



Universidad Nacional Autónoma de México  
Facultad de Ingeniería

---

# **Pseudocódigo y diagrama de flujo del Proyecto Final**

## **(semestre 2021-1)**

Sánchez Sánchez Daniel  
Grupo: 25  
22 de Enero del 2020

## PSEUDOCODIGO

Algoritmo (Principal)

Variables:

    Enteros: op, op\_Gato, op\_Conecta  
    Char: g [3][3]

INICIO

    ESCRIBIR ("Bienvenido!, porfavor degite el juego de su eleccion\n")  
    ESCRIBIR ("1) Gato\n")  
    ESCRIBIR ("2) Conecta 4\n")  
    ESCRIBIR ("\nEl juego que deseo juegas es: ");  
    LEER (op);  
    ESCRIBIR ("\n\n\n");

SELECCIONAR (op) EN

CASO 1 ->

        ESCRIBIR ("Bienvenido al juego de Gato!\n\n");  
        ESCRIBIR ("Por favor eliga un modo de juego\n");  
        ESCRIBIR ("1) Jugador vs Maquina\n");  
        ESCRIBIR ("2) Jugados vs Jugador\n");  
        ESCRIBIR ("\nMi elección es: ");  
        LEER (op\_Gato);  
        ESCRIBIR ("\n\n\n");

    SELECCIONAR (op\_Gato) EN

        CASO 1 ->

            loop\_JvM (g)

        CASO 2 ->

            loop\_JvJ (g)

    FIN SELECCIONAR

CASO 2 ->

        ESCRIBIR ("Bienvenido al juego de Conecta 4!\n\n");  
        ESCRIBIR ("Por favor elija un modo de juego\n");  
        ESCRIBIR ("1) Jugador vs Jugador\n");  
        ESCRIBIR ("2) Jugados vs Maquina\n");  
        ESCRIBIR ("\nMi elección es: ");  
        LEER (op\_Conecta);  
        ESCRIBIR ("\n\n\n");

    SI (op\_Conecta <= 0 || op\_Conecta > 3)  
        return 0;

    FIN SI

    jugar(op\_Conecta)  
    return 0;

\*\* Funciones de Gato

INICIO

\*\*Función que hace la lógica de un Jugador vs Maquina  
FUNC loop\_JvM (char g [3][3])

Variables:

    Enteros: i, j  
    i = 0

    FUNC Introducción (g)  
    HACER

        Tablero (g)  
        SI (i == 0)  
            FUNC Turno\_Jugador1(g)  
        FIN SI  
        SINO  
            FUNC Turno\_IA(g)  
            ESCRIBIR ("\n\n")  
        FIN SINO

    MIENTRAS (i <= 9 && j == 2)

        Tablero(g);

        SI (j == 0)  
            ESCRIBIR ("\nGANASTE!!!\n");  
            ESCRIBIR ("By: Dan Sanchez\n\n");  
        FIN SI  
        SINO (j == 1)  
            ESCRIBIR ("\nPerdiste:(\n");  
            ESCRIBIR ("By: Dan Sanchez\n\n");  
        FIN SINO  
        SINO  
            ESCRIBIR ("\nVaya empate!, Vuelvelo a intentar\n");  
            ESCRIBIR ("By: Dan Sanchez\n\n");  
        FIN SINO

    FIN FUNC

FIN

INICIO

\*\*Función que hace la lógica de un Jugador vs Jugador

FUNC loop\_JvJ (char g [3][3])

Variables:

Enteros: i, j

i = 0;

FUNC Introduccion(g)

HACER

FUNC Tablero(g)

SI (i%2 == 0)

FUNC Turno\_Jugador1(g)

FIN SI

SINO

FUNC Turno\_Jugador2(g)

ESCRIBIR ("\n\n")

FIN SINO

j = Condicion (g)

i++

MIENTRAS(i <= 9 && j == 2);

Tablero(g);

SI (j == 0)

ESCRIBIR ("\nGANASTE JUGADOR 1!!!\n")

ESCRIBIR ("By: Dan Sanchez\n\n")

FIN SI

SINO (j == 1)

ESCRIBIR ("\nGANASTE JUGADOR 2!!!\n")

ESCRIBIR ("By: Dan Sanchez\n\n")

FIN SINO

SINO

ESCRIBIR ("\nVaya empate!, Vuelvanlo a intentar\n")

ESCRIBIR ("By: Dan Sanchez\n\n")

FIN SINO

FIN FUNC

FIN

INICIO

\*\*Funcion para escribir los numeros en el tablero

FUNC Introduccion (char g [3][3])

Variables:

Enteros: i, j

Char: aux

aux = '1'

PARA i = 0 HASTA i < 3 HACER i = i + 1

PARA j = 0 HASTA j < 3 HACER j = j + 1

g[i][j] = aux++;

FIN PARA

FIN PARA

FIN FUNC

FIN

INICIO

\*\*Funcion para hacer las lineas de la matriz del tablero

FUNC Tablero (char g [3][3])

Variables:

Enteros: i, j

PARA i = 0 HASTA i < 3 HACER i++

PARA j = 0 HASTA j < 3 HACER j++

SI (j < 2)

ESCRIBIR (" %c |", g[i][j])

FIN SI

SINO

ESCRIBIR (" %c ", g[i][j])

FIN SINO

SI (i < 2)

ESCRIBIR ("\n-----\n")

FIN SI

ESCRIBIR ("\n\n")

FIN FUNC

FIN

## INICIO

\*\* Funcion donde le pide al Jugador 1 el espacio de la matriz donde desea tirar y lo escribe en el tablero

FUNC Turno\_Jugador1(char g [3][3]){  
Variables:

    Enteros: i = 0, j = 0, k  
    Char: aux

## HACER

### HACER

    ESCRIBIR ("Turno del jugador 1\n");

    ESCRIBIR ("Escoge la casilla: ");

    LEER ("%c", &aux);

    ESCRIBIR ("\n\n");

    MIENTRAS (aux < '1' || aux > '9');

        k = 0;

        SELECCIONAR (aux)

CASO 1->

    i = 0

    j = 0

    SI (g[i][j] == 'X' || g[i][j] == 'O')  
        k = 1;

        ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n");

    FIN SI

CASO 2 ->

    i = 0

    j = 1

    SI (g[i][j] == 'X' || g[i][j] == 'O')  
        k = 1;

        ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n");

    FIN SI

CASO 3 ->

    i = 0;

    j = 2;

    SI (g[i][j] == 'X' || g[i][j] == 'O')  
        k = 1;

        ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n");

    FIN SI

CASO 4->

    i = 1

    j = 0

    SI (g[i][j] == 'X' || g[i][j] == 'O')  
        k = 1;

        ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n");

    FIN SI

CASO 5 ->

i = 1  
j = 1

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;  
ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")  
FIN SI

CASO 6 ->

i = 1  
j = 2

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;  
ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")  
FIN SI

CASO 7 ->

i = 2;  
j = 0;

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;  
ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")  
FIN SI

CASO 8 ->

i = 2  
j = 1

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;  
ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n");  
FIN SI

CASO 9 ->

i = 2  
j = 2

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;  
ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")  
FIN SI

FIN SELECCIONAR

MIENTRAS (k == 1)

g[i][j] = 'X'

FIN FUNC

FIN

## INICIO

\*\* Funcion donde le pide al Jugador 1 el espacio de la matriz donde desea tirar y lo escribe en el tablero

FUNC Turno\_Jugador2 (char g [3][3])  
Variables:

Enteros: i = 0, j = 0, k

Char: aux

## HACER

### HACER

ESCRIBIR ("Turno del jugador 2\n");

ESCRIBIR ("Escoge la casilla: ");

LEER ("%c", &aux);

ESCRIBIR ("\n\n");

MIENTRAS (aux < '1' || aux > '9');

k = 0;

SELECCIONAR (aux)

CASO 1->

i = 0

j = 0

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;

ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n");

FIN SI

CASO 2 ->

i = 0

j = 1

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;

ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")

FIN SI

CASO 3 ->

i = 0;

j = 2;

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;

ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")

FIN SI

CASO 4->

i = 1

j = 0

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;

ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")

FIN SI

CASO 5 ->

i = 1  
j = 1

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;  
ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")  
FIN SI

CASO 6 ->

i = 1  
j = 2

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;  
ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")  
FIN SI

CASO 7 ->

i = 2;  
j = 0;

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;  
ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")  
FIN SI

CASO 8 ->

i = 2  
j = 1

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;  
ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n");  
FIN SI

CASO 9 ->

i = 2  
j = 2

SI (g[i][j] == 'X' || g[i][j] == 'O')  
k = 1;  
ESCRIBIR ("CASILLA OCUPADA! Escoga otra\n\n")  
FIN SI

FIN SELECCIONAR

MIENTRAS (k == 1)

g[i][j] = 'X'

FIN FUNC

FIN

## INICIO

\*\* Función para crear un numero en aleatorio que posteriormente se leera como una elección de un numero donde la maquina deseó tirar

FUNC Turno\_IA (char g[3][3])

Variables:

    Enteros: i, j, k

srand (time(NULL));

HACER:

    i = rand() %3

    j = rand() %3

    k = 0

    SI (g[i][j] == 'X' || g[i][j] == 'O')

        k = 1

    FIN SI

    MIENTRAS (k == 1);

        g [i][j] = 'O'

    FIN FUNC

FIN

## INICIO

\*\*En este tipo de funcion creamos la condicion de gane  
FUNC Condicion (char g[3][3])

```
SI (g[0][0] == 'X' || g[0][0] == 'O')
    SI (g[0][0] == g[0][1] && g[0][0] == g[0][2])
        SI (g[0][0] == 'X')
            return 0 **Gana el primer turno
        FIN SI
    SINO
        return 1 **Gana el segundo turno
    FIN SINO
FIN SI

SI (g[0][0] == g[1][0] && g[0][0] == g[2][0])
    SI (g[0][0] == 'X')
        return 0 **Gana el primer turno
    FIN SI
    SINO
        return 1 **Gana el segundo turno
    FIN SINO
FIN SI

SI (g[1][1] == 'X' || g[1][1] == 'O')
    SI (g[1][1] == g[0][0] && g[1][1] == g[2][2])
        SI (g[1][1] == 'X')
            return 0 **Gana el primer turno
        FIN SI
        SINO
            return 1 **Gana el segundo turno
    FIN SINO
FIN SI

SI (g[1][1] == g[1][0] && g[1][1] == g[1][2])
    SI (g[1][1] == 'X')
        return 0 **Gana el primer turno
    FIN SI
    SINO
        return 1 **Gana el segundo turno
    FIN SINO
FIN SI

SI (g[1][1] == g[2][0] && g[1][1] == g[0][2])
    SI (g[1][1] == 'X')
        return 0 **Gana el primer turno
    FIN SI
    SINO
        return 1 **Gana el segundo turno
    FIN SINO
FIN SI
```

```

SI (g[1][1] == g[0][1] && g[1][1] == g[2][1]){
    SI (g[1][1] == 'X'){
        return 0 **Gana el primer turno
    FIN SI
    SINO
        return 1 **Gana el segundo turno
    FIN SINO
FIN SI

SI (g[2][2] == 'X' || g[2][2] == 'O')
    SI (g[2][2] == g[2][0] && g[2][2] == g[2][1])
        SI (g[2][2] == 'X')
            return 0 **Gana el primer turno
        FIN SI
        SINO
            return 1 **Gana el segundo turno
    FIN SINO

SI (g[2][2] == g[0][2] && g[2][2] == g[1][2])
    SI (g[2][2] == 'X')
        return 0 **Gana el primer turno
    FIN SI
    SINO
        return 1 **Gana el segundo turno
    FIN SINO
FIN SI
FIN SI

return 2

FIN FUNC
FIN

```

\*\* Funciones de Conecta 4

INICIO

FUNC obtenerFilaDesocupada(columna, tablero[FILAS][COLUMNAS])  
Variables:

    Enteros: i, FILAS, COLUMNAS, columna  
    Char: tablero

    FILAS = 6  
    COLUMNAS = 7

    PARA i = FILAS - 1 HASTA i >= 0 HACER i = i - 1  
        SI (tablero[i][columna] == ESPACIO\_VACIO)  
            return i;

    FIN SI

FIN PARA

    return FILA\_NO\_ENCONTRADA

FIN FUNC

FIN

INCIO

FUNC colocarPieza(jugador columna tablero[FILAS][COLUMNAS])  
Variables:

    Enteros: columna, FILAS, COLUMNAS  
    Char: jugador, tablero

    FILAS = 6  
    COLUMNAS = 7

    SI (columna < 0 || columna >= COLUMNAS)  
        return ERROR\_FILA\_INVALIDA

    FIN SI

    Entero: fila = obtenerFilaDesocupada(columna, tablero)

    SI (fila == FILA\_NO\_ENCONTRADA)  
        return ERROR\_COLUMNILLAENA

    FIN SI

    tablero[fila][columna] = jugador  
    return ERROR\_NINGUNO

FIN FUNC

FIN

INICIO

FUNC limpiarTablero(char tablero[FILAS][COLUMNAS]) {

Variables:

    Enteros: FILAS, COLUMNAS

    Char: tablero

    FILAS = 6

    COLUMNAS = 7

    Entero: i

    PARA i = 0 HASTA i < FILAS HACER i++

        Entero j

        PARA j = 0 HASTA j < COLUMNAS HACER ++j

            tablero[i][j] = ESPACIO\_VACIO;

        FIN PARA

    FIN PARA

    FIN FUNC

FIN

INCIO

FUNC dibujarEncabezado(columnas)

Variable:

    Entero: i

    ESCRIBIR ("\n");

    PARA i = 0 HASTA i < columnas HACER ++i

        ESCRIBIR ("|%d", i + 1);

    SI (i + 1 >= columnas)

        ESCRIBIR ("|");

    FIN SI

    FIN PARA

    FIN FUNC

FIN

INICIO

FUNC dibujarTablero(tablero[FILAS][COLUMNAS])

Variables:

Enteros: FILAS, COLUMNAS, i, j

Char: tablero

FILAS = 6

COLUMNAS = 7

dibujarEncabezado(COLUMNAS);

ESCRIBIR ("\n");

PARA i = 0 HASTA i < FILAS HACER i++

PARA j = 0 HASTA j < COLUMNAS HACER j++

ESCRIBIR ("|%c", tablero[i][j]);

SI (j + 1 >= COLUMNAS)

ESCRIBIR ("|");

FIN SI

FIN PARA

ESCRIBIR ("\n");

FIN PARA

return 0;

FIN FUNC

FIN

INICIO

FUNC esEmpate(tablero[FILAS][COLUMNAS])

Variables:

Enteros: FILAS, COLUMNAS, i

Char: tablero

FILAS = 6

COLUMNAS = 7

PARA i = 0 HASTA i < COLUMNAS HACER i++

Entero: resultado = obtenerFilaDesocupada(i, tablero);

SI (resultado != FILA\_NO\_ENCONTRADA) {

return 0;

FIN SI

FIN PARA

return 1

FIN FUNC

FIN

\*\* Funciones hacer el conteo de conexiones

INICIO

    \*\* Hace el conteo por columnas  
    FUNC contarArriba(x, y, jugador, tablero[FILAS][COLUMNAS])  
    Variables:

        Enteros: x, y, FILAS, COLUMNAS, yInicio, contador  
        Char: jugador, tablero

        FILAS = 6  
        COLUMNAS = 7

        yInicio = (y - CONECTA >= 0) ? y - CONECTA + 1 : 0;  
        contador = 0;

        HASTA yInicio <= y HACER yInicio++

            SI (tablero[yInicio][x] == jugador)  
                contador++;  
            FIN SI  
            SINO  
                contador = 0;  
            FIN SINO

        FIN HASTA  
        return contador;

    FIN FUNC

FIN

INICIO

    \*\* Hace el conteo por renglones  
    FUNC contarDerecha (x, y, jugador, tablero[FILAS][COLUMNAS])  
    Variables:

        Enteros: x, y, FILAS, COLUMNAS, xFin, contador  
        Char: jugador, tablero

        FILAS = 6  
        COLUMNAS = 7

        xFin = (x + CONECTA < COLUMNAS) ? x + CONECTA - 1 : COLUMNAS - 1;  
        contador = 0;

        HASTA x <= xFin HACER x++

            SI (tablero[y][x] == jugador)  
                contador++;  
            FIN SI  
            SINO  
                contador = 0;  
            FIN SINO

        FIN HASTA  
        return contador;

    FIN FUNC

FIN

## INICIO

\*\* Hace el conteo en diagonal hacia arriba

FUNC contarArribaDerecha(x, y, jugador, tablero[FILAS][COLUMNAS])

Variables:

Enteros: x, y, FILAS, COLUMNAS, xFin, yInicio, contador

Char: jugador, tablero

xFin = (x + CONECTA < COLUMNAS) ? x + CONECTA - 1 : COLUMNAS - 1;

yInicio = (y - CONECTA >= 0) ? y - CONECTA + 1 : 0;

contador = 0;

MIENTRAS (x <= xFin && yInicio <= y)

SI(tablero[y][x] == jugador)

    contador++;

FIN SI

SINO

    contador = 0;

FIN SINO

    x++

    y--

FIN MIENTRAS

return contador;

FIN FUNC

FIN

## INICIO

\*\* Hace el conteo en diagonal hacia abajo

FUNC contarAbajoDerecha (x, y, jugador, tablero[FILAS][COLUMNAS])

Variables:

Enteros: x, y, FILAS, COLUMNAS, xFin, yFin, contador

Char: jugador, tablero

xFin = (x + CONECTA < COLUMNAS) ? x + CONECTA - 1 : COLUMNAS - 1;

yFin = (y + CONECTA < FILAS) ? y + CONECTA - 1 : FILAS - 1;

contador = 0;

MIENTRAS (x <= xFin && y <= yFin) {

    SI (tablero[y][x] == jugador) {

        contador++;

    FIN SI

    SINO

        contador = 0;

    FIN SINO

        x++;

        y++;

    FIN MIENTRAS

    return contador

FIN FUNC

FIN

## INICIO

\*\* Revisa si alguien ya realizo 4 conexiones, anida las otras funciones

FUNC ganador(char jugador, char tablero[FILAS][COLUMNAS]) {

Variables:

    Enteros: FILAS, COLUMNAS, y, x

    FILAS = 6

    COLUMNAS = 7

PARA y = 0 HASTA y < FILAS HACER y++

    PARAx = 0 HASTA x < COLUMNAS HACER x++

        FUNC conteoArriba = contarArriba(x, y, jugador, tablero);

        SI (conteoArriba >= CONECTA)

            return FUNC CONECTA\_ARRIBA;

        FIN SI

        SI(contarDerecha(x, y, jugador, tablero) >= CONECTA)

            return FUNC CONECTA\_DERECHA

        FIN SI

        SI (contarArribaDerecha(x, y, jugador, tablero) >= CONECTA)

            return FUNC CONECTA\_ARRIBA\_DERECHA

        FIN SI

        SI (contarAbajoDerecha(x, y, jugador, tablero) >= CONECTA)

            return FUNC CONECTA\_ABAJO\_DERECHA

        FIN SI

    FIN PARA

    FIN PARA

        return FUNC NO\_CONECTA;

    FIN FUNC

FIN

\*\* Funciones de IA

## INICIO

FUNC clonarMatriz(tableroOriginal[FILAS][COLUMNAS], destino[FILAS][COLUMNAS])

Variables:

    Enteros: FILAS, COLUMNAS

    Char: tableroOriginal, destino

    memcpy(destino, tableroOriginal, TAMANIO\_MATRIZ);

    FIN FUNC

FIN

INICIO

FUNC obtenerColumnaGanadora(jugador, tableroOriginal[FILAS][COLUMNAS])  
Variables:

Char: tablero, jugador  
Enteros, FILAS, COLUMNAS, i

FILAS = 6  
COLUMNAS = 7

PARA i = 0 HASTA i < COLUMNAS HACER i++  
    clonarMatriz(tableroOriginal, tablero)  
    FUNC resultado = colocarPieza(jugador, i, tablero)

    SI (resultado == ERROR\_NINGUNO)  
        FUNC gana = ganador(jugador, tablero)

        SI (gana != NO\_CONECTA)  
            return i

    FIN SI

FIN SI

FIN PARA

    return COLUMNA\_GANADORA\_NO\_ENCONTRADA;

FIN FUNC

FIN

INICIO

FUNC obtenerPrimeraFilaLlena(columna, tablero[FILAS][COLUMNAS])  
Variables:

Char: tablero  
Enteros: i, columna, FILAS, COLUMNAS

PARA i = 0 HASTA i < FILAS HACER i++

    SI (tablero[i][columna] != ESPACIO\_VACIO)  
        return i;

    FIN SI

FIN PARA

    return FILA\_NO\_ENCONTRADA;

FIN FUNC

FIN

## INICIO

\*\* Los dos últimos apuntadores son porque no podemos regresar dos variables  
FUNC obtenerColumnaEnLaQueSeObtieneMayorPuntaje(jugador,  
tableroOriginal[FILAS][COLUMNAS], \*conteo, \*indice)

Variables:

Enteros: FILAS, COLUMNAS, conteoMayor, i  
Char: jugador, tableroOriginal, tablero

FILAS = 6  
COLUMNAS = 7  
conteoMayor = 0,  
indiceColumnaConConteoMayor = -1;

PARA (i = 0; i < COLUMNAS; i++)  
    clonarMatriz(tableroOriginal, tablero);  
    FUNC estado = colocarPieza(jugador, i, tablero);

    SI (estado == ERROR\_NINGUNO) {  
        FUNC filaDePiezaRecienColocada = obtenerPrimeraFilaLlena  
        SI (filaDePiezaRecienColocada != FILA\_NO\_ENCONTRADA) {  
            FUNC c = contarArriba  
            SI (c > conteoMayor)  
                conteoMayor = c  
                indiceColumnaConConteoMayor = i  
            FIN SI

            c = contarArribaDerecha  
            SI (c > conteoMayor)  
                conteoMayor = c;  
                indiceColumnaConConteoMayor = i;  
            FIN SI

            c = contarDerecha

            SI (c > conteoMayor)  
                conteoMayor = c;  
                indiceColumnaConConteoMayor = i;  
            FIN SI

            c = contarAbajoDerecha

            SI(c > conteoMayor) {  
                conteoMayor = c;  
                indiceColumnaConConteoMayor = i;  
            FIN SI

    FIN SI  
FIN PARA

\*conteo = conteoMayor;  
\*indice = indiceColumnaConConteoMayor;

FIN FUNC  
FIN

INICIO

FUNC obtenerColumnaAleatoria(char jugador, char tableroOriginal[FILAS][COLUMNAS])  
Variables:

    Enteros: FILAS, COLUMNAS  
    Char: jugador, tablero

    FILAS = 6  
    COLUMNAS = 7

    MIENTRAS (1)

        char tablero[FILAS][COLUMNAS];  
        clonarMatriz(tableroOriginal, tablero);  
        Entero: columna = aleatorio\_en\_rango(0, COLUMNAS - 1);  
        Entero resultado = colocarPieza(jugador, columna, tablero);

        SI (resultado == ERROR\_NINGUNO)  
            return columna

    FIN SI

    FIN MIENTRAS

FIN FUNC

FIN

INICIO

FUNC obtenerColumnaCentral(jugador, tableroOriginal[FILAS][COLUMNAS])

Variables:

    char tablero[FILAS][COLUMNAS];  
    Enteros: FILAS, COLUMNAS  
    clonarMatriz(tableroOriginal, tablero)  
    Entero: mitad = (COLUMNAS - 1) / 2  
    Entero: resultado = colocarPieza(jugador, mitad, tablero)

    FILAS = 6  
    COLUMNAS = 7

    SI (resultado == ERROR\_NINGUNO) {  
        return mitad;

    FIN SI

    return COLUMNA\_GANADORA\_NO\_ENCONTRADA;

FIN FUNC

FIN

INICIO

```
FUNC elegirColumnaCpu(char jugador, char tablero[FILAS][COLUMNAS]) {
    ** Voy a comprobar si puedo ganar...
    FUNC posibleColumnaGanadora = obtenerColumnaGanadora(jugador, tablero);

    SI (posibleColumnaGanadora != COLUMNA_GANADORA_NO_ENCONTRADA) {
        ESCRIBIR ("*Elijo ganar*\n");
        return posibleColumnaGanadora;
    FIN SI

    ** Si no, voy a comprobar si mi oponente gana con el siguiente movimiento, para
       evitarlo
    char oponente = obtenerOponente(jugador);
    FUNC posibleColumnaGanadoraDeOponente = obtenerColumnaGanadora

    SI (posibleColumnaGanadoraDeOponente != COLUMNA_GANADORA_NO_ENCONTRADA)
        ESCRIBIR ("*Elijo evitar que mi oponente gane*\n");
        return posibleColumnaGanadoraDeOponente;
    FIN SI

    ** En caso de que nadie pueda ganar en el siguiente movimiento, buscaré en dónde se
       obtiene el mayor puntaje al colocar la pieza
    FUNC conteoCpu, columnaCpu
    obtenerColumnaEnLaQueSeObtieneMayorPuntaje
    FUNC conteoOponente, columnaOponente
    obtenerColumnaEnLaQueSeObtieneMayorPuntaje

    SI (conteoOponente > conteoCpu)
        ESCRIBIR ("*Elijo quitarle el puntaje a mi oponente*\n")
        return columnaOponente
    FIN SI
    SINO (conteoCpu > 1)
        ESCRIBIR ("*Elijo colocarla en donde obtengo un mayor puntaje*\n");
        return columnaCpu;
    FIN SINO

    ** Si no, regresar la central por si está desocupada
    FUNC columnaCentral = obtenerColumnaCentral(jugador, tablero);

    SI (columnaCentral != COLUMNA_GANADORA_NO_ENCONTRADA) {
        ESCRIBIR ("*Elijo ponerla en el centro*\n");
        return columnaCentral;
    FIN SI

    ** Finalmente, devolver la primera disponible de manera aleatoria
    FUNC columna = obtenerColumnaAleatoria(jugador, tablero);

    SI (columna != FILA_NO_ENCONTRADA) {
        ESCRIBIR ("*Elijo la primera vacía aleatoria*\n");
        return columna;
    FIN SI
```

```

        ESCRIBIR ("Esto no debería suceder\n");
        return 0;

    FIN FUNC
FIN

** Funciones necesarias para el funcionamiento completo del juego
INICIO
    FUNC obtenerOponente(char jugador) {

        SI (jugador == JUGADOR_1)
            return JUGADOR_2
        FIN SI
        SINO
            return JUGADOR_1;
        FIN SINO

    FUNC aleatorio_en_rango(int minimo, int maximo)
        return minimo + rand() / (RAND_MAX / (maximo - minimo + 1)) + 1;
    FIN FUNC

    FUNC elegirJugadorAlAzar()
        FUNC numero = aleatorio_en_rango(0, 1);

        SI (numero) {
            return JUGADOR_1
        FIN SI
        SINO
            return JUGADOR_2;
        FIN SINO

    FIN FUNC
FIN FUNC

    FIN FUNC
FIN

INICIO
    FUNC solicitarColumnaAJugador() {
        Variables
            Enteros: columna = 0

            ESCRIBIR ("Escribe la columna en donde colocar la pieza: ")
            LEER ("%d", &columna)

            ** Necesitamos índices de arreglos
            columna--

            return columna

    FIN FUNC
FIN

```

\*\* Funcion donde anida todas las anteriores funciones para el funcionamiento correcto del juego

INICIO

FUNC jugar(int op\_Conecta) {

Variables:

char: tablero[FILAS][COLUMNAS];

FUNC limpiarTablero(tablero);

char: jugadorActual = FUNC elegirJugadorAlAzar();

ESCRIBIR ("Comienza el jugador %c\n", jugadorActual);

MIENTRAS (1)

Entero: columna = 0;

ESCRIBIR ("\n\nTurno del jugador %c", jugadorActual);

FUNC dibujarTablero(tablero);

\*\* Esta seccion es para saber que modo de juego será

\*\* Primero será modo Jugador vs Maquina

SI (op\_Conecta == MODO\_HUMANO\_CONTRA\_CPU)

SI (jugadorActual == JUGADOR\_CPU\_2)

ESCRIBIR ("CPU 2 analizando... ")

columna = FUNC elegirColumnaCpu(jugadorActual, tablero)

FIN SI

SINO

columna = FUNC solicitarColumnaAJugador()

FINSINO

\*\* Esta parte del codigo sera una sorpresa para el jugador

\*\* Será un modo CPU vs CPU

FIN SI

SINO (op\_Conecta == MODO\_CPU\_CONTRA\_CPU) {

ESCRIBIR ("CPU analizando... ", jugadorActual == JUGADOR\_CPU\_1 ?

