```c
//-----------------------------------------------------------------------
// TP 5 exercice 1
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define N 10

int main (void) {
    int i,neg;
    int A[N], B[N];

    srand(time(0));

    //remplissage des tableaux
    for (i=0;i<N;i++) {
        A[i]=i;
        B[i]=rand()%10;
    }

    //affichage des tableaux
    printf("\nA : ");
    for (i=0;i<N;i++) {
        printf("%2d",A[i]);
    }
    printf("\nB : ");
    for (i=0;i<N;i++) {
        printf("%2d",B[i]);
    }

    //comparaison des 2 tableaux element par element
    for (neg=0,i=0;i<N;i++) {
        if (A[i]==B[i])
            neg++;
    }

    //affichage
    printf("\nil y a %d case(s) identique(s)\n",neg);

    return 0;
}

//-----------------------------------------------------------------------
// TP 5 Exercice 2
#include <stdio.h>
#define NMAX    10

int main (void) {
    int i,max,tab[NMAX];

    //saisie du tableau
    printf("entrez les valeurs : ");
    for (i=0;i<NMAX;i++)
        scanf("%d",&tab[i]);

    //affichage dans l'ordre inverse
    for (i=NMAX-1;i>=0;i--)
        printf("%d  ",tab[i]);

    //initialisation du max
    max=tab[0];

    //recherche dans le reste du tableau
    for (i=1;i<NMAX;i++) {
        if (tab[i]>max)
            max=tab[i];
    }

    //affichage du max
    printf("\nle maximum est %d\n",max);

    return 0;
}

```

```c
//-----------------------------------------------------------------------
// TP 5 Exercice 3
#include<stdio.h>
#include <string.h>
#define NMAX 128

int letmin(char); //fonction indiquant si le caractere passe en argument est une lettre minuscule

int main (void) {
  int i,l,n;
  char chaine[NMAX];

  printf("entrez une chaine de caracteres : ");
  gets(chaine);

  //on affiche la chaine
  puts(chaine);

  //longueur de la chaine avec fonction strlen
  l=strlen(chaine);
  printf("votre chaine contient %d caracteres\n",l);

  //longueur de la chaine sans fonction strlen
  for (i=0;i<NMAX;i++)          //on parcourt le tableau
        if (chaine[i]=='\0')    //on sort de la boucle des qu'on voit
            break;              //le caractere de fin de chaine
  printf("votre chaine contient %d caracteres\n",i);

  //nombre de lettres minuscules
  n=0;
  for (i=0;i<l;i++)
    if (letmin(chaine[i]))
        n++;

  printf("votre chaine contient %d lettres minuscules\n",n);

  return 0;
}

int letmin(char c)
{
    if('a'<=c && c<='z')
        return 1;
    else
        return 0;
}

//-----------------------------------------------------------------------
// TP 5 Exercice 4
#include <stdio.h>
#define NMAX    20

//prototypes des fonctions
void LirTab(int[],int); //fonction qui remplit le tableau
void AffTab(int[],int); //fonction qui affiche le talbeau
float MoyTab(int[],int);//fonction qui renvoie la moyenne du tableau
int PosTab(int[],int);  //fonction qui renvoie le nombre d'elements positifs du tableau
int ImaxTab(int[],int); //fonction qui renvoie l'indice du maximum du tableau
int InTab(int[],int,int);//fonction qui indique si une valeur donnée est présente dans le tableau

int main (void) {
    int imax,n,npos,x,present,tab[NMAX];
    float moyenne;

    //saisie du tableau
    do {
        printf("entrez la taille du tableau (entre 1 et %d) : ",NMAX);
        scanf("%d",&n);
    } while (n<1 || n>NMAX);

```

```c
    LirTab(tab,n);
    AffTab(tab,n);
    moyenne=MoyTab(tab,n);
    npos=PosTab(tab,n);
    imax=ImaxTab(tab,n);

    printf("entrez la valeur recherchee : ");
    scanf("%d",&x);
    present=InTab(tab,n,x);

    //affichage
    printf("la moyenne est %f\n",moyenne);
    printf("il y a %d elements strictement positifs\n",npos);
    printf("le max vaut %d, c'est l'element %d du tableau\n",tab[imax],imax+1);
    if (present)
        printf("la valeur %d est presente dans le tableau\n",x);
    else
         printf("la valeur %d n'est pas presente dans le tableau\n",x);

    return 0;
}

//fonction remplissant le tableau tab de taille n
