C qui lit les coordonnées des deux extrémités d'un segment, et lit ensuite les coordonnées d'un point dans le plan et dit si ce dernier se trouve ou non sur le segment.



Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<math.h>
4. int main()
5. {
6.     double x1,y1,x2,y2,a,b;
7.     double dist1,dist2,dist;
8.     printf("Entrez le x1 et y1\n");
9.     scanf("%lf%lf",&x1,&y1);
10.    printf("Entrez le x2 et y2\n");
11.    scanf("%lf%lf",&x2,&y2);
12.    printf("Entrez le a et b\n");
13.    scanf("%lf%lf",&a,&b);
14.    dist1 = sqrt( (x1-a)*(x1-a)+(y1-b)*(y1-b) );
15.    dist2 = sqrt( (x2-a)*(x2-a)+(y2-b)*(y2-b) );
16.    dist = sqrt( (x1-x2)*(x1-x2)+(y1-y2)*(y1-y2) );
17.    if( dist == dist1 + dist2 ) printf("Le point est sur
   le segment.\n");
18.    else printf("Le point n'est pas sur le segment.\n");
19.    system("pause");
20.    return 0;
21.}
```

Exercice

Ecrire un programme C qui lit deux instants dans le format *HH:MM:SS*, et affiche un des messages suivants:

- Le premier instant vient avant le deuxième;
- Le deuxième instant vient avant le premier;
- Il s'agit du même instant.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     int h1,h2;
6.     int m1,m2;
7.     int s1,s2;
8.     int k=1;
9.     printf("Entrez le premier instant:\n");
10.    scanf("%d:%d:%d",&h1,&m1,&s1);
11.    printf("Entrez le deuxieme instant:\n");
12.    scanf("%d:%d:%d",&h2,&m2,&s2);
13.    if(h1>h2) k=2;
14.    if(h1==h2 && m1>m2) k=2;
15.    if(h1==h2 && m1==m2 && s1>s2) k=2;
16.    if(h1==h2 && m1==m2 && s1==s2) k=0;
17.    switch(k)
18.    {
19.       case 0: printf("Il s'agit du meme instant.\n");
20.            break;
```

```
21.         case 1: printf("Le premier instant vient avant le
    deuxieme.\n");
22.             break;
23.         case 2: printf("Le deuxieme instant vient avant l
   e premier.\n");
24.     }
25.     system("pause");
26.     return 0;
27. }
```

Exercice

Ecrire un programme C qui affiche d'une manière aléatoire un des jours de la semaine.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<time.h>
4. int main()
5. {
6.     int a;
7.     srand(time(NULL));
8.     a = rand();
9.     a = a%7;
10.    switch(a)
11.    {
12.         case 0: printf("Lundi\n");
13.             break;
14.         case 1: printf("Mardi\n");
```

```
15.                 break;
16.           case 2: printf("Mercredi\n");
17.                 break;
18.           case 3: printf("Jeudi\n");
19.                 break;
20.           case 4: printf("Vendredi\n");
21.                 break;
22.           case 5: printf("Samedi\n");
23.                 break;
24.           case 6: printf("Dimanche\n");
25.                 break;
26.      }
27.      system("pause");
28.      return 0;
29. }
```

BOUCLES

Exercice

Ecrire un programme C qui définit un nombre magique (un nombre secret), et lit des entiers à l'entrée jusqu'à ce que l'utilisateur trouve ce nombre. En lui indiquant à chaque fois s'il est en dessus ou au-dessous du nombre magique.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
```

```
4. {
5.     int mag,a;
6.     mag=12;
7.     printf("Tentez votre chance:\n");
8.     do{
9.        scanf("%d",&a);
10.       if(a<mag) printf("Pas encore! essayez un nombre p
  lus grand.\n");
11.       if(a>mag) printf("Pas encore! essayez un nombre p
  lus petit.\n");
12.       if(a==mag) printf("Bravo! vous avez trouver le no
  mbre magique.\n");
13.    }while(a!=mag);
14.    system("pause");
15.    return 0;
16.}
```