1 : 2)

columna = elegirColumnaCpu(jugadorActual, tablero)

\*\* Esta perte es para la opción Jugador vs Jugador

FIN SINO

SINO (op\_Conecta == MODO\_HUMANO\_CONTRA\_HUMANO)

columna = FUNC solicitarColumnaAJugador()

FIN SI

// Esta parte es por si hay algun problema a la ora de poner la pieza

Entero: estado = colocarPieza(jugadorActual, columna, tablero)

SI (estado == ERROR\_COLUMNAS\_LLENA)

ESCRIBIR ("Error: Columna llena")

FIN SI

SINO (estado == ERROR\_FILA\_INVALIDA)

ESCRIBIR ("Error: Fila no correcta")

FIN SINO

SINO (estado == ERROR\_NINGUNO)

Entero: g = FUNC ganador(jugadorActual, tablero)

SI (g != NO\_CONECTA) {

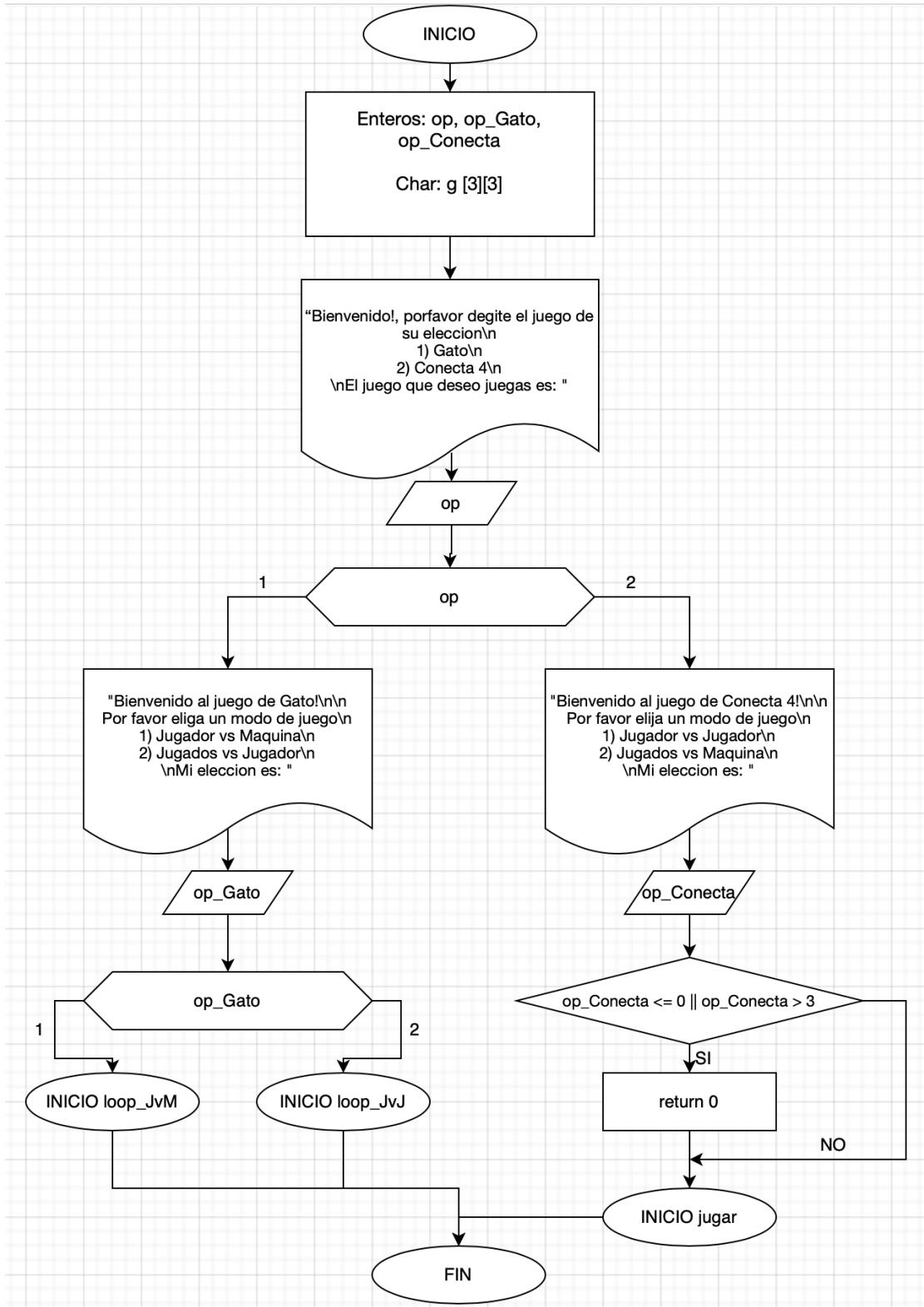
FUNC dibujarTablero(tablero)

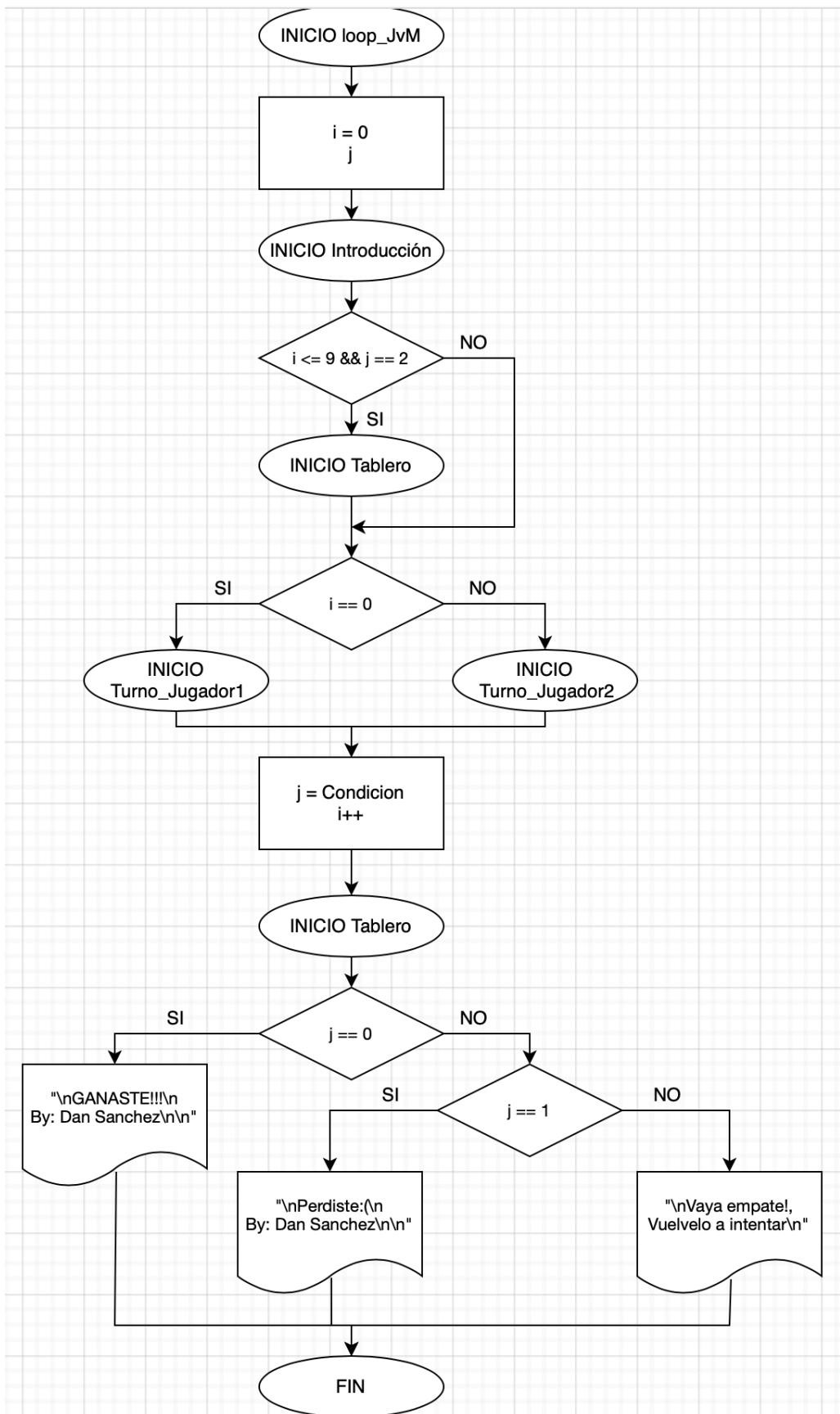
```
** Esta parte es para poner quien ah ganado
ESCRIBIR ("Gana el jugador %c. \n", jugadorActual)
ESCRIBIR ("By: Dan Sanchez\n\n")
break
FIN SINO
SINO (esEmpate(tablero))
    dibujarTablero(tablero)
    ESCRIBIR ("EMPATEEE!")
    ESCRIBIR ("By: Dan Sanchez\n\n")
    break
FIN SINO
FIN MIENTRAS

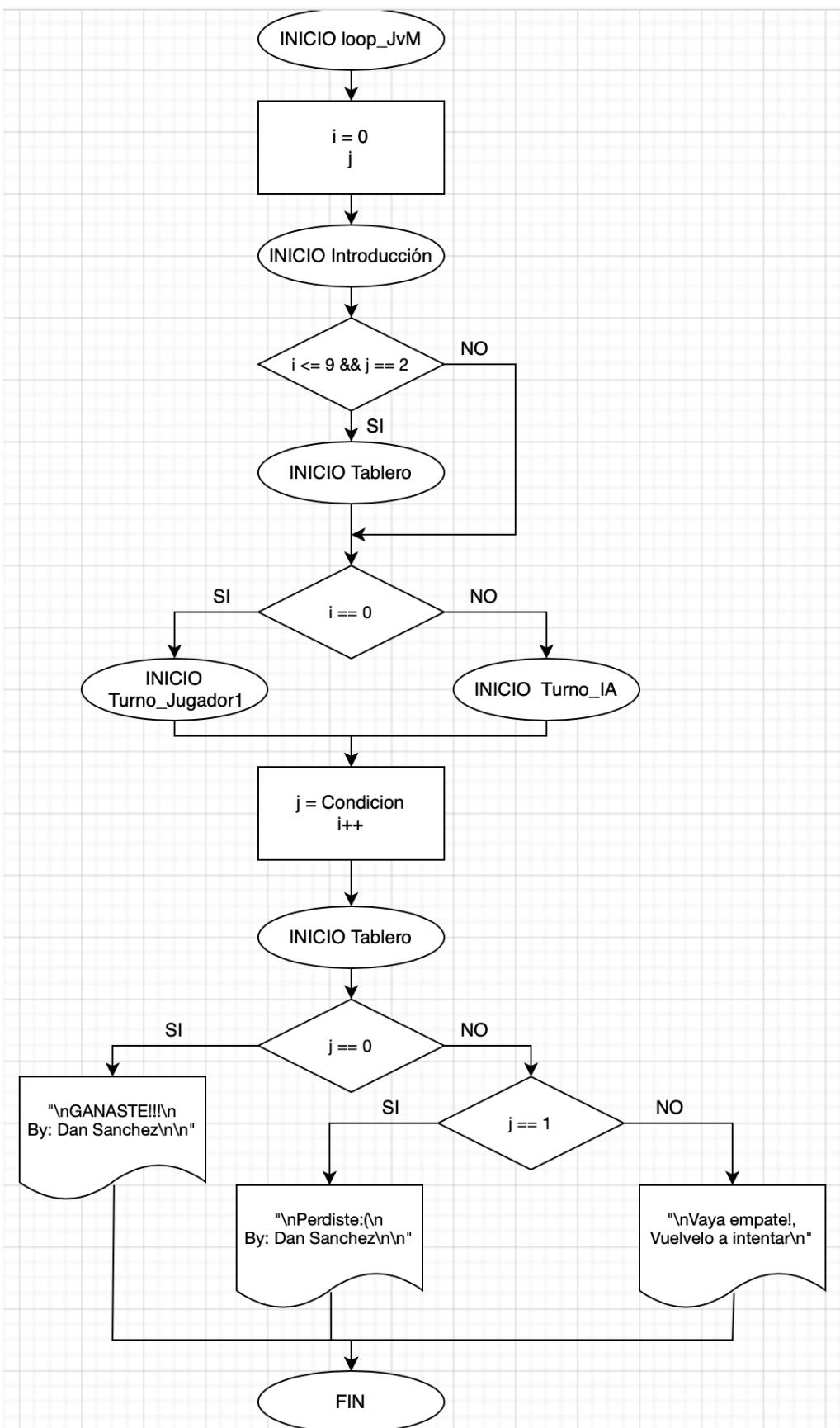
jugadorActual = FUNC obtenerOponente(jugadorActual)

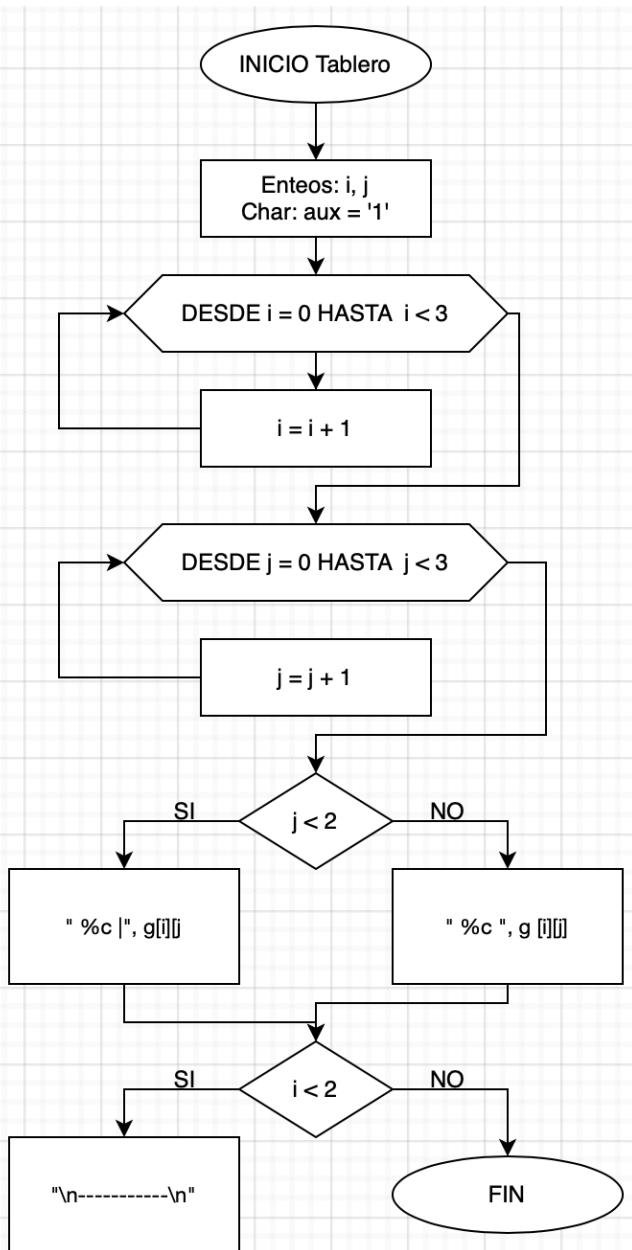
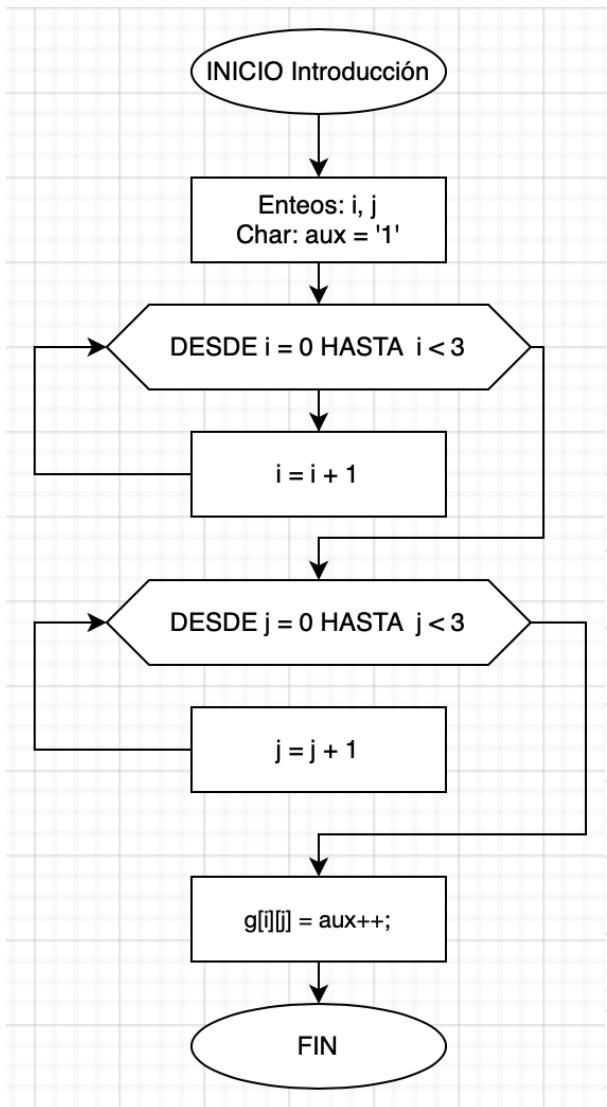
FIN FUNC
FIN
```

