Trouver un élément

```
// trouver si une val est dans le tab
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//-----------------------------définition d'un tableau test
int dim_Tab = 5;
int Tab[] = {1,2,3,4,5};
//-------------------------------------------------------------

//-----------------------------Rentrer la valeur à chercher
int val_cherchee;
printf("rentrer la valeur a chercher\n");
scanf("%d",&val_cherchee);
//-------------------------------------------------------------


//-----------------------------Recherche d'un élément
int i=0;
int trouve= FALSE; // ne pas oublié d'initialiser
for (i=0;i<dim_Tab;i++){
  if (Tab[i]==val_cherchee){
     trouve = TRUE;
  }
}
//-----------------------------Affiche si trouve ou pas
if (trouve){
          printf ("la valeur %d est dans le
tableau\n",val_cherchee);
          }
else{
          printf ("la valeur %d n'est pas dans le
tableau\n",val_cherchee);
    }


//-------------------------------------------------------------
 system("PAUSE");
 return 0;
}


/* attention :
          -La boucle for est à bannir ici car dès que vous avez
          trouvé un element on "devrait" stopper la recherche.
          Preferer While (not trouve && pas fin_du_tableau)
          -la boucle for est a garder si on cherche le nombre
d'occurence
          cad le nombre de fois que val est dans le tableau.
          -la variable val_cherchee (avec ee) est mal definie ;
          preferez val tout simplement.
*/
```

trouver si une val est dans le tab avec while

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

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//-----------------------------définition d'un tableau test
int dim_Tab = 5;
int Tab[] = {1,2,3,4,5};
//-------------------------------------------------------------

//-----------------------------Rentrer la valeur à chercher
int val_cherchee;
printf("rentrer la valeur a chercher\n");
scanf("%d",&val_cherchee);
//-------------------------------------------------------------


//-----------------------------Recherche d'un élément
int i=0;
int trouve= FALSE; // ne pas oublié d'initialiser
while (!trouve && i<dim_Tab){
  if (Tab[i]==val_cherchee){
     trouve = TRUE;
  }
  i++; // pour faire avancer la boucle
}
//-----------------------------Affiche si trouve ou pas
if (trouve){
          printf ("la valeur %d est dans le
tableau\n",val_cherchee);
          }
else{
          printf ("la valeur %d n'est pas dans le
tableau\n",val_cherchee);
    }


//-------------------------------------------------------------
 system("PAUSE");
 return 0;
}


/* attention :
          La boucle while stop des que l'element est trouve
          on ne peut compter le nombre d'occurrences
*/
```

Envoyer l’indice de l’element chercher

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

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//------------------------------définition d'un tableau test
int dim_Tab = 5;
int Tab[] = {1,2,3,4,5};
//-------------------------------------------------------------

//------------------------------Rentrer la valeur à chercher
int val_cherchee;
printf("rentrer la valeur a chercher\n");
scanf("%d",&val_cherchee);
//-------------------------------------------------------------


//------------------------------Recherche d'un élément
int i=0;
int trouve= FALSE; // ne pas oublié d'initialiser
while (!trouve && i<dim_Tab){
   if (Tab[i]==val_cherchee){
      trouve = TRUE;
   }
   i++; // pour faire avancer la boucle
}
//------------------------------Affiche si trouve ou pas
if (trouve){
          printf ("la valeur %d est dans le tableau a l'indice %d\n",val_cherchee,i-1);
          }
else{
          printf ("la valeur %d n'est pas dans le tableau\n",val_cherchee);
     }


//-------------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

Ecrire plus compact

```
// trouver si une val est dans le tab avec while
#include <stdio.h>
#include <stdlib.h>



int main(int argc, char *argv[])
{
//----------------------------définition d'un tableau test
int dim_Tab = 5;
int Tab[] = {1,2,3,4,5};
//-----------------------------Rentrer la valeur à chercher
int val_cherchee;
printf("rentrer la valeur a chercher\n");
scanf("%d",&val_cherchee);
//-----------------------------Recherche d'un élément
int i=0;
while (i<dim_Tab && Tab[i++]!=val_cherchee){
}
if (i<dim_Tab){
   printf ("la valeur %d est dans le tableau\n",val_cherchee);
}
else{
     printf ("la valeur %d n'est pas dans le
tableau\n",val_cherchee);
}
//------------------------------------------------------------
  system("PAUSE");
  return 0;
}
//------------------------------------------------------------
#include <stdio.h>
#include <stdlib.h>


int main(int argc, char *argv[])
{
int dim_Tab = 5;
int Tab[] = {1,2,3,4,5};
int test(int i){
         printf("-> dans test i= %d\t",i);

         printf(" Tab %d\t\n",Tab[i]);
   return 1;
}
//----------------------------définition d'un tableau test