Exercice

Ecrire un programme C qui lit un entier puis détermine s'il est premier ou non.
On rappelle qu'un entier est dit premier s'il a exactement deux diviseurs différents; *1* et lui-même.
Ex: *2, 3, 7, 17, 101* sont tous premiers, et *4, 10, 27* ne le sont pas.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     int p,i,pr;
```

```
    printf("Donnez un entier:\n");
    scanf("%d",&p);
    if(p==0 || p==1) printf("%d n'est pas premier.\n",p)
;
    else
    {
        pr=1;
        for(i=2;i<p;i++)
        {
            if(p%i==0) {pr=0; break; }
        }
        if(pr==1) printf("%d est premier.\n",p);
        else printf("%d n'est pas premier.\n",p);
    }
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui lit une série d'entiers positifs inférieurs à *100* terminée par *0*. Et qui doit négliger toute entrée strictement supérieure à *100*. Puis calcule et affiche la somme et le max des éléments de cette série.

Solution

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

```
5.      int som,max,a;
6.      som=0;
7.      max=0;
8.      a=1;
9.      printf("Entrez une serie d'entier, pour finir entrez
    0.\n");
10.     while(a!=0)
11.     {
12.        do{
13.             scanf("%d",&a);
14.             if(a>100) printf("Entrez SVP un entier infer
   ieur ou egal a 100\n");
15.        }while(a>100);
16.        som+=a;
17.        if(a>max) max=a;
18.     }
19.     printf("la somme: %d\n",som);
20.     printf("le max: %d\n",max);
21.     system("pause");
22.     return 0;
23. }
```

Exercice

Ecrire un programme C qui lit un entier et l'affiche inversé. On choisira de ne pas afficher chiffre par chiffre mais de construire l'entier inversé puis l'afficher.

Ex: si l'entrée est *12345* on doit afficher l'entier *54321*.

Solution

```c
#include<stdio.h>
#include<stdlib.h>
int main()
{
    int r,a,b;
    printf("Donner un entier positif:\n");
    scanf("%d",&a);
    b=0;
    while(a>0)
    {
        r=a%10;
        b=10*b+r;
        a=a/10;
    }
    printf("l'inverse de l'entier donne en entrée est %d \n",b);
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui lit un entier puis affiche tous les nombres premiers qui lui sont inférieurs.

Solution

```c
#include<stdio.h>
#include<stdlib.h>
int main()
```

```
{
    int n,i,p,ok;
    printf("Donnez un entier:\n");
    scanf("%d",&n);
    printf("Les nombres premiers inferieurs a %d sont:\n",n);
    for(p=2;p<=n;p++)
    {
        ok=1;
        for(i=2;i<p;i++)
        {
            if(p%i==0) {ok=0; break; }
        }
        if(ok==1) printf("%d\n",p);
    }
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui calcule le $n^{ième}$ terme de la suite de Fibonacci, définie comme suit: $U_n=U_{n-1}+U_{n-2}$ où $U_1=U_0=1$.

Solution

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

```
5.      int n,i,A,B,C;
6.      printf("Entrez un entier:\n");
7.      scanf("%d",&n);
8.      A=1;
9.      B=1;
10.     for(i=2;i<=n;i++)
11.     {
12.        C=B+A;
13.        A=B;
14.        B=C;
15.     }
16.     if(n<=1) printf("U(%d) = 1.\n",n);
17.     else printf("U(%d) = %d.\n",n,C);
18.     system("pause");
19.     return 0;
20. }
```

Exercice

Ecrire un programme C qui utilise le principe de dichotomie pour trouver la solution de l'équation $x^3+12x^2+1=0$ dans l'intervalle $[-15,-10]$ avec une précision de $0,00001$.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.      float a,b,m,fa,fb,fm;
```