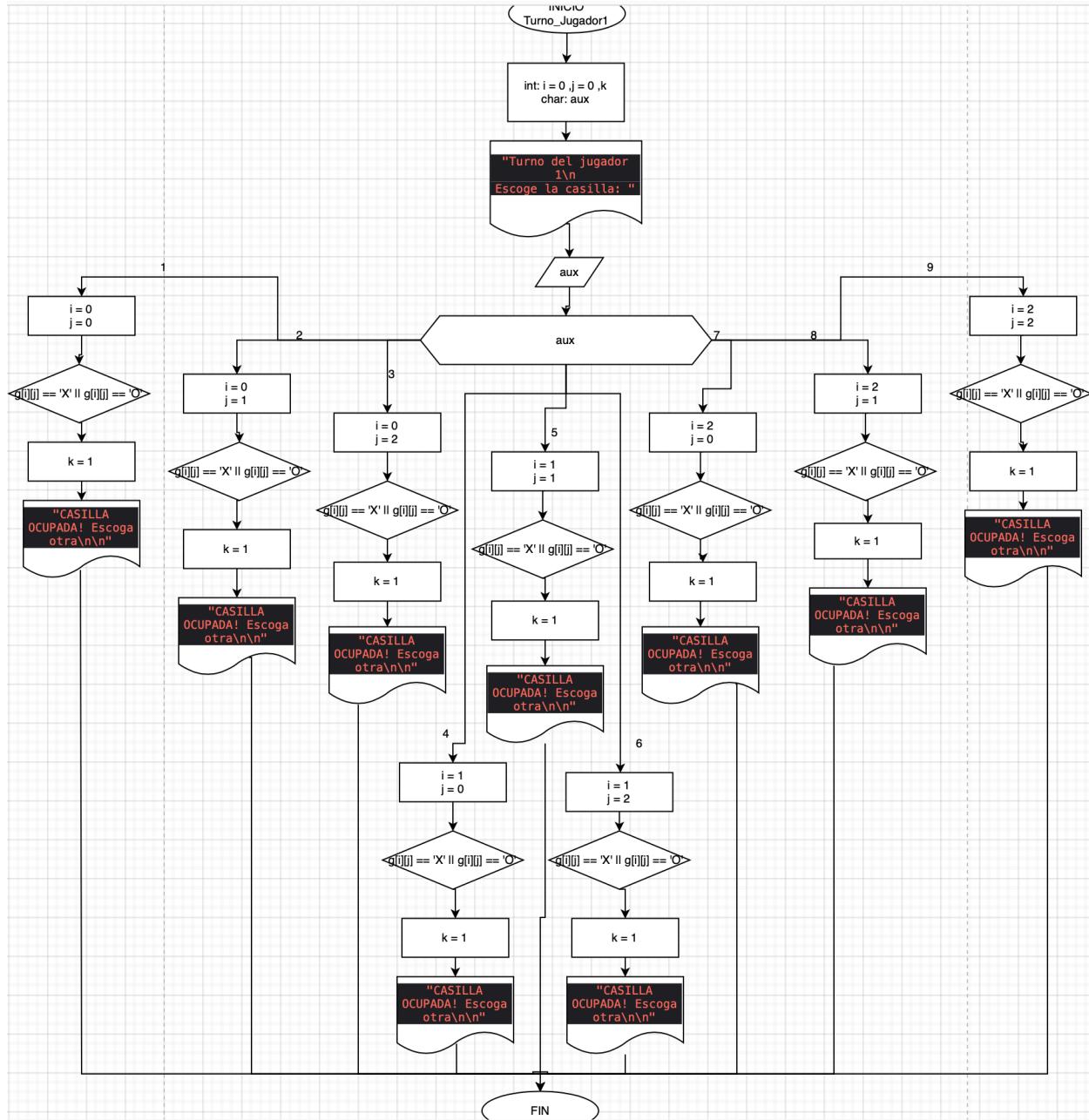
## Diagrama de Flujo Diagrama del menu

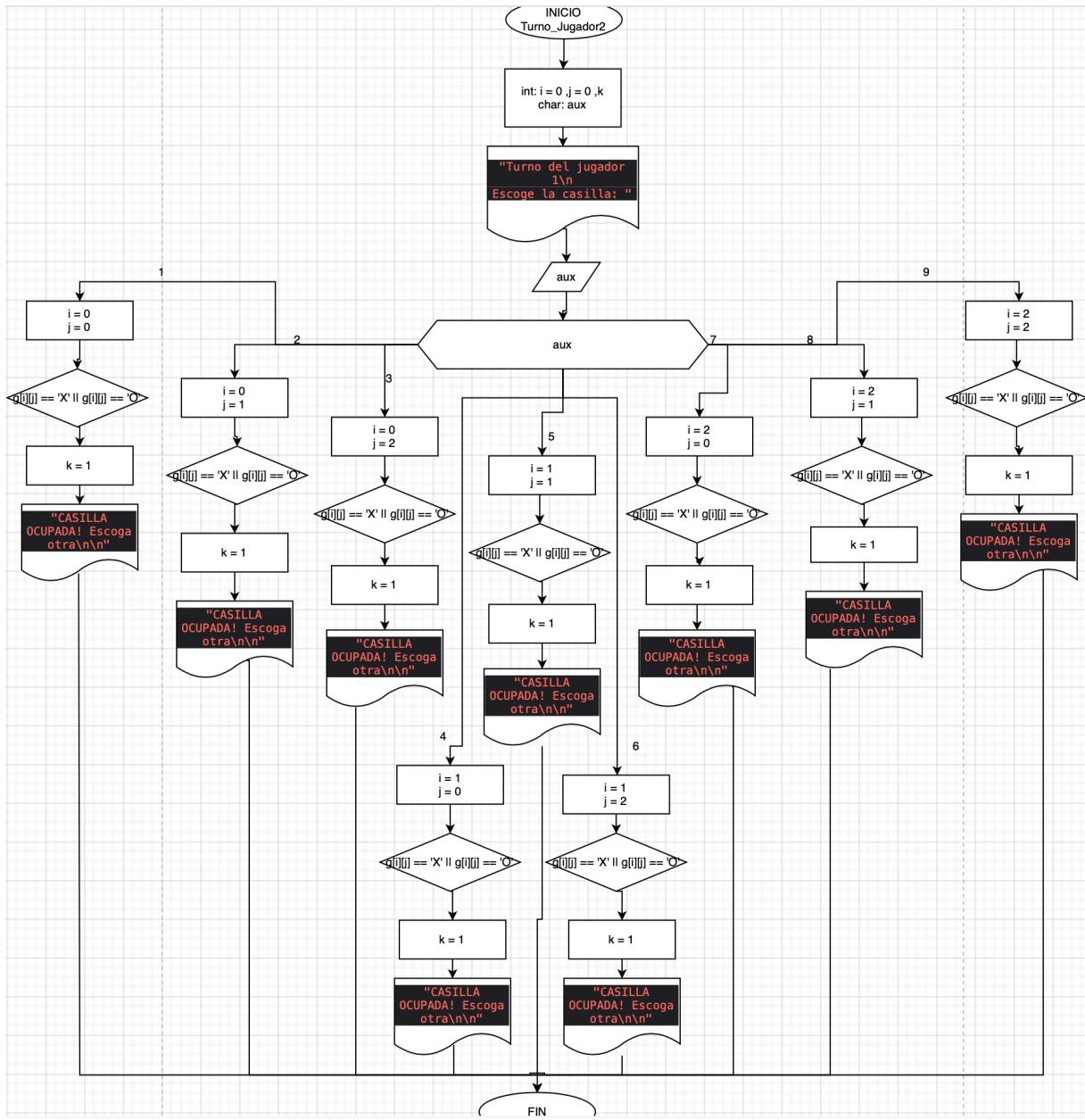


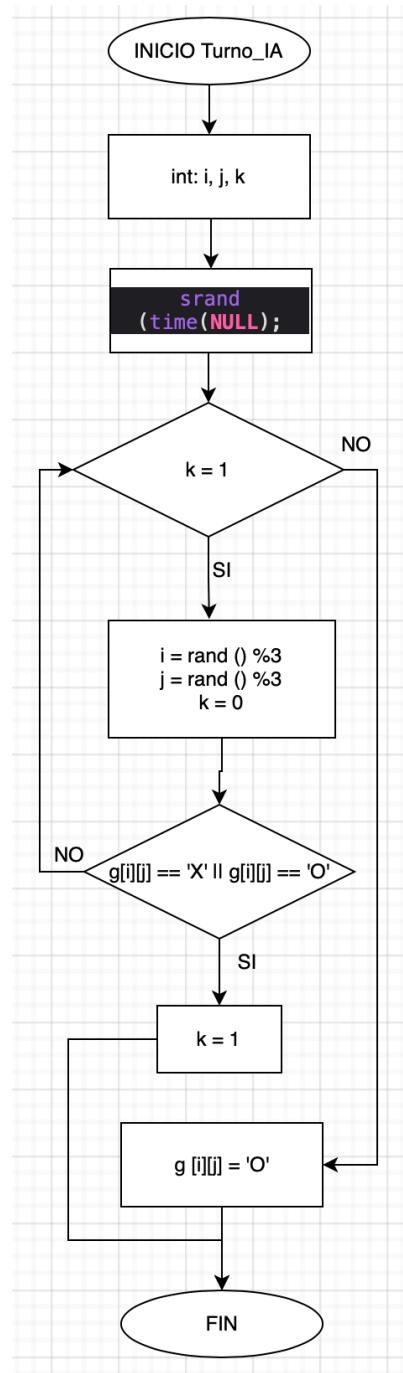