//-----------------------------Rentrer la valeur à chercher
int val_cherchee;
printf("rentrer la valeur a chercher\n");
scanf("%d",&val_cherchee);
//-----------------------------Recherche d'un élément
int i=0;
while (i<dim_Tab && test(i) && Tab[i++]!=val_cherchee ){
}
if (i<dim_Tab){
   printf ("la valeur %d est dans le tableau\n",val_cherchee);
}
else{
     printf ("la valeur %d n'est pas dans le
tableau\n",val_cherchee);
}
//------------------------------------------------------------
  system("PAUSE");
  return 0;
}
#include <stdio.h>
#include <stdlib.h>


int main(int argc, char *argv[])
{
int dim_Tab = 5;
int Tab[] = {1,2,3,4,5};
int test(int i){
         printf("-> dans test i= %d\t",i);

         printf(" Tab %d\t\n",Tab[i]);
   return 1;
}
//----------------------------définition d'un tableau test

//-----------------------------Rentrer la valeur à chercher
int val_cherchee;
printf("rentrer la valeur a chercher\n");
scanf("%d",&val_cherchee);
//-----------------------------Recherche d'un élément
int i=0;
while (i<dim_Tab  && Tab[i++]!=val_cherchee && test(i)){
}
if (i<dim_Tab){
   printf ("la valeur %d est dans le tableau\n",val_cherchee);
}
else{
     printf ("la valeur %d n'est pas dans le
tableau\n",val_cherchee);
}
//------------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

Trouver le nombre d’occurrences d’une valeur

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

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//-----------------------------définition d'un tableau test
int dim_Tab = 5;
int Tab[] = {1,2,1,2,2};
//-------------------------------------------------------------

//-----------------------------Rentrer la valeur à chercher
int val_cherchee;
printf("rentrer la valeur a chercher\n");
scanf("%d",&val_cherchee);
//-------------------------------------------------------------


//-----------------------------Recherche d'un élément-----------------------------------------------------------------------------------------
int i=0;
int nb=0; // ne pas oublié d'initialiser
for (i=0;i<dim_Tab;i++){
   if (Tab[i]==val_cherchee){
      nb++;
   }
}
//-------------------------------Affiche si trouve ou pas
if (nb){
          printf ("la valeur %d est dans le tableau %d fois \n",val_cherchee,nb);
          }
else{
          printf ("la valeur %d n'est pas dans le tableau\n",val_cherchee);
    }


//-------------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

Remplacer les occurrences suivante par -1

```
/*µremplacer des la deuxième occurence la val par -1*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//-----------------------------définition d'un tableau test
int dim_Tab = 5;
int Tab[] = {1,2,1,2,2};
int i;

//-----------------------------Rentrer la valeur à chercher
int val_cherchee;
printf("rentrer la valeur a chercher\n");
scanf("%d",&val_cherchee);

//-----------------------------affichage du tableau avant
 printf ("Voici le tableau avant le remplacement\n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
              printf ("%d\t",Tab[i]);
          }
          printf ("]\n");

//-----------------------------Recherche d'un élément----------------------------------------------------------------------------------------

int nb=0; // ne pas oublié d'initialiser
for (i=0;i<dim_Tab;i++){
   if (Tab[i]==val_cherchee){
      if (nb==0) {
        nb++; // c'est la premiere occurence
      }
      else {
          nb++; // c'est au moins la 2 occurence
          Tab[i]=-1; // on remplace
      }
      // si pas la valeur on continue
   }
}
//-------------------------------Affiche si trouve ou pas
if (nb){
          printf ("la valeur %d est dans le tableau %d fois \n\n",val_cherchee,nb);
          printf ("Voici le tableau apres le remplacement\n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
              printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
}
else{
          printf ("la valeur %d n'est pas dans le tableau\n",val_cherchee);
     }


//------------------------------------------------------------
  system("PAUSE");
  return 0;
}


/* attention :
          On pourrait aussi utiliser while i<dim_tab
*/
```