```
6.      a=-15;
7.      b=-10;
8.      while(b-a>0.000001)
9.      {
10.        m=a+(b-a)/2;
11.        fa=a*a*a+12*a*a+1;
12.        fb=b*b*b+12*b*b+1;
13.        fm=m*m*m+12*m*m+1;
14.        if(fa*fm<=0) b=m;
15.        else a=m;
16.     }
17.     fa=a*a*a+12*a*a+1;
18.     printf("%.5f %.5f\n",a,fa);
19.     system("pause");
20.     return 0;
21. }
```

TABLEAUX

Exercice

Ecrire un programme C qui lit un entier *n*, puis *n* autres entiers positifs dans un tableau, l'affiche puis calcul la somme, le max, et le min de ses éléments.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
```

```
{
    int t[100];
    int i,n,som,max,min;
    printf("Donnez le nombre des element du tableau:\n")
  ;
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
       printf("Donnez l'element %d:\n",i+1);
       scanf("%d",&t[i]);
    }
    som=0;
    min=t[0];
    max=t[0];
    for(i=0;i<n;i++)
    {
       printf("%d ",t[i]);
       som+=t[i];
       if(t[i]>max) max=t[i];
       if(t[i]<min) min=t[i];
    }
    printf("\n");
    printf("La somme est: %d\n",som);
    printf("Le max est: %d\n",max);
    printf("Le min est: %d\n",min);
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui lit un entier *n*, puis *n* autres éléments dans un tableau. Et, affiche ce dernier avant et après la suppression de toutes les occurrences d'un nombre entré par l'utilisateur.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     int t[100];
6.     int i,j,n,a;
7.     printf("Donnez le nombre des elements du tableau:\n"
   );
8.     scanf("%d",&n);
9.     for(i=0;i<n;i++)
10.    {
11.       printf("Donnez l'element %d:\t",i+1);
12.       scanf("%d",&t[i]);
13.    }
14.    for(i=0;i<n;i++)
15.       printf("%d ",t[i]);
16.    printf("\n");
17.    printf("Donnez l'element a supprimer du tableau:\n")
   ;
18.    scanf("%d",&a);
19.    for(i=0;i<n;i++)
20.    {
21.       if(t[i]==a)
```

```
22.         {
23.             for(j=i;j<n-1;j++)
24.               t[j]=t[j+1];
25.             n--;
26.             i--;
27.         }
28.     }
29.     for(i=0;i<n;i++)
30.         printf("%d ",t[i]);
31.     printf("\n");
32.     system("pause");
33.     return 0;
34. }
```

Exercice

Ecrire un programme C qui lit un entier *n*. Puis *n* autres entiers inférieurs à *100*, dans un tableau. Et affiche le nombre d'occurrences de chaque élément du tableau de la façon suivante:

Si le tableau est: 1 2 5 2 1 2, on affiche:

1	est	répété	2	fois.
2	est	répété	3	fois.

5 est répété 1 fois.

Pas nécessairement dans un ordre précis, mais chaque élément ne doit être cité qu'une seule fois.

Solution

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

```
3. int main()
4. {
5.     int t[100],occ[100];
6.     int i,n;
7.     printf("Donnez la taille du tableau:\n");
8.     scanf("%d",&n);
9.     printf("Entrez %d element(s):\n",n);
10.    for(i=0;i<n;i++)
11.       scanf("%d",&t[i]);
12.    for(i=0;i<100;i++)
13.       occ[i]=0;
14.    for(i=0;i<n;i++)
15.       occ[t[i]]++;
16.    for(i=0;i<100;i++)
17.    {
18.       if(occ[i]!=0)
19.          printf("%d est repete %d fois.\n",i,occ[i]);
20.    }
21.    system("pause");
22.    return 0;
23.}
```

Exercice

Ecrire un programme C qui construit la table de multiplication des entiers entre 1 et 9 puis l'affiche.

