En una matriz cuadrada m de orden N se dan las definiciones siguientes:
1. Son adyacentes a un elemento aij de la matriz m aquellos akl tales que:

$i-1 \le k \le i+1, con\ i-1 \ge 0, i+1 < N$
$j-1 \le l \le j+1, con\ j-1 \ge 0, j+1 < N$, con $i \ne k\ ||\ j \ne l$

2. Se dice que un elemento aij de la matriz m es un acumulador si y solo si aij es la suma de todos los valores de sus adyacentes.

Ejemplo: Dada la matriz m

6	2	−1	5
2	2	**6**	2
1	1	−6	1
0	2	3	**−2**

Acumuladores de m = {6 posición (0, 0), 6 posición (1, 2) y -2 posición (3, 3)}

SE PIDE:
a) Escribir una función en C con el prototipo siguiente:
`int esAcumulador (int i, int j, int m[][N]) ;`

que devuelva 1 si el elemento m[i][j] es un acumulador en la matriz m, y 0, en otro caso.

b) Escribir un programa en C que escriba todos los elementos de una matriz cuadrada de orden N que son acumuladores indicando la posición que ocupan en la matriz como en el ejemplo utilizando la función anterior. Si la matriz m no tuviese acumuladores se escribirá el mismo mensaje sin ningún valor entre las llaves.
NOTA: NO se pueden utilizar estructuras auxiliares para la resolución del problema.

```
#include <stdio.h>
#define N 4

int esAcumulador(int i, int j, int m[N][N]);
void leerMatriz (int m[N][N]);
void main()
{
    int i,j, cont=0, pos, fil, col;
    int mat[N][N]={{6,2,-1,5},{2,2,6,2},{1,1,-6,1},{0,2,3,-2}};

    printf("Acumuladores de m = {");
    for(i=0; i<N; i++)
    {
        for(j=0; j<N; j++)
        {
            if (esAcumulador(i, j, mat))
            {
                cont++;
                if (cont==1)
                    printf("%d posicion (%d, %d)", mat[i][j], i, j);
                else
```

```
                {
                    if (cont>2)
                        printf(", %d posicion (%d, %d)", pos, fil,
col);
                    pos=mat[i][j];
                    fil=i;
                    col=j;
                }
            }
        }
    }
    if (cont>1)
        printf(" y %d posicion (%d, %d)", pos, fil, col);
    printf("}");
}

int esAcumulador(int i, int j, int m[N][N])
{
    int f, c, fi, ff, ci, cf, sum=0;
    if (i==0) fi=i;
    else fi=i-1;
    if (i==N-1) ff=i;
    else ff=i+1;
    if (j==0) ci=j;
    else ci=j-1;
    if (j==N-1) cf=j;
    else cf=j+1;

    for(f=fi; f<=ff; f++)
        for(c=ci; c<=cf; c++)
            sum=sum + m[f][c];
    return sum-m[i][j]== m[i][j];
}
```

// Otra solución sin comas ni el “y” separando el último valor:

```
#include <stdio.h>
#define N 4

int esAcumulador(int i, int j, int m[ ][N]);

void main()
{
    int i,j, cont=0, pos, fil, col;
    int mat[N][N]= {{6,2,-1,5},{2,2,6,2},{1,1,-6,1},{0,2,3,-2}};

    printf("Acumuladores de m = {");
    for(i=0; i<N; i++)
    {
        for(j=0; j<N; j++)
        {
```

```
            if (esAcumulador(i, j, mat))
                printf("%d posicion (%d, %d) ", mat[i][j], i, j);
        }
    }
    printf("}");
}

int esAcumulador(int i, int j, int m[N][N])
{
    int f, c, fi, ff, ci, cf, sum=0;
    if (i==0) fi=i;
    else fi=i-1;
    if (i==N-1) ff=i;
    else ff=i+1;
    if (j==0) ci=j;
    else ci=j-1;
    if (j==N-1) cf=j;
    else cf=j+1;

    for(f=fi; f<=ff; f++)
        for(c=ci; c<=cf; c++)
            sum=sum + m[f][c];
    return sum-m[i][j]== m[i][j];
}
```