Remplace pour tt les valeurs les suivantes à -1

```
/*µremplacer des la deuxième occurence la val par -1*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//------------------------------définition d'un tableau test
int dim_Tab = 5;
int Tab[] = {1,1,2,1,2};
int i,j;
int val_cherchee;

//-------------------------------affichage du tableau avant
 printf ("Voici le tableau avant le remplacement\n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
              printf ("%d\t",Tab[i]);
          }
          printf ("]\n");

//--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
for (j=0; j<dim_Tab;j++){
   if (Tab[j]!=-1){
      val_cherchee=Tab[j];
      int nb=0; // ne pas oublié d'initialiser
      for (i=0;i<dim_Tab;i++){
          if (Tab[i]==val_cherchee){
              if (nb==0) {
                 nb++; // c'est la premiere occurence
                                          }
              else {
                 nb++; // c'est au moins la 2 occurence
                 Tab[i]=-1; // on remplace
              }
          } // si pas la valeur on continue
       }
   }//la valeur vaut -1 on ne fait rien
}//on recherche la val suivante
//-------------------------------Affiche si trouve ou pas

     printf ("Voici le tableau apres le remplacement\n");
     printf ("tab = [");
     for (i=0;i<dim_Tab;i++){
         printf ("%d\t",Tab[i]);
     }
     printf ("]\n");



//-----------------------------------------------------------
  system("PAUSE");
  return 0;
}


/* attention :
          On pourrait aussi utiliser while i<dim_tab
*/
```

Recherche un code en début de tableau

```
/*recherche si un code est au debut d'un tab tab
Attention on suppose ici les tests de dim verifier
a savoir que le code à chercher est inferieur au tab et que le reste de la div des dim est zéro!*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//------------------------------définition d'un tableau test
int dim_Code = 2;
int Code[] = {1,2};
int dim_Tab = 5;
int Tab[] = {1,2,1,2,2,3};
int i;
int trouve = FALSE;

//------------------------------Rentrer la valeur à chercher
// on pourrait demander à l'utilisateur le code à trouver

//------------------------------affichage du tableau avant
 printf ("Voici le tableau \n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
              printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
  printf ("Voici le code \n");
          printf ("Code = [");
          for (i=0;i<dim_Code;i++){
              printf ("%d\t",Code[i]);
          }
          printf ("]\n");

//------------------------------Recherche d'un code en début----------------------------------------------------------------------------------------------
i = 0;
while (i<dim_Code && Code[i]==Tab[i]){
      i++;
}
if (i==dim_Code){
   trouve= TRUE;
}


//------------------------------Affiche si trouve ou pas
if (trouve){
          printf ("le code est present en debut");

}
else{
          printf ("le code n'est pas present");
     }


//------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

Attention a ne pas metre le test des valeur avant le test des indice.

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

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//------------------------------définition d'un tableau test
int dim_Code = 2;
int Code[] = {1,2};
int dim_Tab = 5;
int Tab[] = {1,2,1,2,2,3};
int i;
int trouve = FALSE;

//------------------------------Rentrer la valeur à chercher
// on pourrait demander à l'utilisateur le code à trouver

//------------------------------affichage du tableau avant
 printf ("Voici le tableau \n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
              printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
  printf ("Voici le code \n");
          printf ("Code = [");
          for (i=0;i<dim_Code;i++){
              printf ("%d\t",Code[i]);
          }
          printf ("]\n");
//
int test(i){
     printf(" i = %d",i);
     printf(" Tab = %d",Tab[i]);
     printf(" Code = %d\n",Code[i]);
     return (Tab[i]==Code[i]);
}


//------------------------------Recherche d'un code en début----------------------------------------------------------------------------------------
i = 0;
while ( test(i) && i<dim_Code){
     i++;
}
if (i==dim_Code){
   trouve= TRUE;
}


//------------------------------Affiche si trouve ou pas
if (trouve){
          printf ("le code est present en debut");

}
else{
          printf ("le code n'est pas present");
    }


//------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

Recherche un code à une position quelconque

```
/* Attention on suppose ici les tests de dim verifier
a savoir que le code à chercher est inferieur au tab et que le reste de la div des dim est zéro!
et que l'indice de recherche est inf à dim-(dim-code)*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//------------------------------définition d'un tableau test
int dim_Code = 2;
int Code[] = {1,2};
int dim_Tab = 6;
int Tab[] = {1,2,1,2,2,3};
int i;
int indice_recherche = 0;
int trouve = FALSE;