Solution

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

```c
#include<stdlib.h>
int main()
{
    int t[9][9];
    int i,j;
    for(i=0;i<9;i++)
    {
       for(j=0;j<9;j++)
       {
          t[i][j]=(i+1)*(j+1);
       }
    }
    for(i=0;i<9;i++)
    {
       for(j=0;j<9;j++)
       {
          printf("%2d ",t[i][j]);
       }
       printf("\n");
    }
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui lit un entier *n* inférieur à *10*, et une matrice (tableau à deux dimensions) carrée *nxn*. Puis l'affiche et vérifie si tous les entiers entre *1* et n^2 y sont présents ou non.

Ex: la matrice *3x3* suivante vérifie cette contrainte.

```
2 5 3
1 9 6
7 4 8
```

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     int t[10][10],existe[101];
6.     int i,j,n,k;
7.     printf("Donnez un entier inferieur a 10:\n");
8.     scanf("%d",&n);
9.     printf("Entrez les elements de la matrice:\n");
10.    for(i=0;i<n;i++)
11.    {
12.       for(j=0;j<n;j++)
13.       {
14.          printf("t[%d][%d] = ",i,j);
15.          scanf("%d",&t[i][j]);
16.       }
17.    }
18.    for(i=1;i<=n*n;i++) existe[i]=0;
19.    for(i=0;i<n;i++)
20.    {
21.       for(j=0;j<n;j++)
22.       {
23.          printf("%d ",t[i][j]);
24.          existe[t[i][j]]=1;
```

```
25.         }
26.         printf("\n");
27.     }
28.     k=1;
29.     for(i=1;i<=n*n;i++)
30.     {
31.         if(existe[i]==0)
32.         {
33.             k=0; break;
34.         }
35.     }
36.     if(k==0) printf("les entiers de 1 a %d ne sont pas t
  ous dans la matrice.\n",n*n);
37.     else printf("tout les entiers de 1 a %d se trouvent
  dans la matrice.\n",n*n);
38.     system("pause");
39.     return 0;
40. }
```

Exercice

Ecrire un programme C qui lit un tableau de taille *nxm*, et cherche tous les points-col qui s'y trouvent.

Un point-col est un élément qui est max sa ligne et min sur sa colonne.

Solution

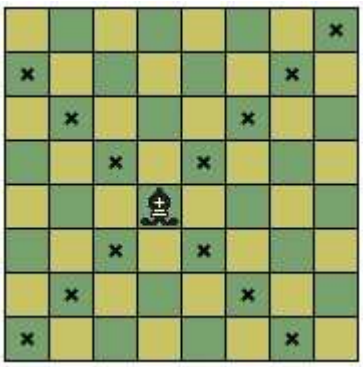
```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
```

```
4. {
5.     int t[100][100],min[100][100],max[100][100];
6.     int i,j,k,n,m;
7.     int MAX,MIN;
8.     printf("Donnez le nombre de lignes et de colonnes:\n"
   );
9.     scanf("%d%d",&n,&m);
10.    printf("Entrez les element du tableau:\n");
11.    for(i=0;i<n;i++)
12.       for(j=0;j<m;j++)
13.       {
14.          scanf("%d",&t[i][j]);
15.          min[i][j]=0;
16.          max[i][j]=0;
17.       }
18.    for(i=0;i<n;i++)
19.    {
20.       MAX=t[i][0];
21.       for(j=1;j<m;j++)
22.       {
23.          if(MAX<t[i][j]) MAX=t[i][j];
24.       }
25.       for(k=0;k<m;k++)
26.          if(t[i][k]==MAX) max[i][k]=1;
27.    }
28.    for(j=0;j<m;j++)
29.    {
30.       MIN=t[0][j];
31.       for(i=1;i<n;i++)
```

```
32.          if(MIN>t[i][j]) MIN=t[i][j];
33.       for(k=0;k<n;k++)
34.          if(t[k][j]==MIN) min[k][j]=1;
35.    }
36.    for(i=0;i<n;i++)
37.       for(j=0;j<m;j++)
38.       {
39.          if(min[i][j]==1 && max[i][j]==1)
40.             printf("La position (%d,%d) est un point-col
   .\n",i,j);
41.       }
42.    system("pause");
43.    return 0;
44. }
```