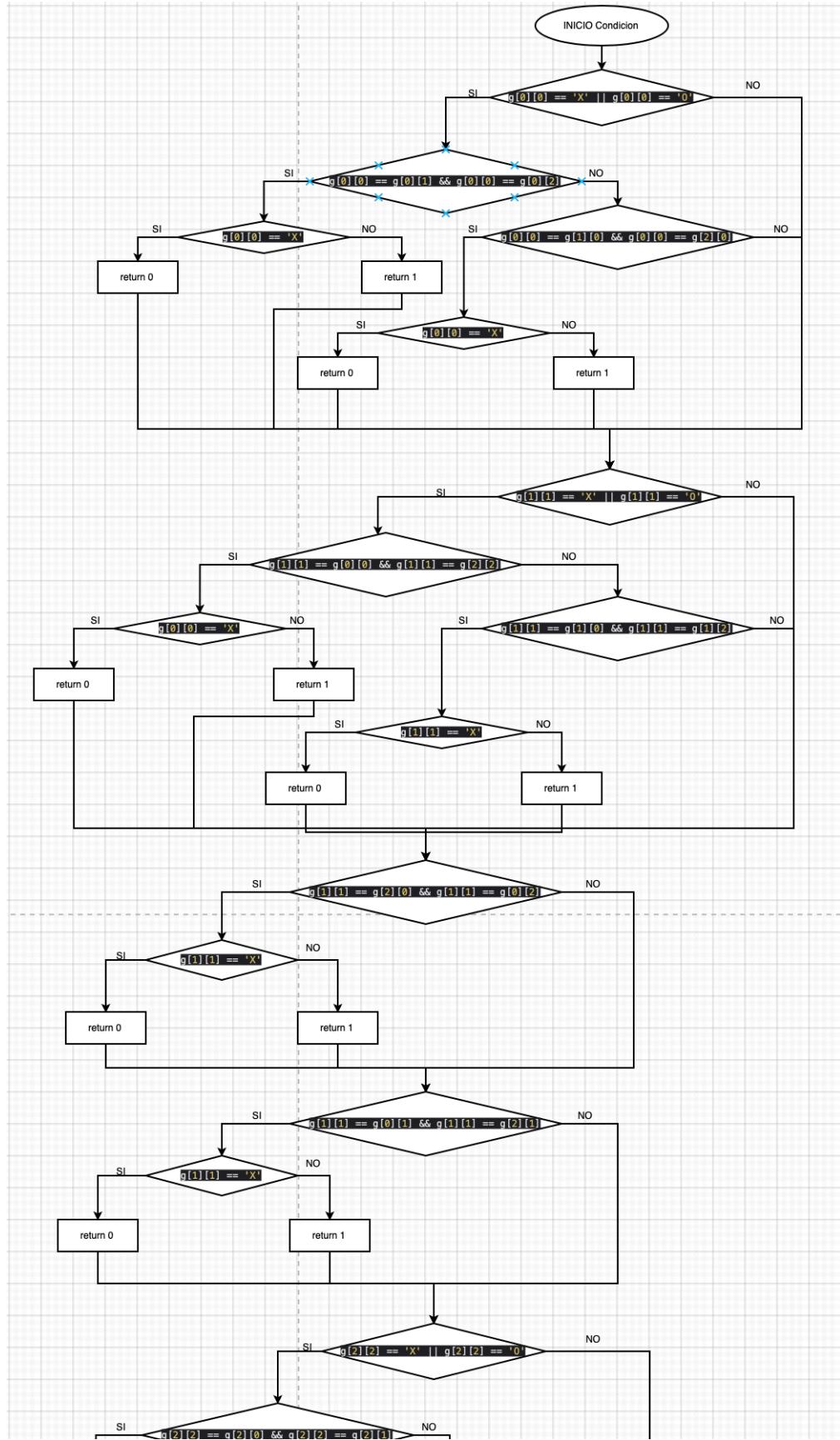




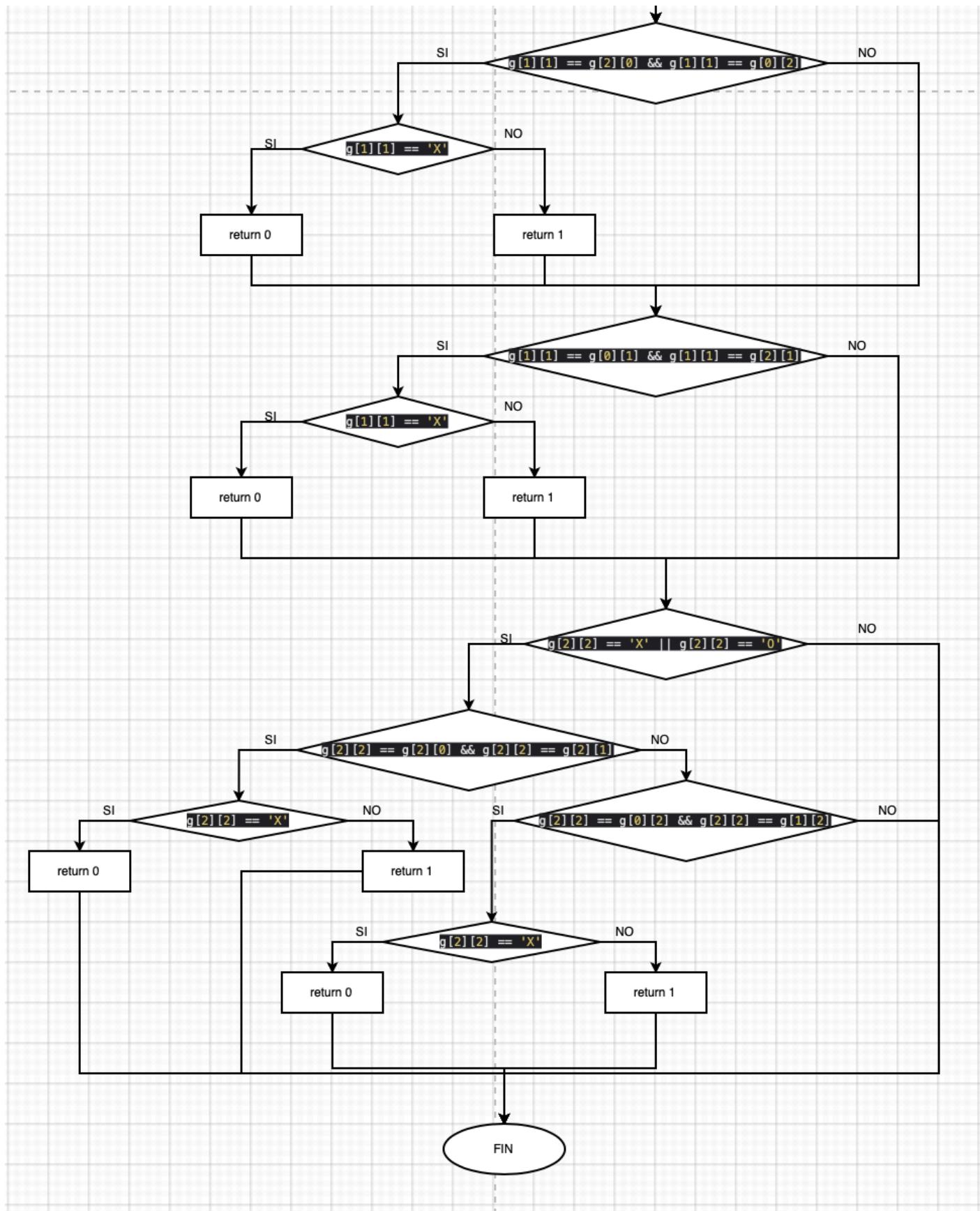


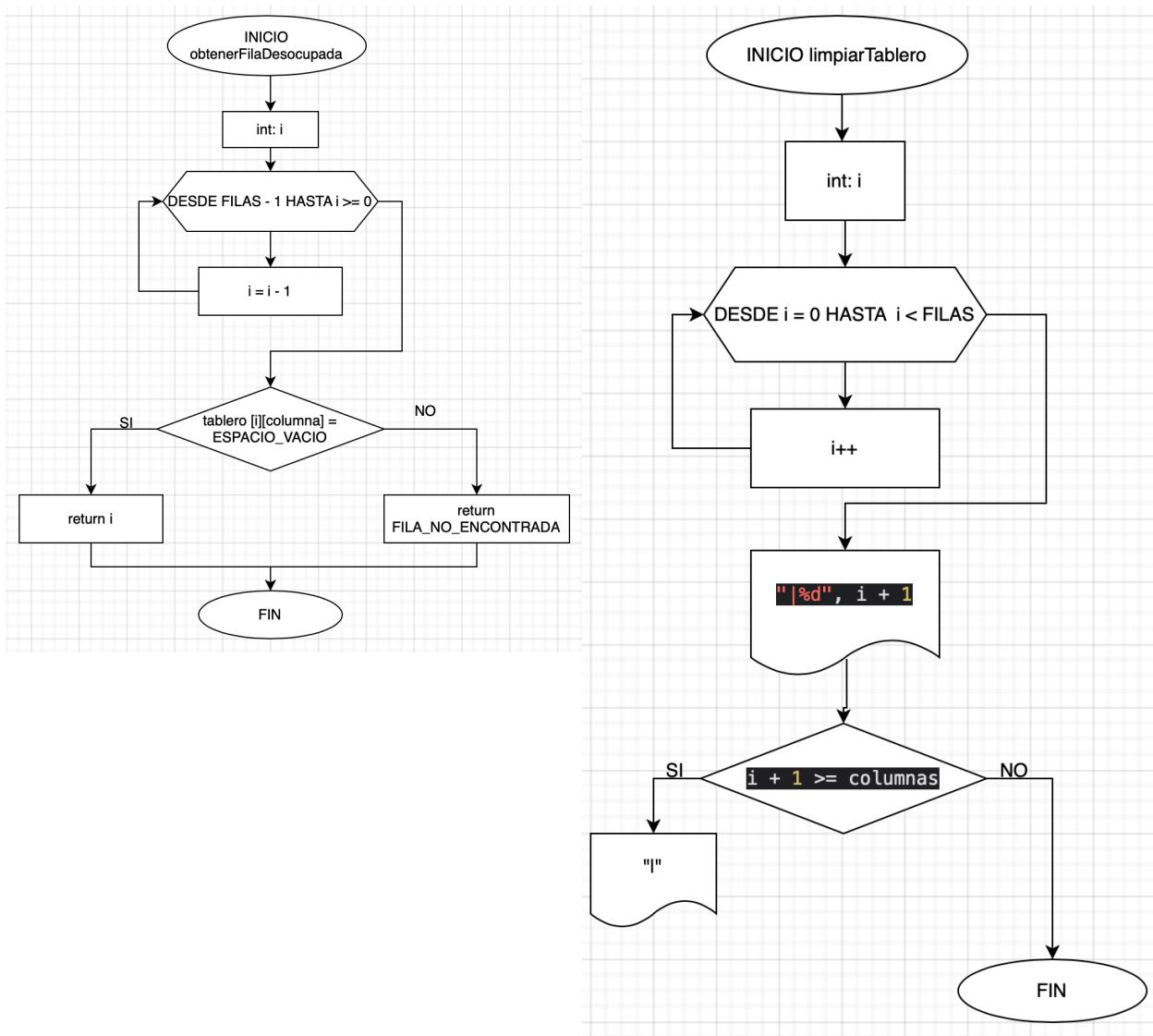


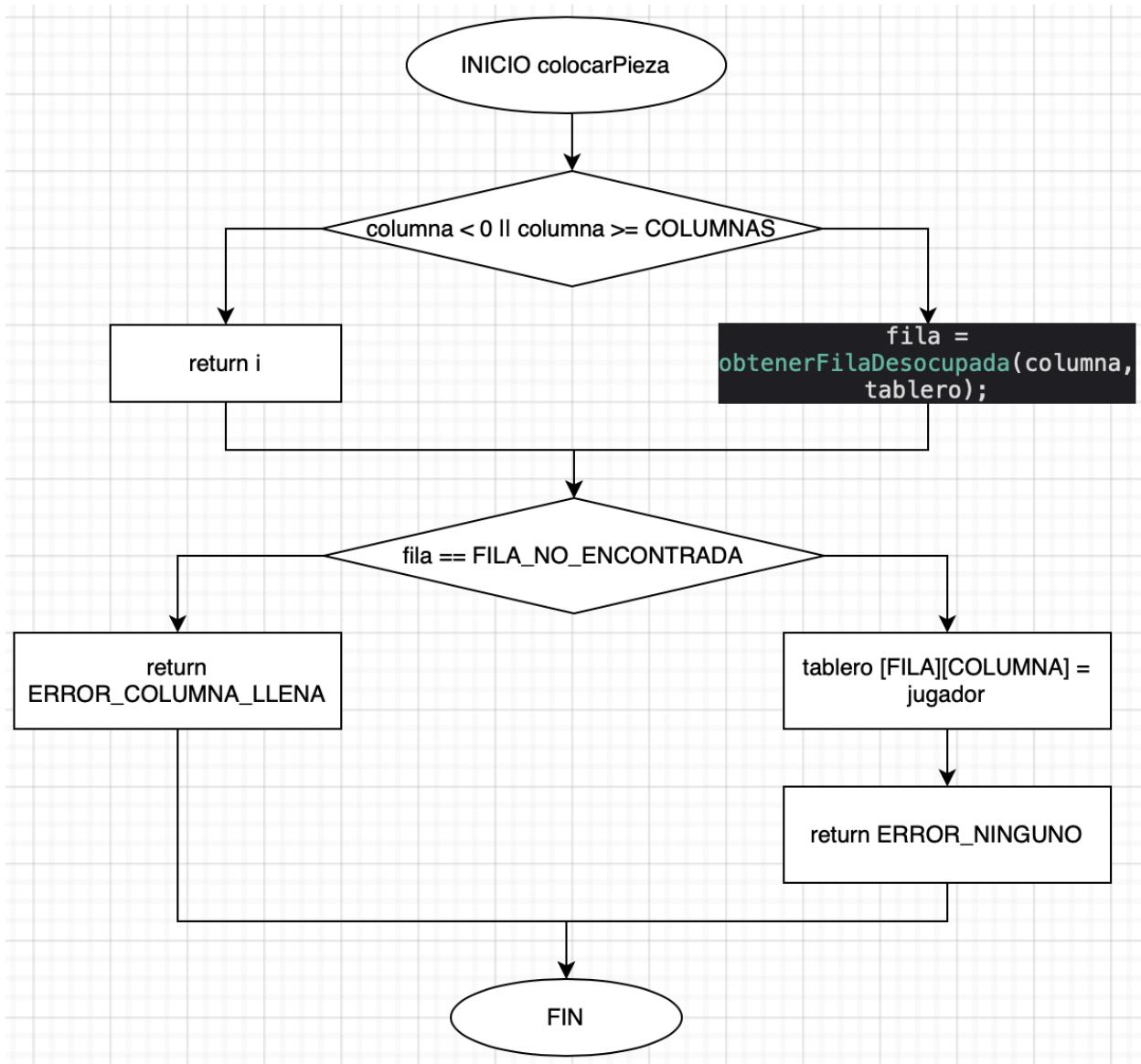