//------------------------------Demander la position de recherche
 printf ("Rentrez la position ou on recherche le code \n");
 scanf("%d",&indice_recherche);

//------------------------------affichage du tableau avant
 printf ("Voici le tableau \n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
              printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
  printf ("Voici le code \n");
          printf ("Code = [");
          for (i=0;i<dim_Code;i++){
              printf ("%d\t",Code[i]);
          }
          printf ("]\n");

//------------------------------Recherche d'un code----------------------------------------------------------------------------------------
i = 0;
while (i<dim_Code && Code[i]==Tab[i+indice_recherche]){
      i++;
}
if (i==dim_Code){
   trouve= TRUE;
}


//------------------------------Affiche si trouve ou pas
if (trouve){
          printf ("le code est présent en position %d\n",indice_recherche);

}
else{
          printf ("le code n'est pas en position %d\n",indice_recherche );
     }


//------------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

Recherche si un code est dans un tableau à partir d'une position quelconque.

```
/*
Attention on suppose ici les tests de dim verifier
a savoir que le code à chercher est inferieur au tab et que le reste de la div des dim est zéro!
et que l'indice de recherche est inf à dim-(dim-code)*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//------------------------------définition d'un tableau test
int dim_Code = 2;
int Code[] = {1,2};
int dim_Tab = 5;
int Tab[] = {1,2,1,2,2,3};
int i;
int indice_recherche = 0;
int trouve = FALSE;

//------------------------------Demander la position de recherche
 printf ("Rentrez la position a partir de laquelle on recherche le code \n");
 scanf("%d",&indice_recherche);

//------------------------------affichage du tableau avant
 printf ("Voici le tableau \n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
              printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
  printf ("Voici le code \n");
          printf ("Code = [");
          for (i=0;i<dim_Code;i++){
              printf ("%d\t",Code[i]);
          }
          printf ("]\n");

//------------------------------Recherche d'un code----------------------------------------------------------------------------------------
i = 0;
while (!trouve && indice_recherche <= (dim_Tab - dim_Code)){
  while (i<dim_Code && Code[i]==Tab[i+indice_recherche]){
          i++;
  }
  if (i==dim_Code){
     trouve= TRUE;
  }
  indice_recherche++;
}


//------------------------------Affiche si trouve ou pas
if (trouve){
          printf ("le code est présent en position %d\n",indice_recherche-1);

}
else{
          printf ("le code n'est pas present\n");
     }
//------------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

Recherche un code est donne sa position

```
/*
Attention on suppose ici les tests de dim verifier
a savoir que le code à chercher est inferieur au tab et que le reste de la div des dim est zéro!
et que l'indice de recherche est inf à dim-(dim-code)*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//-----------------------------définition d'un tableau test
int dim_Code = 2;
int Code[] = {1,2};
int dim_Tab = 6;
int Tab[] = {5,2,1,2,2,3};
int i;
int indice_recherche = 0;
int trouve = FALSE;

//-----------------------------affichage du tableau avant
 printf ("Voici le tableau \n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
             printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
  printf ("Voici le code \n");
          printf ("Code = [");
          for (i=0;i<dim_Code;i++){
             printf ("%d\t",Code[i]);
          }
          printf ("]\n");

//-----------------------------Recherche d'un code en début----------------------------------------------------------------------------------------
i = 0;
indice_recherche=0; // force la recherche à commencer tt seule
while (!trouve && indice_recherche < (dim_Tab - dim_Code)){
  while (i<dim_Code && Code[i]==Tab[i+indice_recherche]){
          i++;
  }
  if (i==dim_Code){
     trouve= TRUE;
  }
  indice_recherche++;
}


//-------------------------------Affiche si trouve ou pas
if (trouve){
          printf ("le code est présent en position %d\n",indice_recherche-1);

}
else{
          printf ("le code n'est pas en position\n");
     }


//------------------------------------------------------------
  system("PAUSE");
  return 0;
}


/* attention :
          On pourrait aussi utiliser while i<dim_tab
```

```
*/
```

Recherche le nombre de fois que le code est dans un tab de façon imbriquée