Exercice

Ecrire un programme C qui lit une position *(i,j)* d'un tableau *8x8*, rempli par des zéros. Et marque toutes les cases des deux diagonales qui passent par cette position en y mettant des uns.



Solution

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

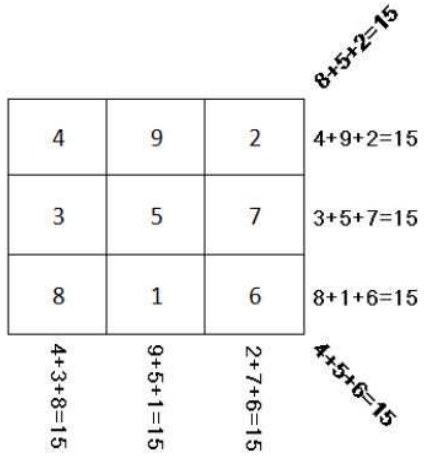
```
#include<stdlib.h>
int main()
{
    int t[8][8];
    int i,j;
    int a,b;
    for(i=0;i<8;i++)
       for(j=0;j<8;j++)
          t[i][j]=0;
    printf("Entrez une position:\n");
    scanf("%d%d",&a,&b);
    i=a;
    j=b;
    while(i>=0 && j>=0)
    {
       t[i][j]=1;
       i--;
       j--;
    }
    i=a;
    j=b;
    while(i<8 && j<8)
    {
       t[i][j]=1;
       i++;
       j++;
    }
    i=a;
    j=b;
```

```
31.     while(i>=0 && j<8)
32.     {
33.        t[i][j]=1;
34.        i--;
35.        j++;
36.     }
37.     i=a;
38.     j=b;
39.     while(i<8 && j>=0)
40.     {
41.        t[i][j]=1;
42.        i++;
43.        j--;
44.     }
45.     for(i=0;i<8;i++)
46.     {
47.        for(j=0;j<8;j++)
48.        printf("%d ",t[i][j]);
49.        printf("\n");
50.     }
51.     system("pause");
52.     return 0;
53. }
```

Exercice

Ecrire un programme C qui lit un tableau *nxn*, et vérifie s'il est magique ou non. Un tableau est dit magique, si la somme des éléments de chaque ligne, de chaque colonne et de chaque une des deux diagonales est la même. En plus il doit contenir tous les entiers

entre 1 et n^2.
L'exemple suivant montre un carré magique de dimension *3x3* :



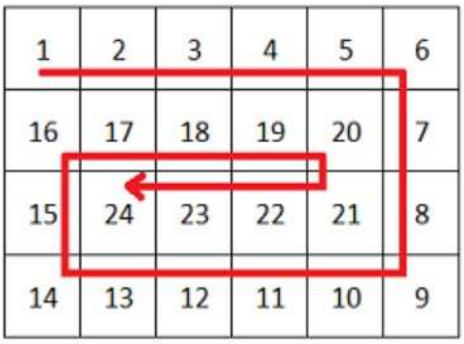
Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     int t[15][15],v[300];
6.     int i,j,n,som_mag,s;
7.     int mag=1;
8.     printf("Donnez la dimension du carre:\n");
9.     scanf("%d",&n);
10.    for(i=0;i<n;i++)
11.       for(j=0;j<n;j++)
12.          scanf("%d",&t[i][j]);
13.    som_mag=0;
14.    for(i=0;i<n;i++) som_mag+=t[i][i];
15.    for(i=0;i<n;i++)
16.    {
```