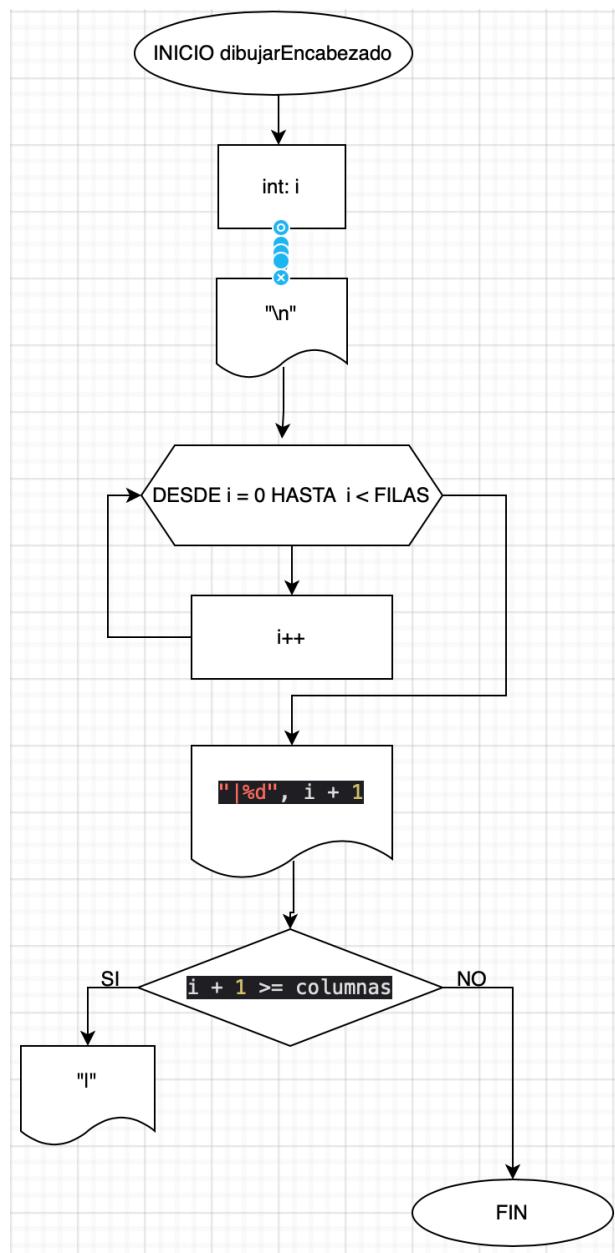
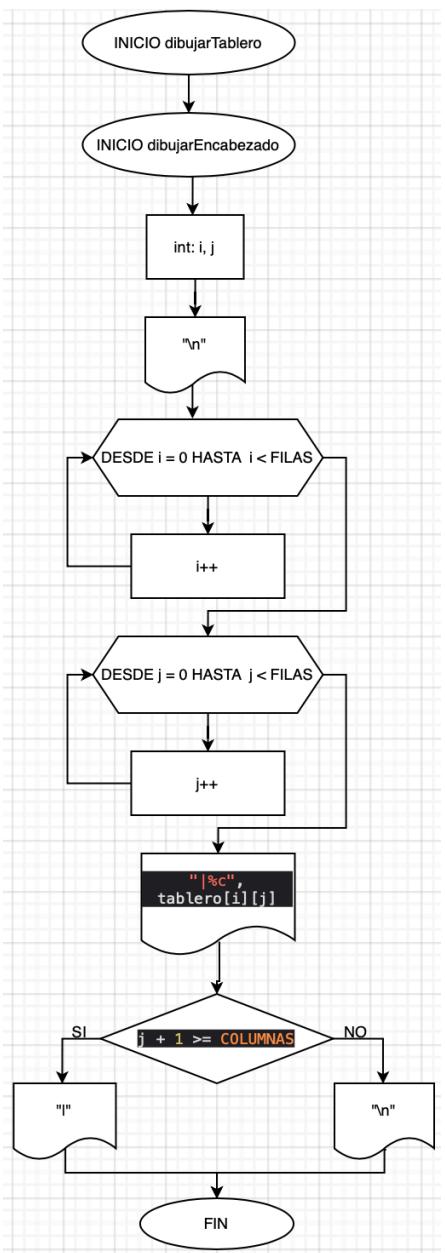


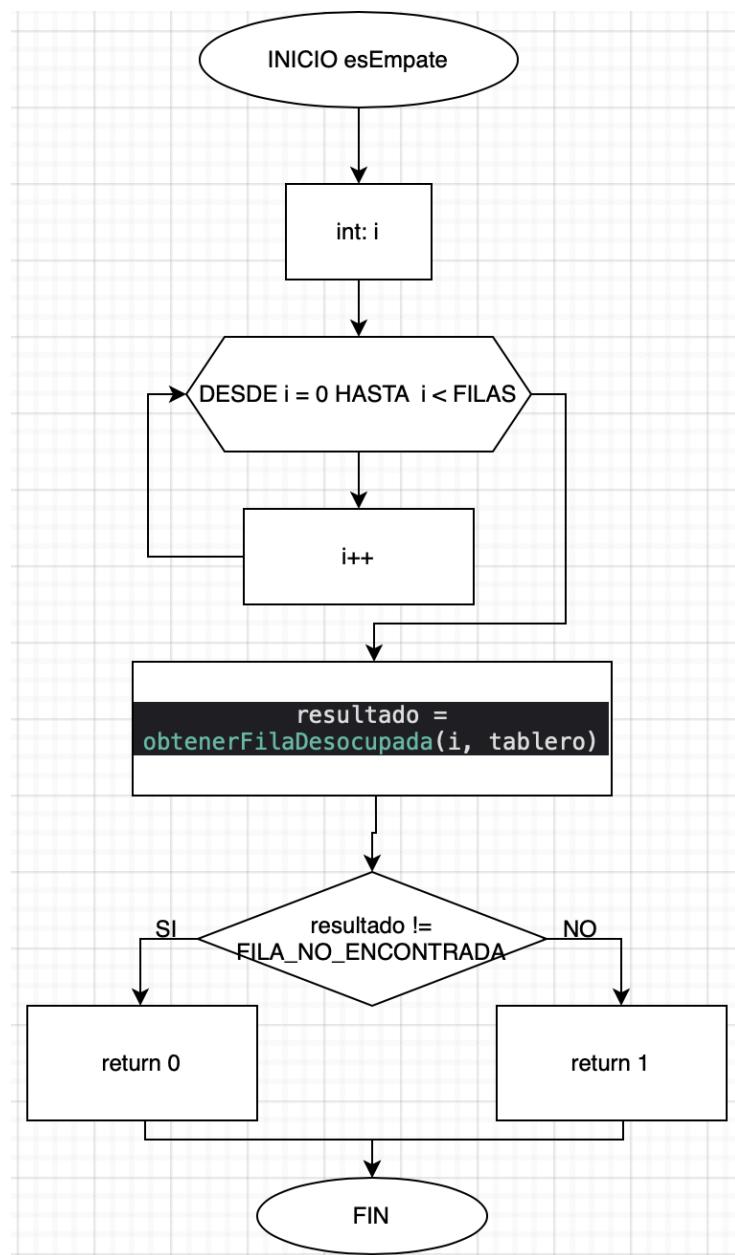
Esta es la continuación del diagrama de arriba