```
/*nb fois un code est dans un tableau (code imbriqué)
Attention on suppose ici les tests de dim verifier
a savoir que le code à chercher est inferieur au tab et que le reste de la div des dim est zéro!
et que l'indice de recherche est inf à dim-(dim-code)*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//------------------------------définition d'un tableau test
int dim_Code = 2;
int Code[] = {1,1};
int dim_Tab = 5;
int Tab[] = {1,1,1,2,2,3};
int i;
int indice_recherche = 0;
int trouve = FALSE;
int nb;

//------------------------------Demander la position de recherche
 printf ("Rentrez la position a partir de laquelle on recherche le code \n");
 scanf("%d",&indice_recherche);

//------------------------------affichage du tableau avant
 printf ("Voici le tableau \n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
              printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
  printf ("Voici le code \n");
          printf ("Code = [");
          for (i=0;i<dim_Code;i++){
              printf ("%d\t",Code[i]);
          }
          printf ("]\n");

//------------------------------Recherche d'un code ----------------------------------------------------------------------------------------

nb=0;
while (indice_recherche <= (dim_Tab - dim_Code)){
  i = 0; // attention a bien initialiser la variable de recherche
  while (i<dim_Code && Code[i]==Tab[i+indice_recherche]){
          i++;
  }
  if (i==dim_Code){
     nb++;
  }
  indice_recherche++;
}


//------------------------------Affiche si trouve ou pas
if (nb){
          printf ("le code est present  %d fois \n",nb);

}
else{
          printf ("le code n'est pas present\n");
     }


//------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

Trouver l'erreur dans ce code qui cherche le nombre d'occurrences du code dans un tableau est donne 1 fois

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

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//-----------------------------définition d'un tableau test
int dim_Code = 2;
int Code[] = {1,1};
int dim_Tab = 6;
int Tab[] = {1,1,1,2,1,1};
int i;
int indice_recherche = 0;
int trouve = FALSE;
int nb;
//-----------------------------Demander la position de recherche
 printf ("Rentrez la position a partir de laquelle on recherche le
code \n");
 scanf("%d",&indice_recherche);

//-----------------------------affichage du tableau avant
 printf ("Voici le tableau \n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
             printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
  printf ("Voici le code \n");
          printf ("Code = [");
          for (i=0;i<dim_Code;i++){
             printf ("%d\t",Code[i]);
          }
          printf ("]\n");
//-----------------------------Recherche d'un code
nb=0;
while (indice_recherche <= (dim_Tab - dim_Code)){
  i = 0; // attention a bien initialiser la variable de recherche
  while (i<dim_Code && Code[i]==Tab[i+indice_recherche]){
          i++;
  }
  if (i==dim_Code){
     nb++;
  }
  // si imbrique indice++ sinon on saute de dim_code-1 indice
  indice_recherche+=dim_Code;
  printf("indice_recherche %d", indice_recherche);
}


//-----------------------------Affiche si trouve ou pas
if (nb){
          printf ("le code est present  %d fois \n",nb);

}
else{
          printf ("le code n'est pas present\n");
    }


//------------------------------------------------------
  system("PAUSE");
  return 0;
```

correction

```
    }
}/*nb fois un code est dans un tableau (code imbriqué) a savoir
11 dans 111 = 1 fois
Attention on suppose ici les tests de dim verifier
a savoir que le code à chercher est inferieur au tab et que le reste
de la div des dim est zéro!
et que l'indice de recherche est inf à dim-(dim-code)*/

#include <stdio.h>
#include <stdlib.h>
#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0
int main(int argc, char *argv[])
{
//-----------------------------définition d'un tableau test
int dim_Code = 2;
int Code[] = {1,1};
int dim_Tab = 6;
int Tab[] = {1,1,1,1,1,1};
int i;
int indice_recherche = 0;
int trouve = FALSE;
int nb;
//-----------------------------Demander la position de recherche
 printf ("Rentrez la position a partir de laquelle on recherche le
code \n");
 scanf("%d",&indice_recherche);
//-----------------------------affichage du tableau avant
 printf ("Voici le tableau \n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
             printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
  printf ("Voici le code \n");
          printf ("Code = [");
          for (i=0;i<dim_Code;i++){
             printf ("%d\t",Code[i]);
          }
          printf ("]\n");
//-----------------------------Recherche d'un code en début
nb=0;
while (indice_recherche <= (dim_Tab - dim_Code)){
  i = 0; // attention a bien initialiser la variable de recherche
  while (i<dim_Code && Code[i]==Tab[i+indice_recherche]){
          i++;
  }
  if (i==dim_Code){
     nb++;
     indice_recherche+=dim_Code;
  }
  else{
      indice_recherche++;
  }
  // si imbrique indice++ sinon on saute de dim_code-1 indice

  printf("indice_recherche %d", indice_recherche);
}
//-----------------------------Affiche si trouve ou pas
if (nb){
          printf ("le code est present  %d fois \n",nb);
}
else{
          printf ("le code n'est pas present\n");
//------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