void LirTab(int tab[],int n)
{
    int i;
    printf("entrez les %d valeurs :\n",n);
    for (i=0;i<n;i++)
        scanf("%d",&tab[i]);
}

//fonction affichant le tableau tab de taille n
void AffTab(int tab[],int n)
{
    int i;
    printf("[");
    for (i=0;i<n;i++)
        printf("%d ",tab[i]);
    printf("\b]\n");
}

//fonction renvoyant la moyenne du tableau tab de taille n
float MoyTab(int tab[],int n)
{
    int i,somme;
    for (somme=i=0;i<n;i++) {
        somme+=tab[i];
    }
    return (1.*somme)/n; //on force la division réelle
}
//fonction renvoyant le nombre de valeurs strictement positives
//dans le tableau tab de taille n
int PosTab(int tab[],int n)
{
    int i, npos=0;
    for (i=0;i<n;i++)
        if (tab[i]>0)
            npos++;
    return npos;
}

//fonction renvoyant l'indice du maximum du tableau tab de taille n
int ImaxTab(int tab[],int n)
{
    int i,imax;
    imax=0;
    for (i=0;i<n;i++) {
        if (tab[i]>tab[imax]) {
            imax=i;
        }
    }
    return imax;
}
```

```c

//fonction indiquant si la valeur x est presente
//dans le tableau tab de taille n
int InTab(int tab[],int n,int x)
{
    int i;
    for (i=0;i<n;i++) {
        if (tab[i]==x) {
            return 1;
        }
    }
    return 0; //on a parcouru tout le tableau et
}             //on n'a jamais trouve la valeur cherchee




//-----------------------------------------------------------------------
// TP 5 Exercice 5

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

#define NMAX 128

int main (void) {
    int n,i;
    char radical[NMAX],verbe[NMAX],pc;

    printf("entrez un verbe du premier groupe : ");
    gets(verbe);

    //on verifie qu'il s'agit d'un verbe du 1er groupe
    n=strlen(verbe);
    if ((verbe[n-1]!='r') || (verbe[n-2]!='e') || !strcmp(verbe,"aller")) {
        printf("vous n'avez pas entre un verbe du 1er groupe, au revoir\n");
        exit(0);
    }

    strcpy(radical,verbe);
    radical[n-2]='\0'; //on supprime la terminaison 'er' du verbe

    printf("votre verbe conjugue donne :\n");
    printf("je %se\n",radical);
    printf("tu %ses\n",radical);
    printf("il %se\n",radical);
    printf("nous %sons\n",radical);
    printf("vous %sez\n",radical);
    printf("ils %sent\n",radical);

    return 0;
}
```

```c
//-----------------------------------------------------------------------
// TP 5 Exercice 6

#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#define N 101

//prototypes des fonctions
int de_elec(int);

//programme principal
int main(void)
{
    int i,n,p,T[N],p2;

    srand(time(0));  //initialisation des nombres aleatoires

    do {
        printf("nombre de faces de votre de (compris entre 1 et %d) : ",N-1);
        scanf("%d",&n);
    } while (n<1 || n>=N);

    do {
        printf("nombre total de lancers : ");
        scanf("%d",&p);
        if (p<=0)
            printf("rentrez une valeur positive !!\n\n");
    } while (p<=0);

    //initialisation des compteurs
    for (i=1;i<=n;i++)      //la case T[i] contiendra le nombre de fois
        T[i]=0;             //ou le de est tombe sur la face i

    //boucle sur les lancés de dés
    for (i=0;i<p;i++)
    {
        T[de_elec(n)]++;
    }

    //affichage des resultats
    printf("Sur %d lances de de :\n",p);
    for (i=1;i<=n;i++)
        printf("nombre de %d : %6d  (%2.2f%%)\n",i,T[i],100.*T[i]/p);

    //verification finale
    p2=0;
    for (i=1;i<=n;i++)
        p2+=T[i];
    if (p2!=p)
        printf("des des sont apparus ou ont disparus !!\n");

    return 0;
}

//definition des fonctions
int de_elec(int a) {
    return (1+rand()%a);
}




//-----------------------------------------------------------------------
// TP 5 Exercice 7

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define NMAX 20

```

```c
//prototypes des fonctions
int  Aleat(int min, int max);
void RandTab(int T[], int n, int min, int max);
void AffTab(int T[],int n);
void TriTab(int T[], int n);

int main(void)
{
    int n,min,max,Tab[NMAX];