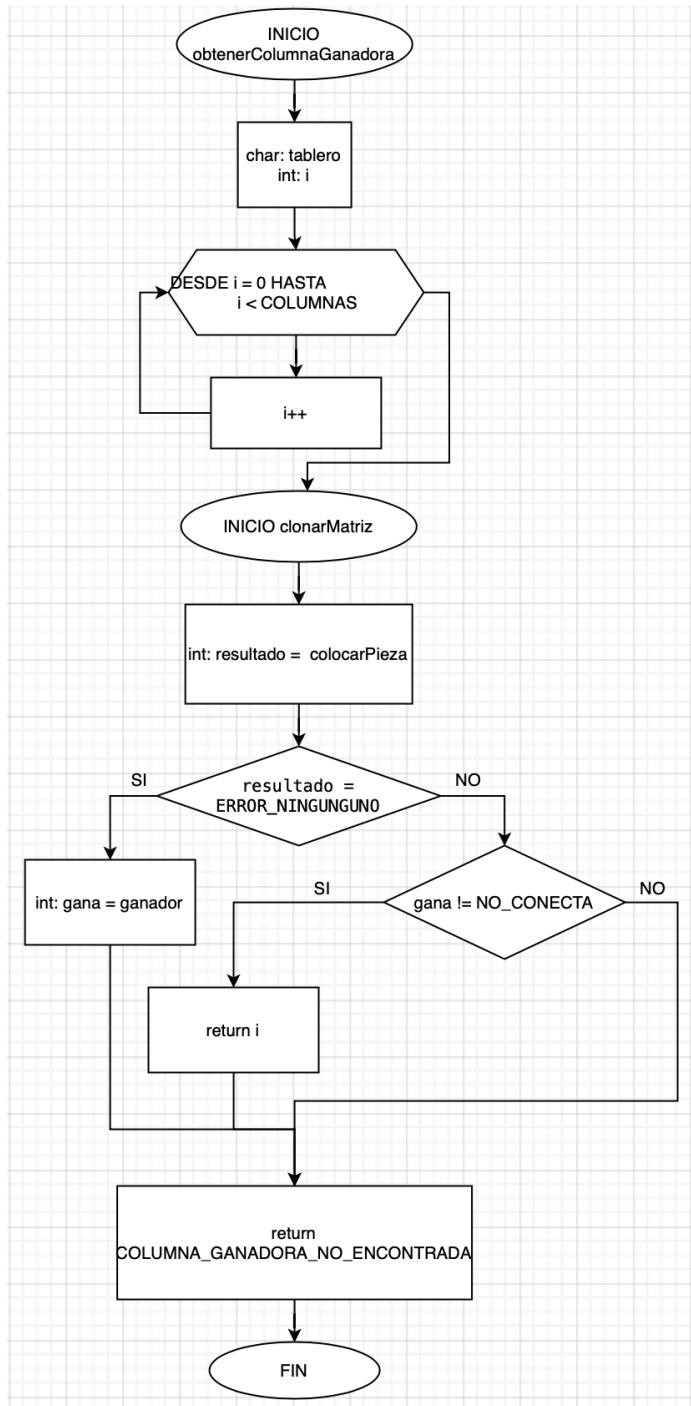


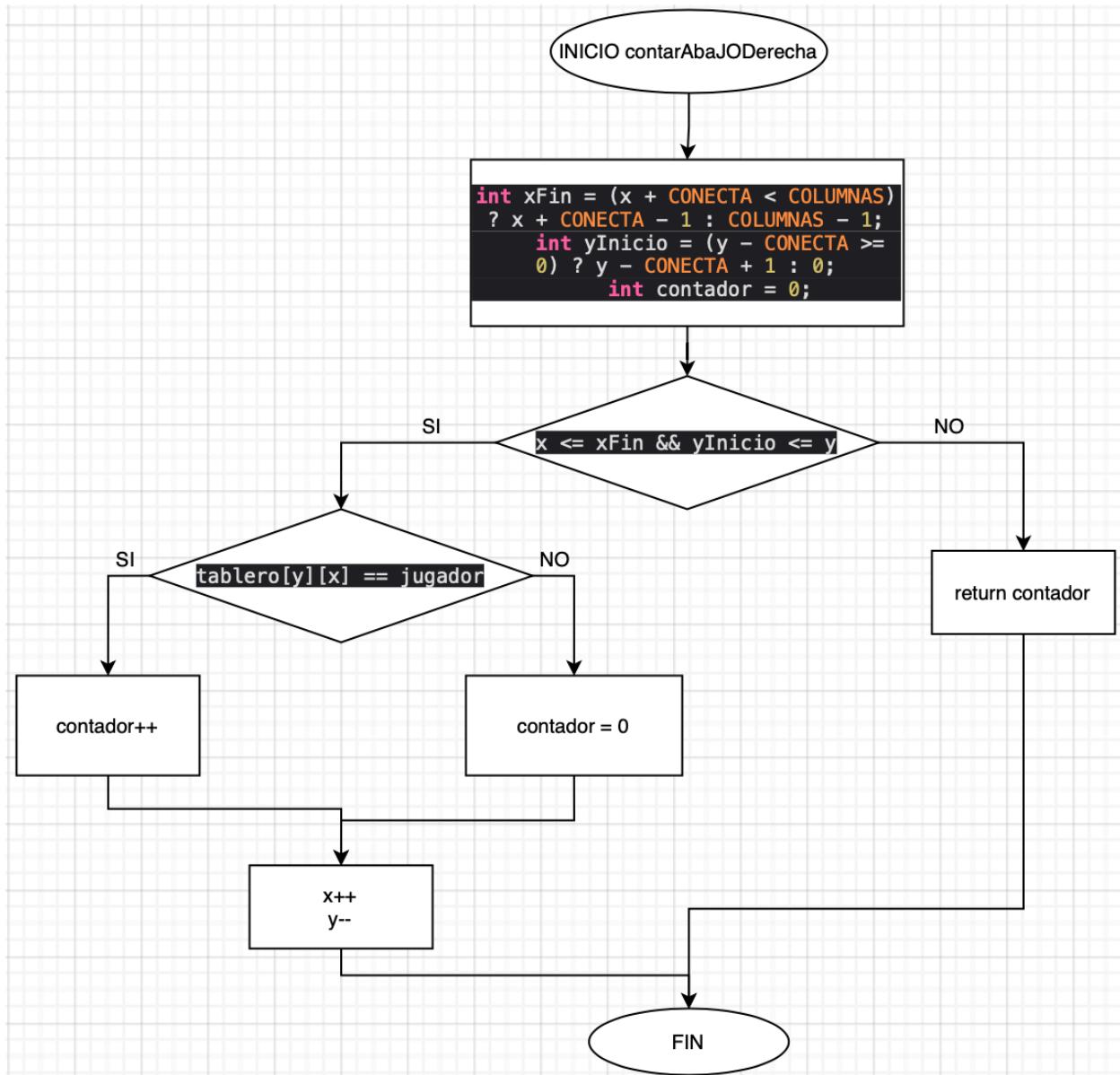


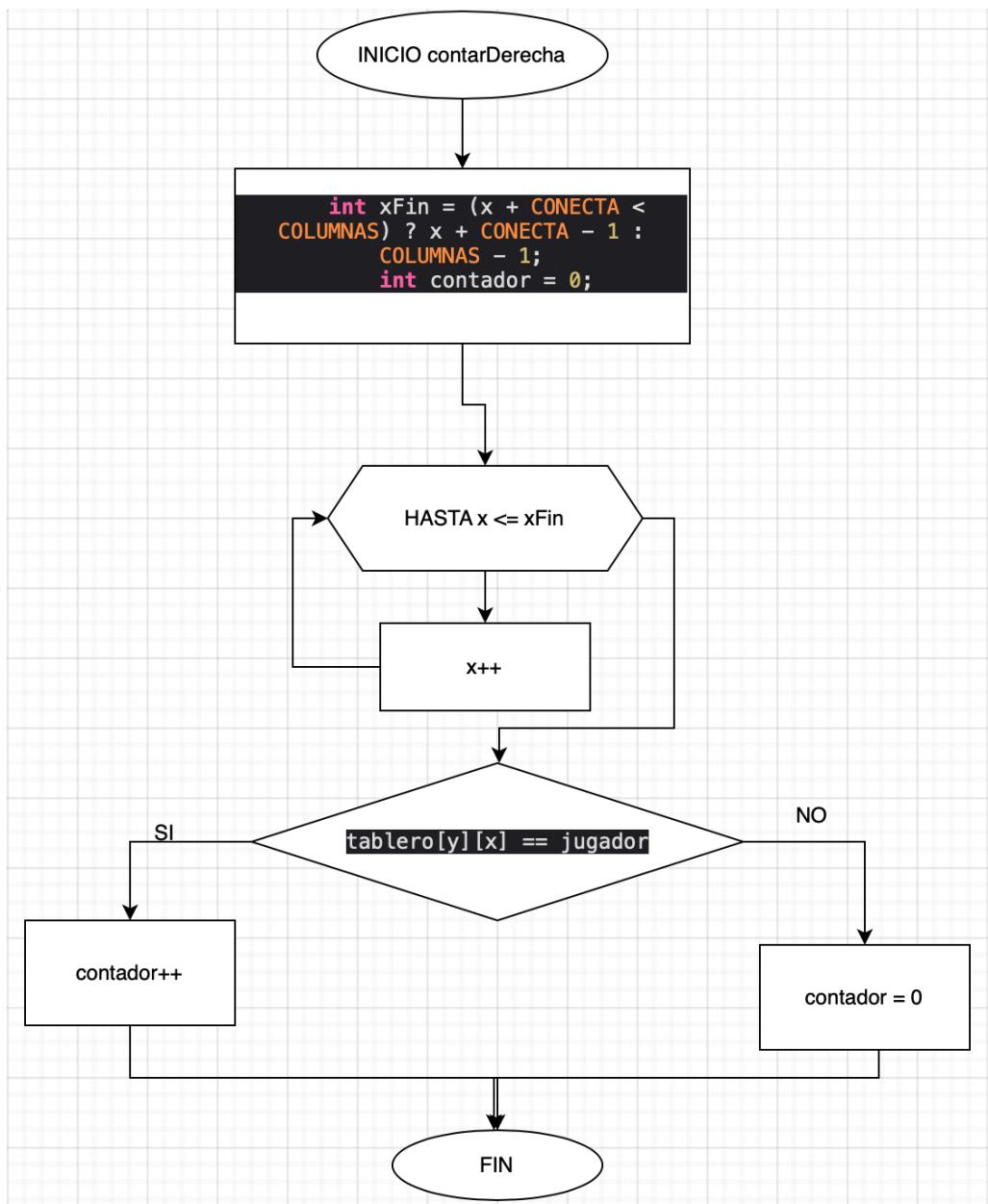


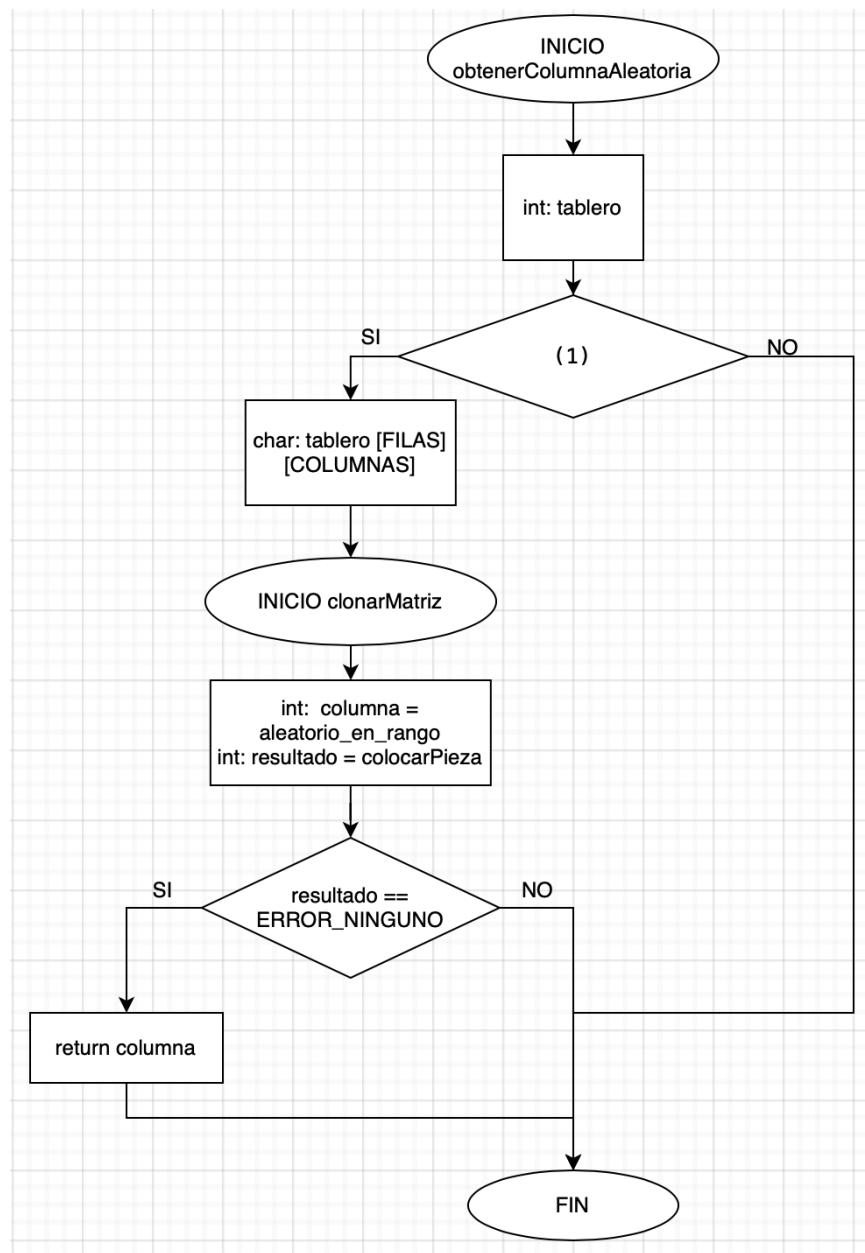


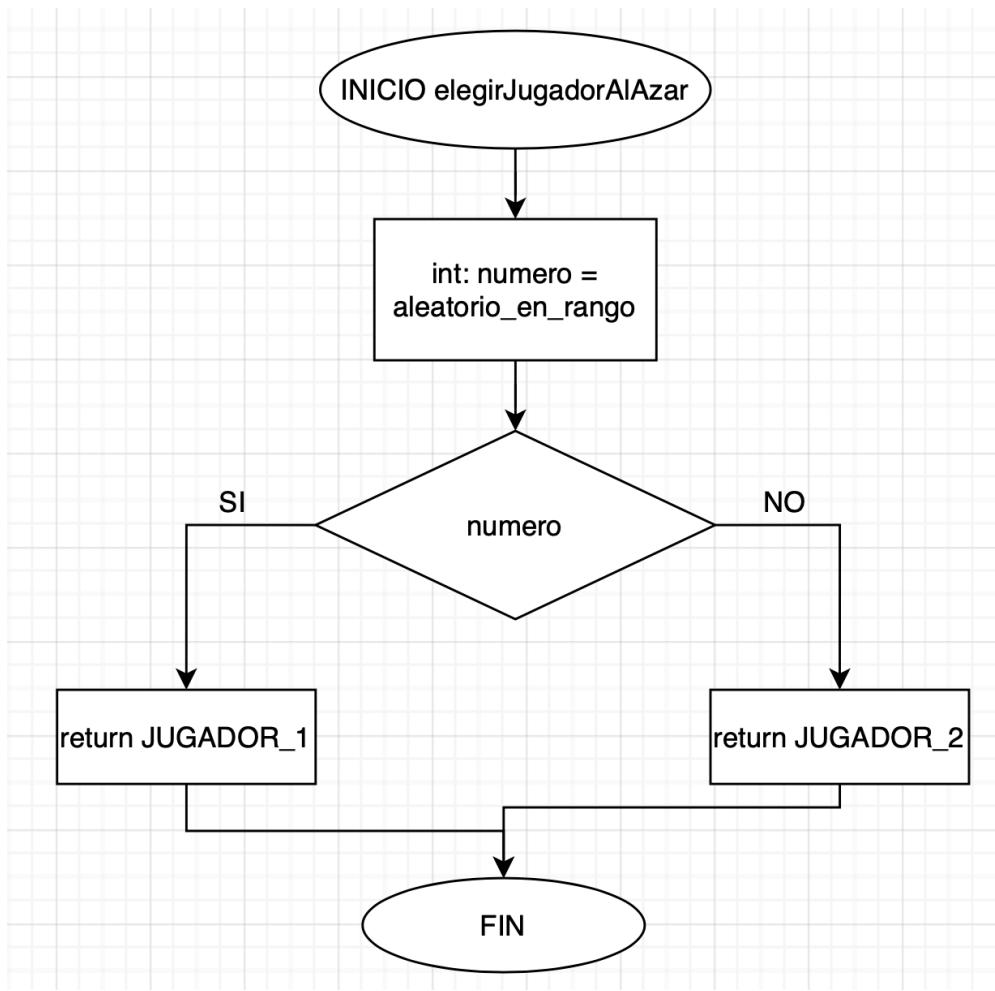