Elimine toutes les occurrences suivante du code dans un tableau

```
/*nb fois un code est dans un tableau (code imbriqué) a savoir 11
dans 111 = 1 fois
On remplace les occurence suivante du code
Attention on suppose ici les tests de dim verifier
a savoir que le code à chercher est inferieur au tab et que le reste
de la div des dim est zéro!
et que l'indice de recherche est inf à dim-(dim-code)*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//-----------------------------définition d'un tableau test
int dim_Code = 2;
int Code[] = {1,1};
int dim_Tab = 6;
int Tab[] = {1,1,1,0,1,1};
int i;
int indice_recherche = 0;
int trouve = FALSE;
int nb;

//-----------------------------Demander la position de recherche
 printf ("Rentrez la position a partir de laquelle on recherche le
code \n");
 scanf("%d",&indice_recherche);

//-----------------------------affichage du tableau avant

  printf ("Voici le code \n");
          printf ("Code = [");
          for (i=0;i<dim_Code;i++){
              printf ("%d\t",Code[i]);
          }
          printf ("]\n");
  printf ("Voici le tableau \n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
              printf ("%d\t",Tab[i]);
          }
          printf ("]\n");

//-----------------------------Recherche d'un code
nb=0;
int j=0;
while (indice_recherche <= (dim_Tab - dim_Code)){
  i = 0; // attention a bien initialiser la variable de recherche
  while (i<dim_Code && Code[i]==Tab[i+indice_recherche]){
          i++;
  }
  if (i==dim_Code){
     nb++;
     indice_recherche+=dim_Code;
// on teste si ce n'est pas la premiere fois et on remplace
     if (nb>1){
     printf("---------- Elimination de la %d occurence\n", nb);
     printf("de l'indice =%d",indice_recherche-dim_Code);
     printf(" a l'indice =%d\n",indice_recherche-1);
    // on met les dim_code indice précedent à -1
        for (i=indice_recherche-
dim_Code;i<indice_recherche;i++){
            Tab[i]=-1;
        }
     }
  }
  else{
      indice_recherche++;
  }
  // si imbrique indice++ sinon on saute de dim_code-1 indice

  //printf("indice_recherche %d", indice_recherche);
}

//-----------------------------Affiche si trouve ou pas
if (nb){
          printf ("le code est present  %d fois \n",nb);
          printf ("Voici le tableau apres elimination\n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
              printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
}
else{
          printf ("le code n'est pas present\n");
    }
//------------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

Trouve une sequence non contigue 123 12233 ->ok

```
/*Trouve une sequence non contigue 123 12233 ->ok
 */
/*on recherche chaque element et on donne sa position, on
recommence a chercher l'element suivant a partir de cette
position tt que depasse pas la capa*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//----------------------------définition d'un tableau test----------------
--------------------------------------------------------------------------
int dim_Tab = 5;
int Tab[5] = {0,2,3,1,4};
int dim_Code = 2;
int Code[2] = {1,4};
int val_cherchee;

int pos_recherche=0;
int indice_code=0;
int trouve= TRUE;
while (trouve && indice_code<dim_Code){
   // TT que on trouve et qu'il existe des elements àa chercher
   printf("-- Recherche de %d ----------\n",Code[indice_code]);
   printf("------------------------ en position %d\n",pos_recherche);
    trouve= FALSE; // je n'ai pas encore trouve l'ele
    val_cherchee=Code[indice_code];
 //   printf("la valeur recherche = %d\n",val_cherchee);
    while (!trouve && pos_recherche<dim_Tab){
      if (Tab[pos_recherche]==val_cherchee){
        printf("== trouve en position %d\n\n",pos_recherche);
        trouve = TRUE; //permet de passer à l'element suivant
      }
      pos_recherche++; // pour faire avancer la boucle
      if (!trouve){
        printf("------------------------ en position
%d\n",pos_recherche);
      }
    }
    indice_code++; // recherche l'element suivant
// affichage resultat intermediaire
    if (!trouve){
          printf(" la valeur n'est pas dans le tableau\n\n");
    }
}