```
17.         s=0;
18.         for(j=0;j<n;j++) s+=t[i][j];
19.         if(s!=som_mag){ mag=0; break;}
20.         s=0;
21.         for(j=0;j<n;j++) s+=t[j][i];
22.         if(s!=som_mag){ mag=0; break;}
23.     }
24.     s=0;
25.     for(i=0;i<n;i++) s+=t[i][n-i-1];
26.     if(s!=som_mag) mag=0;
27.     if(mag==1)
28.     {
29.         for(i=1;i<=n*n;i++) v[i]=0;
30.         for(i=0;i<n;i++)
31.         {
32.            for(j=0;j<n;j++)
33.               if(v[t[i][j]]==1 || t[i][j]>n*n || t[i][j]
   <1){ mag=0; break; }
34.               else v[t[i][j]]=1;
35.            if(mag==0) break;
36.         }
37.     }
38.     if(mag==1) printf("Le tableau est un carre magique.\
   n");
39.     else printf("Le tableau n'est pas un carre magique.\
   n");
40.     system("pause");
41.     return 0;
42. }
```

Exercice

Ecrire un programme C qui lit les dimensions *n* et *m* d'un tableau à deux dimensions, le remplit d'une manière spirale en utilisant les entiers de l'intervalle *1* à *n*m* comme indiqué sur le tableau ci-dessous:



Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. int main()
4. {
5.     int t[15][15];
6.     int i,j,k,n,m;
7.     int a;
8.     printf("Entrez les dimensions du tableau:\n");
9.     scanf("%d%d",&n,&m);
10.    k=1;
11.    a=0;
12.    while(2>0)
13.    {
14.       i=a;
15.       for(j=a;j<=m-a-1;j++) t[i][j]=k++;
```

```
        if(k>n*m) break;
        j=m-a-1;
        for(i=a+1;i<=n-a-2;i++) t[i][j]=k++;
        i=n-a-1;
        for(j=m-a-1;j>=a;j--) t[i][j]=k++;
        if(k>n*m) break;
        j=a;
        for(i=n-a-2;i>=a+1;i--) t[i][j]=k++;
        a++;
    }
    for(i=0;i<n;i++)
    {
        for(j=0;j<m;j++)
            printf("%4d",t[i][j]);
        printf("\n");
    }
    system("pause");
    return 0;
}
```

CHAÏNE DE CARACTERES

Exercice

Ecrire un programme C qui lit une chaîne de caractères et vérifie si elle est palindrome ou non. On rappelle qu'une chaîne de caractères est dite palindrome, si elle se lit de la même manière dans les deux sens. Exemple: *non*, *touot* et *1234321* sont toutes des chaînes de caractères palindromes.

Solution

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int main()
{
    char s[100];
    int i,j,ok;
    printf("Donnez une chaine de caracteres:\n");
    scanf("%s",s);
    ok=1;
    for(i=0,j=strlen(s)-1;i<j;i++,j--)
    {
       if(s[i]!=s[j])
       {
         ok=0;
         break;
       }
    }
    if(ok==1) printf("%s est palindrome.\n",s);
    else printf("%s n'est pas palindrome.\n",s);
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui lit deux chaînes de caractères et les affiche dans l'ordre alphabétique, en utilisant les deux méthodes suivantes:

- En utilisant la fonction *strcmp*.
- Sans utiliser la fonction *strcmp*.