    do {
        printf("entrez la taille du tableau (entre 1 et %d) : ",NMAX);
        scanf("%d",&n);
    } while (n<1 || n>NMAX);

    printf("valeur min : ");
    scanf("%d",&min);
    do {
        printf("valeur max : ");
        scanf("%d",&max);
    } while (max < min);

    RandTab(Tab,n,min,max);
    printf("Voici le tableau : \n");
    AffTab(Tab,n);
    TriTab(Tab,n);
    printf("Voici le tableau trie : \n");
    AffTab(Tab,n);

    return 0;
}

int  Aleat(int min, int max)
{
    static int i=0;
    if (i==0)
        srand(time(i++));

    return min+rand()%(max-min+1);
}

void RandTab(int T[], int n, int min, int max)
{
    int i;
    for (i=0;i<n;i++)
        T[i]=Aleat(min,max);
}

void AffTab(int T[],int n)
{
    int i;
    printf("\n[");
    for (i=0;i<n;i++)
        printf("%d ",T[i]);
    printf("\b]\n");
}

void TriTab(int T[], int n)
{
    int i,j,jmin,temp;
    for (i=0;i<n-1;i++) {
        //on cherche le minium entre la case i et la fin du tableau
        jmin=i;
        for(j=i+1;j<n;j++)
            if (T[j]<T[jmin])
                jmin=j;
        //on permute T[i] et ce minimum
        temp=T[i];
        T[i]=T[jmin];
        T[jmin]=temp;
    }
}
```

```c
//-----------------------------------------------------------------
// TP 5 Exercice 8

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define NMAX    20
#define NR      10

//prototypes des fonctions
void LirTab(int[],int);          //fonction qui remplit le tableau
void AffTab(int[],int);          //fonction qui affiche le talbeau
float MoyTab(int[],int);         //fonction qui renvoie la moyenne du tableau
int PosTab(int[],int);           //fonction qui renvoie le nombre d'elements positifs du tableau
int ImaxTab(int[],int);          //fonction qui renvoie l'indice du maximum du tableau
int InTab(int[],int,int);        //fonction qui indique si une valeur donnée est présente dans le tableau
int SuppValTab(int[],int,int);   //fonction supprimant une valuer donnée dans le tableau
int MedTab(int[],int);           //fonction renvoyant la mediane du tableau

int main (void) {
    int imax,n,npos,x,present,tab[NMAX],med;
    float moyenne;

    //saisie du tableau
    do {
        printf("entrez la taille du tableau (entre 1 et %d) : ",NMAX);
        scanf("%d",&n);
    } while (n<1 || n>NMAX);

    LirTab(tab,n);
    AffTab(tab,n);
    moyenne=MoyTab(tab,n);
    npos=PosTab(tab,n);
    imax=ImaxTab(tab,n);

    printf("entrez la valeur recherchee : ");
    scanf("%d",&x);
    present=InTab(tab,n,x);

    //affichage
    printf("la moyenne est %f\n",moyenne);
    printf("il y a %d elements strictement positifs\n",npos);
    printf("le max vaut %d, c'est l'element %d du tableau\n",tab[imax],imax+1);
    if (present)
        printf("la valeur %d est presente dans le tableau\n",x);
    else
         printf("la valeur %d n'est pas presente dans le tableau\n",x);

    //calcul et affichage de la mediane
    med=MedTab(tab,n);
    printf("la mediane est %d\n",med);

    //Supression d'une valeur
    printf("quelle valeur voulez vous supprimer du tableau : ");
    scanf("%d",&x);
    n=SuppValTab(tab,n,x);
    printf("Voici le nouveau tableau : \n");
    AffTab(tab,n);

    return 0;
}
```

```c
//fonction remplissant le tableau tab de taille n
void LirTab(int tab[],int n)
{
    int i;
    //printf("entrez les %d valeurs :\n",n);
    srand(time(0));
    for (i=0;i<n;i++)
        //scanf("%d",&tab[i]);
        tab[i]=rand()%(2*NR+1)-NR;  //on remplit le tableau avec des valeurs
                                    //aleatoires entre -NR et NR
}

//fonction affichant le tableau tab de taille n
void AffTab(int tab[],int n)
{
    int i;
    printf("[");
    for (i=0;i<n;i++)
        printf("%d ",tab[i]);
    printf("\b]\n");
}

//fonction renvoyant la moyenne du tableau tab de taille n
float MoyTab(int tab[],int n)
{
    int i,somme;
    for (somme=i=0;i<n;i++) {
        somme+=tab[i];

    }
    return (1.*somme)/n; //on force la division réelle