//-------------------------------Affiche si trouve ou pas
if (trouve){
          printf ("\n\n\nBILAN / le code est present\n\n\n\n");
          }
else{
          printf ("\n\n\nBILAN / le code n'est pas present\n\n\n\n");
     }

//------------------------------------------------------------
  system("PAUSE");
  return 0;
}
```

```
/*Cherche code non contigu avec while*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//----------------------------définition d'un tableau test
int dim_Code = 3;
int Code[] = {1,2,3};
int dim_Tab = 5;
int Tab[] = {1,1,2,1,3};
int indice_ele,indice_recherche;
int trouve = TRUE;
int i,j;
//----------------------------affichage du tableau avant
 printf ("Voici le tableau \n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
             printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
  printf ("Voici le code \n");
          printf ("Code = [");
          for (i=0;i<dim_Code;i++){
             printf ("%d\t",Code[i]);
          }
          printf ("]\n");

//----------------------------Recherche d'un code en début
indice_ele=indice_recherche=0;

while (trouve && indice_ele<dim_Code){
     printf("-------Recherche element = %d -- en position %d -----
----\n",Code[indice_ele],indice_recherche);
     trouve= FALSE; //on recherche l'ele courant
     while (Code[indice_ele]!=Tab[indice_recherche] &&
indice_recherche<dim_Tab-(dim_Code-indice_ele)){
          indice_recherche++; // pas trouve l'el, au suivant
          printf("-------Recherche element = %d -- en position %d -
--------\n",Code[indice_ele],indice_recherche);
     }
//      if (indice_recherche<=dim_Tab-(dim_Code-indice_ele)){ //
Attentio ne marche pas
       if (Code[indice_ele]==Tab[indice_recherche]){//utiliser ce
test
        trouve=TRUE;
        printf("Present en position %d \n", indice_recherche);
        indice_recherche++; // prepare la position suivante
     }
/*      else{
         printf("le test est faux %d\n",dim_Tab-(dim_Code-
indice_ele));
         }
*/
     indice_ele++;       // recherche l'el suivant
}
//-------------------------------Affiche si trouve ou pas
if (trouve){
          printf ("\n----------------------------le code est present \n");

}
else{
          printf ("\n\n\n==============================le
code n'est pas present\n");
 }
  system("PAUSE");
  return 0;
}
```

Trouve nb d’occurences

```
/*Trouve nb fois une sequence non contigue 123 12233 ->ok
 */
/*on recherche chaque element et on donne sa position, on
recommence a chercher l'element suivant a partir de cette
position tt que depasse pas la capa*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//-----------------------------définition d'un tableau test----------------
--------------------------------------------------------------------------
int dim_Tab = 5;
int Tab[5] = {1,4,1,1,4};
int dim_Code = 2;
int Code[2] = {1,4};
int val_cherchee;

int pos_recherche=0;
int indice_code=0;
int trouve= TRUE;
int nb=0;
int derniere_pos;

// pour tt les positions
while (pos_recherche <= (dim_Tab - dim_Code)){
   printf("\n ========================== %d
==================================\n",nb);
   indice_code=0; // on initialise une nouvelle recherche
   while (trouve && indice_code<dim_Code){
     // TT que on trouve et qu'il existe des elements a chercher
     printf("-- Recherche de %d ----------\n",Code[indice_code]);
     printf("------------------------- en position
%d\n",pos_recherche);
      trouve= FALSE; // je n'ai pas encore trouve l'ele
      val_cherchee=Code[indice_code];
    //   printf("la valeur recherche = %d\n",val_cherchee);
      while (!trouve && pos_recherche<dim_Tab){
        if (Tab[pos_recherche]==val_cherchee){
          printf("== trouve en position %d\n\n",pos_recherche);
          trouve = TRUE; //permet de passer à l'element suivant
        }
        pos_recherche++; // pour faire avancer la boucle
        if (!trouve){
          printf("------------------------- en position
%d\n",pos_recherche);
        }
      }
      indice_code++; // recherche l'element suivant
   // affichage resultat intermediaire
      if (!trouve){
            printf(" la valeur n'est pas dans le tableau\n\n");
      }
   }
   if (trouve){
     printf(" on incremente le nb de trouve\n");
     nb++;
   }
}