Par exemple, si on donne en entrée les deux chaînes suivantes: *acb* et *abcd*, le programme doit afficher la chaîne abcd puis *acb*.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<string.h>
4. int main()
5. {
6.         char s1[100],s2[100];
7.         int i;
8.         printf("Donnez deux chaines de caracteres:\n");
9.         gets(s1);
10.        gets(s2);
11.        printf("En utilisant la fonction strcmp:\n");
12.        if( strcmp(s1,s2)<0 )
13.        {
14.            puts(s1);
15.            puts(s2);
16.        }
17.        else
18.        {
```

```
            puts(s2);
            puts(s1);
        }
        printf("Sans utilisation de la fonction strcmp:\
  n");
        for(i=0;i<strlen(s1) && i<strlen(s2); i++)
        {
            if(s1[i]==s2[i]) continue;
            if(s1[i]<s2[i]) { puts(s1); puts(s2);}
            else { puts(s2); puts(s1);}
            break;
        }
        if(i==strlen(s1) )
        {
            puts(s1); puts(s2);
        }
        else if(i==strlen(s2) )
        {
            puts(s2); puts(s1);
        }
        system("pause");
        return 0;
}
```

Exercice

Ecrire un programme C qui lit deux chaînes de caractères et permute leurs contenus en utilisant les deux méthodes suivantes:

• Avec la fonction *strcpy*;

• Sans la fonction *strcpy*.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<string.h>
4. int main()
5. {
6.     char s1[100],s2[100],tmp[100],c;
7.     int i;
8.     printf("Donnez deux chaines de caracteres:\n");
9.     gets(s1);
10.    gets(s2);
11.    printf("En utilisant la fonction strcpy:\n");
12.    strcpy(tmp,s1);
13.    strcpy(s1,s2);
14.    strcpy(s2,tmp);
15.    puts(s1);
16.    puts(s2);
17.    printf("Sans utilisation de la fonction strcpy:\n");
18.    for(i=0;i<=strlen(s1) || i<=strlen(s2); i++)
19.    {
20.        c=s1[i];
21.        s1[i]=s2[i];
22.        s2[i]=c;
23.    }
24.    puts(s1);
25.    puts(s2);
```

```
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui lit une chaîne de caractères, et transforme chaque caractère majuscule en minuscule et vice versa.

Solution

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int main()
{
    char s[100];
    int i;
    gets(s);
    for(i=0;i<strlen(s);i++)
    {
        if( 'a'<=s[i] && s[i]<='z' )
        {
            s[i]=s[i]-'a'+'A';
            continue;
        }
        if( 'A'<=s[i] && s[i]<='Z' )
            s[i]=s[i]-'A'+'a';
    }
    puts(s);
```

```
20.     system("pause");
21.     return 0;
22. }
```

Exercice

Ecrire un programme C qui lit deux chaînes de caractères et vérifie si la deuxième est une sous chaîne de la première ou non. Exemple: *tout* est une sous chaîne de *surtout*.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<string.h>
4. int main()
5. {
6.     char s[100],ch[100];
7.     int i,j,ok,n,m;
8.     gets(s);
9.     gets(ch);
10.    n=strlen(s);
11.    m=strlen(ch);
12.    ok=0;
13.    for(i=0;i<n-m+1;i++)
14.    {
15.       if(s[i]==ch[0])
16.       {
17.         for(j=0;j<m;j++)
18.           if(s[i+j]!=ch[j]) break;
```

