```c
#7sibik rabbi ya dorra
#include <stdio.h>
#include <math.h>

#define nb_reines  8
int pos[nb_reines];  //indice : numero de la ligne , contenu
 : numero de la colonne
//rendre 1 si les 2 reines (i1,j1) et (i2,j2) sont en confli
t, 0 sinon
int conflit(int i1,int j1,int i2,int j2)
{
    if (i1==i2 || j1==j2 || (abs(i1-i2)==abs(j1-j2)))
    {
        return 1;
    }
    return 0;
}

//rendre 1 si la reine(i,j) n'est pas en conflit avec toutes
 les reines déjà placées dans pos
int compatible(int i,int j)
{
    for (int k = 0; k < i; k++)
    {
        if (conflit(i,j,k,pos[k]))
        {
            return 0;
        }
    }
    return 1;
}

//affichage d'une solution : affichage du contenu du tab pos
[nb_reine]
void affichage()
{
    for (int i = 0; i < nb_reines; i++)
    {
        printf("%d \t",pos[i]);
```

```
    }
    printf("\n\n");
}

//fonction récursive de placement des reines
void reine(int i)
{
    if (i == nb_reines)
    {
        affichage();
    }
    else
    {
        for (int j = 0; j < nb_reines; j++)
        {
            if (compatible(i, j)) //verification de la compa
tibilité avec toutes les colones
            {
                pos[i] = j;
                reine(i + 1);
            }
        }
    }
}

void main()
{
    reine(0);
}

/*
      Solution 1: 1  3   0  2
      0  1  0  0
      0  0  0  1
      1  0  0  0
      0  0  1  0
      Solution 2: 2  0   3   1
      0  0  1  0
      1  0  0  0
```

```
      0  0  0  1
      0  1  0  0
*/
```