//-------------------------------Affiche si trouve ou pas
if (nb){
          printf ("\n\n\nBILAN / le code est present %d fois
\n\n\n\n",nb);
          }
else{
          printf ("\n\n\nBILAN / le code n'est pas present\n\n\n\n");
          }

//------------------------------------------------------------
 system("PAUSE");

 return 0;
}
```

Destruction des occurrences suivantes

```
/*detruit les occurencesuivante
 */
/*on recherche chaque element et on donne sa position, on
recommence a chercher l'element suivant a partir de cette
position tt que depasse pas la capa*/
#include <stdio.h>
#include <stdlib.h>

#define DIMMAX 10 // pas utilise ici
#define TRUE 1
#define FALSE 0

int main(int argc, char *argv[])
{
//----------------------------définition d'un tableau test----------------
--------------------------------------------------------------------------
int dim_Tab = 8;
int Tab[] = {1,4,1,4,4,1,1,4};
int dim_Code = 2;
int Code[] = {1,4};
int val_cherchee;

int pos_recherche=0;
int indice_code=0;

int trouve= TRUE;
int trouve_detruire= TRUE;
int nb=0;
int derniere_pos=0;
int i;

//------------------------------affichage du tableau apres
 printf ("Voici le tableau avant le remplacement\n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
             printf ("%d\t",Tab[i]);
          }
          printf ("]\n");


// pour tt les positions
while (pos_recherche <= (dim_Tab - dim_Code)){
//    printf("\n ========================== %d
==============================\n",nb);
   derniere_pos=pos_recherche; // pour revenir en arrière d'une
recherche
   indice_code=0; // on initialise une nouvelle recherche
   while (trouve && indice_code<dim_Code){
      // TT que on trouve et qu'il existe des elements a chercher
//       printf("-- Recherche de %d ----------
\n",Code[indice_code]);
//       printf("------------------------- en position
%d\n",pos_recherche);
       trouve= FALSE; // je n'ai pas encore trouve l'ele
       val_cherchee=Code[indice_code];
    //   printf("la valeur recherche = %d\n",val_cherchee);
       while (!trouve && pos_recherche<dim_Tab){
         if (Tab[pos_recherche]==val_cherchee){
//             printf("== trouve en position %d\n\n",pos_recherche);
           trouve = TRUE; //permet de passer à l'element suivant
         }
         pos_recherche++; // pour faire avancer la boucle
         if (!trouve){
//             printf("------------------------- en position
%d\n",pos_recherche);
         }
       }
       indice_code++; // recherche l'element suivant
   // affichage resultat intermediaire
       if (!trouve){
 //             printf(" la valeur n'est pas dans le tableau\n\n");
       }
   }
   if (trouve){
//       printf(" on trouve le code une fois de plus \n");
     nb++;
   }
// destruction
   if (nb>1){// on revient en arriere et on detruit les 1 occurences
de code a partir de la veille pos
       indice_code=0;
       while (indice_code<dim_Code){
           trouve_detruire = FALSE;
         // TT qu'il existe des elements a detruire
         printf("-- Recherche pour detruire %d ----------
\n",Code[indice_code]);
//          printf("------------------------- en position
%d\n",derniere_pos);
          val_cherchee=Code[indice_code];
          while (!trouve_detruire && derniere_pos<dim_Tab){
            if (Tab[derniere_pos]==val_cherchee){
               printf("== destruction en position %d realisee
\n\n",derniere_pos);
               Tab[derniere_pos]=-1;
               trouve_detruire = TRUE; //permet de passer à
l'element suivant
            }
            derniere_pos++; // pour faire avancer la boucle
            if (!trouve){
               printf("------------------------- en position
%d\n",pos_recherche);
            }
          }
          indice_code++; // recherche l'element suivant
       }
   }
}

//--------------------------------Affiche si trouve ou pas
if (nb){
          printf ("\n\n\nBILAN / le code est present %d fois
\n\n\n\n",nb);
          }
else{
          printf ("\n\n\nBILAN / le code n'est pas present\n\n\n\n");
    }


//------------------------------affichage du tableau apres
 printf ("Voici le tableau avant le remplacement\n");
          printf ("tab = [");
          for (i=0;i<dim_Tab;i++){
             printf ("%d\t",Tab[i]);
          }
          printf ("]\n");
//------------------------------------------------------------
  system("PAUSE");
  return 0;
}
```