```
19.           if(j==m) ok=1;
20.         }
21.         if(ok==1) break;
22.     }
23.     if(ok==1) printf("'%s' est une sous chaine de '%s'.\
  n",ch,s);
24.     else printf("'%s' n'est pas une sous chaine de '%s'.
  \n",ch,s);
25.     system("pause");
26.     return 0;
27. }
```

Exercice

Ecrire un programme C, qui lit une chaîne de caractères représentant une phrase, et affiche dans l'ordre alphabétique toutes les lettres qui ne figurent pas dans cette chaîne de caractères.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<string.h>
4. int main()
5. {
6.     char s[100],c;
7.     int t[26],i,j;
8.     printf("Entrez une chaine de caracteres: \n");
9.     gets(s);
10.    for(i=0;i<26;i++)
11.        t[i] = 0;
```

```
    for(i=0;i<strlen(s);i++)
    {
        c = s[i];
        if( !(c>='a' && c<='z') && !(c>='A' && c<='Z')) continue;
        if(c>='A' && c<='Z') c = 'a'+(c-'A');
        t[(int)(c-'a')] = 1;
    }
    for(i=0;i<26;i++)
    {
        if(t[i]==0) printf(" %c",'a'+i);
    }
    printf("\n");
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui lit une chaîne de caractères et supprime toutes les occurrences d'un caractère entré par l'utilisateur.

Solution

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int main()
{
    char s[100];
```

```
7.      char c;
8.      int i,j;
9.      printf("Entrez une chaine de caracteres:\n");
10.     gets(s);
11.     printf("Entrez le caractere a supprimer: ");
12.     scanf("%c",&c);
13.     for(i=0;i<strlen(s);i++)
14.     {
15.        if(s[i]==c)
16.        {
17.           for(j=i;j<strlen(s);j++)
18.              s[j]=s[j+1];
19.           i--;
20.        }
21.     }
22.     puts(s);
23.     system("pause");
24.     return 0;
25. }
```

Exercice

Ecrire un programme C qui lit une chaîne de caractères puis un ensemble de caractères et affiche le nombre de fois que chacun d'eux a apparu dans cette chaîne.

Solution

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

```
#include<string.h>
int main()
{
    char s[100],ch[100];
    int i,j,n,nbr;
    printf("Entrez une chaine de caracteres:\n");
    gets(s);
    printf("Entrez le nombre de caracteres a traiter: ")
;
    scanf("%d",&n);
    printf("Entrez les caracteres: ");
    for(i=0;i<n;i++)
    {
       getchar();
       scanf("%c",&ch[i]);
    }
    for(i=0;i<n;i++)
    {
       nbr = 0;
       for(j=0;j<strlen(s);j++)
       {
          if(s[j]==ch[i]) nbr++;
       }
       printf("Le caractere %c est repete %d fois.\n",ch
[i],nbr);
    }
    system("pause");
    return 0;
}
```

Exercice

Ecrire un programme C qui lit une chaîne de caractères et calcule le nombre d'occurrences de chacun de ses caractères.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<string.h>
4. int main()
5. {
6.     char s[100];
7.     int t[128];
8.     int i,j;
9.     printf("Donnez une chaine de caracteres:\n");
10.    gets(s);
11.    for(i=0;i<128;i++) t[i]=0;
12.    for(i=0;i<strlen(s);i++)
13.    {
14.       j=(int)s[i];
15.       t[j]++;
16.    }
17.    for(i=0;i<128;i++)
18.    {
19.       if(t[i]!=0)
20.       {
21.          printf("Le caractere %c se repete %d fois.\n",
   (char)i,t[i]);
22.       }
23.    }
```

```
24.     system("pause");
25.     return 0;
26. }
```

Exercice

Ecrire un programme C qui lit une liste de prénoms puis demande à l'utilisateur d'entrer une lettre et affiche tous les prénoms commençant par cette lettre.

Solution

```
1. #include<stdio.h>
2. #include<stdlib.h>
3. #include<string.h>
4. int main()
5. {
6.     char t[100][20];
7.     int i,n;
8.     char c;
9.     printf("Donnez le nombre des elements de la liste: "
   );
10.    scanf("%d",&n);
11.    for(i=0;i<n;i++)
12.    {
13.       printf("Donnez le prenom numero %d: ",i+1);
14.       scanf("%s",t[i]);
15.    }
16.    printf("Donnez un caractere: ");
17.    getchar();
18.    scanf("%c",&c);
```

```
19.     printf("Les prenoms qui commencent par %c sont:\n",c
   );
20.     for(i=0;i<n;i++)
21.     {
22.        if(t[i][0]==c) puts(t[i]);
23.     }
24.     system("pause");
25.     return 0;
26. }
```