}

//fonction renvoyant le nombre de valeurs strictement positives
//dans le tableau tab de taille n
int PosTab(int tab[],int n)
{
    int i, npos=0;
    for (i=0;i<n;i++)
        if (tab[i]>0)
            npos++;
    return npos;
}

//fonction renvoyant l'indice du maximum du tableau tab de taille n
int ImaxTab(int tab[],int n)
{
    int i,imax;
    imax=0;
    for (i=0;i<n;i++) {
        if (tab[i]>tab[imax]) {
            imax=i;
        }
    }
    return imax;
}

//fonction indiquant si la valeur x est presente
//dans le tableau tab de taille n
int InTab(int tab[],int n,int x)
{
    int i;
    for (i=0;i<n;i++) {
        if (tab[i]==x) {
            return 1;
        }
    }
    return 0; //on a parcouru tout le tableau et
}             //on n'a jamais trouve la valeur cherchee
```

```c
//fonction renvoyant la mediane du tableau T de taille n
int MedTab(int T[],int n)
{
    int i,med,ninf;

    //on initialise med au minimum du tableau
    med=T[0];
    for (i=1;i<n;i++)
        if (T[i]<med)
            med=T[i];

    while(1) {
        //on compte le nombre de valeurs inferieures a med
        ninf=0;
        for (i=0;i<n;i++)
        {
            if (T[i]<=med)
                ninf++;
        }
        if (ninf>(n/2))
            break; //on sort de la boucle car on a trouve la mediane
        else
            med++; //on incremente med
    }
    return med;
}

//fonction supprimant la valeur v du tableau T de taille n
//et tassant les elements restants
int SuppValTab(int T[], int n, int v)
{
    int i,j;
    for (i=0;i<n;i++)   //on parcourt le tableau
    {
        if (T[i]==v)     //on cherche la valeur a supprimer
        {
            for (j=i;j<n-1;j++)  //on remonte toutes les cases
                T[j]=T[j+1];     //sous la case a supprimer
            n--;    //on decremente la taille du tableau
            i--;    //on decremente i pour retester la case
        }
    }
    return n;   //on renvoie la nouvelle taille du tableau
}               //une fois toutes les cases supprimees
```

```c
//-----------------------------------------------------------------------
// TP 5 Exercice 9
#include<stdio.h>
#include <string.h>

#define NMAX 15

int main (void) {
  int i,n, npoint,ipoint,signe, isnombre;
  char nombre[NMAX];
  double x;

  printf("entrez un nombre : ");
  gets(nombre);

  //Initialisation des differentes variables
  n=strlen(nombre);  //longueur de la chaine
  npoint = 0;        //nombre de '.' ou ','
  isnombre = 1;      //a priori, il s'agit bien d'un nombre
  signe = 1;         //a priori, x sera positif
  x = 0.;            //valeur initiale du nombre

  //Le 1er caractere doit etre un chiffre
  //ou '+' ou '-' ou '.' ou','
  if (nombre[0]>='0' && nombre[0]<='9')
    x=nombre[0]-'0';
  else if (nombre[0]=='+')
    signe = 1;
  else if (nombre[0]=='-')
    signe = -1;
  else if (nombre[0]=='.' || nombre[0]==',') {
    ipoint=0;   //position du point ou virgule
    npoint++;
  }
  else
    isnombre = 0;

  //On teste tous les autres caracteres
  for (i=1;i<n && isnombre == 1;i++) {
    //chaque caractere doit etre un chiffre ou '.' ou ','
    if (nombre[i]>='0' && nombre[i]<='9') {
      x*=10;
      x+=nombre[i]-'0';
    }
    else if (nombre[i]==',' || nombre[i] =='.') {
      if (npoint==0) { //il ne doit y avoir que un
        npoint++;      //seul point ou virgule
        ipoint=i;      //position du point ou virgule
      }
      else
        isnombre = 0 ;
    }
    else
      isnombre = 0;
  }

  //Si la chaine entree est un nombre,
  //on le calcule et on l'affiche
  if (isnombre == 1) {
    if (npoint == 1) //on a trouvé une virgule
      for (i=n-1;i>ipoint;i--) // on divise x par 10 autant de fois
          x/=10;               //qu'il y a de chiffres apres la virgule

    x*=signe;       //on change le signe de x si nécessaire

    printf("le nombre rentre vaut : %le\n",x);
  }
  else {
    printf("ceci n'est pas un nombre valide\n");
  }

  return 0;
}
```