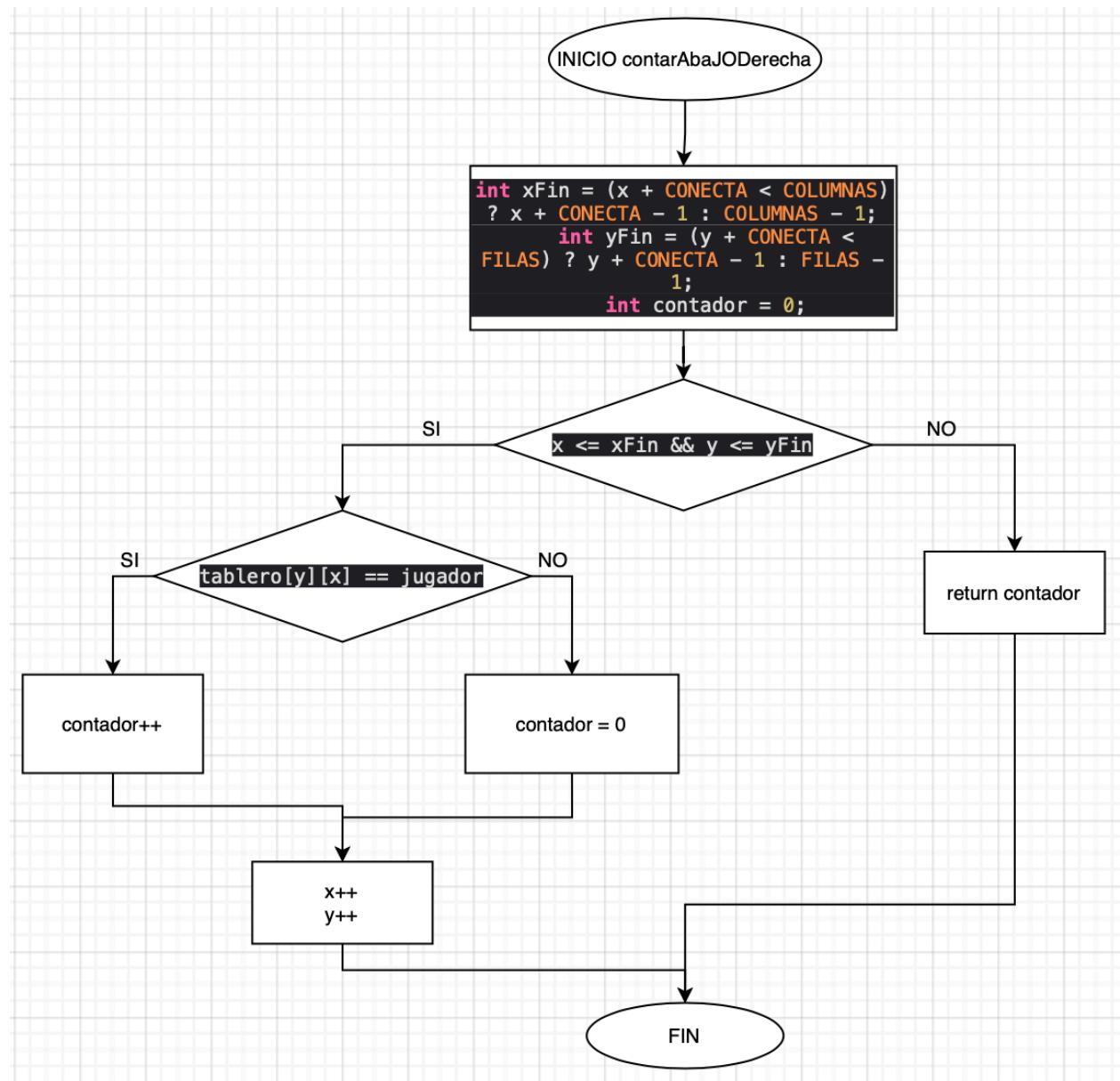


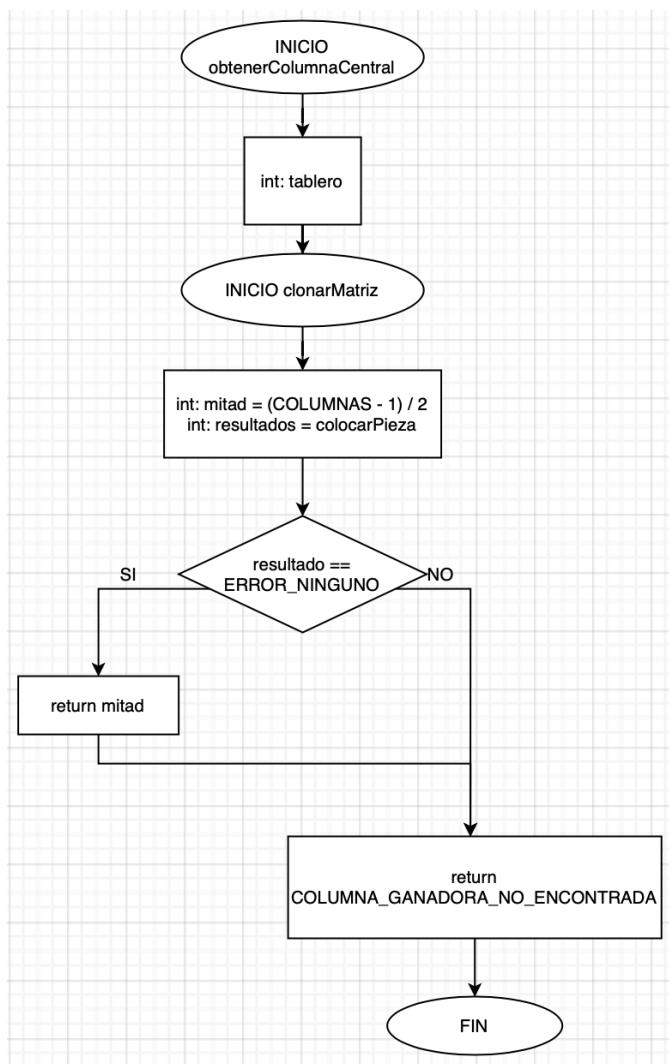
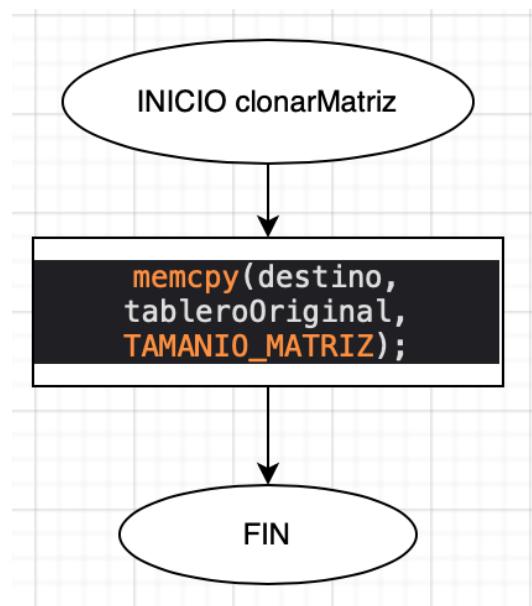


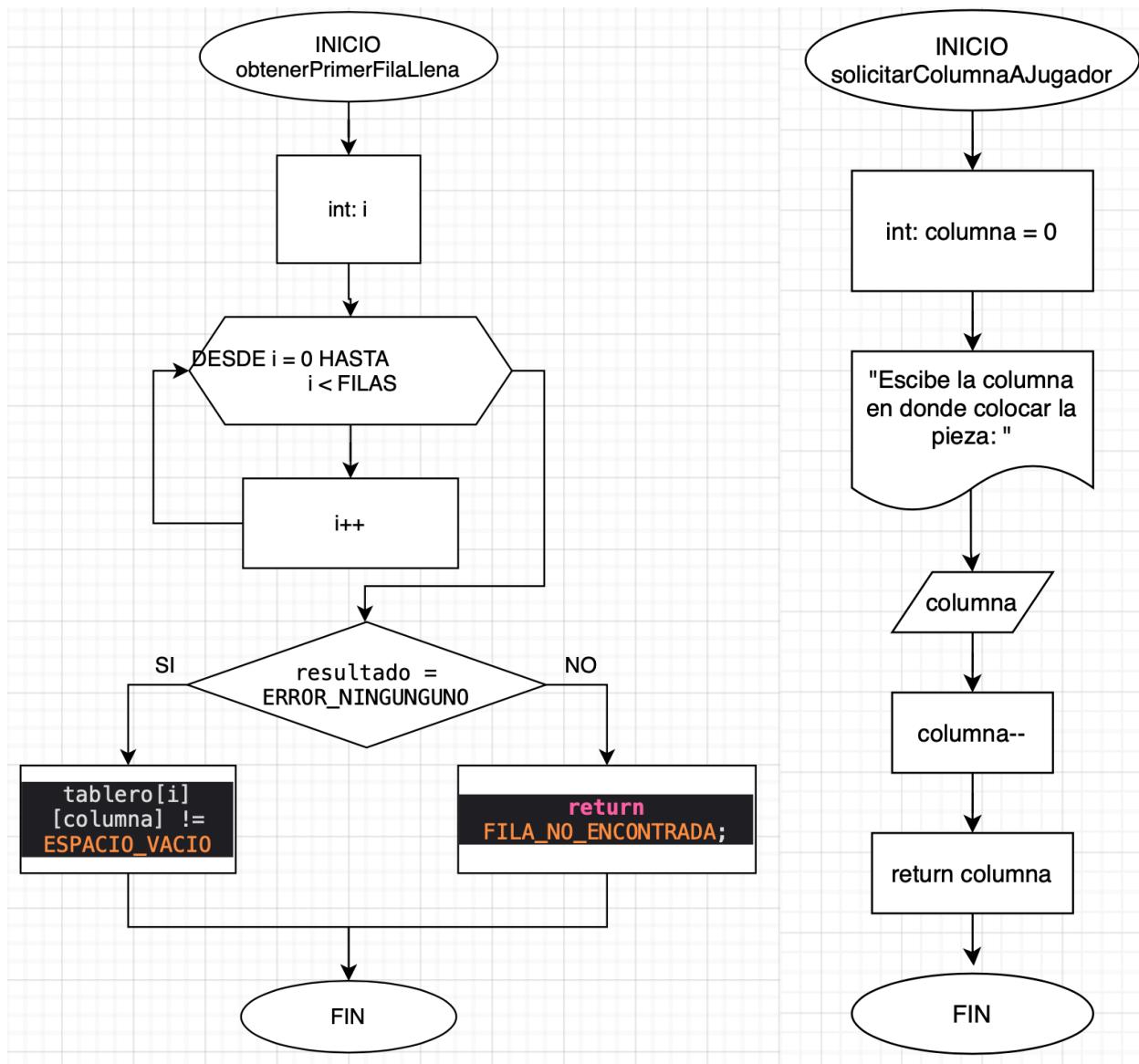
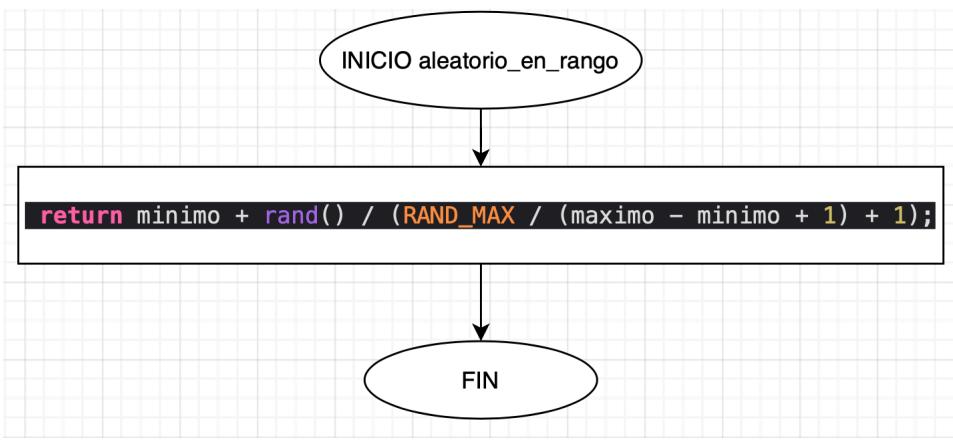


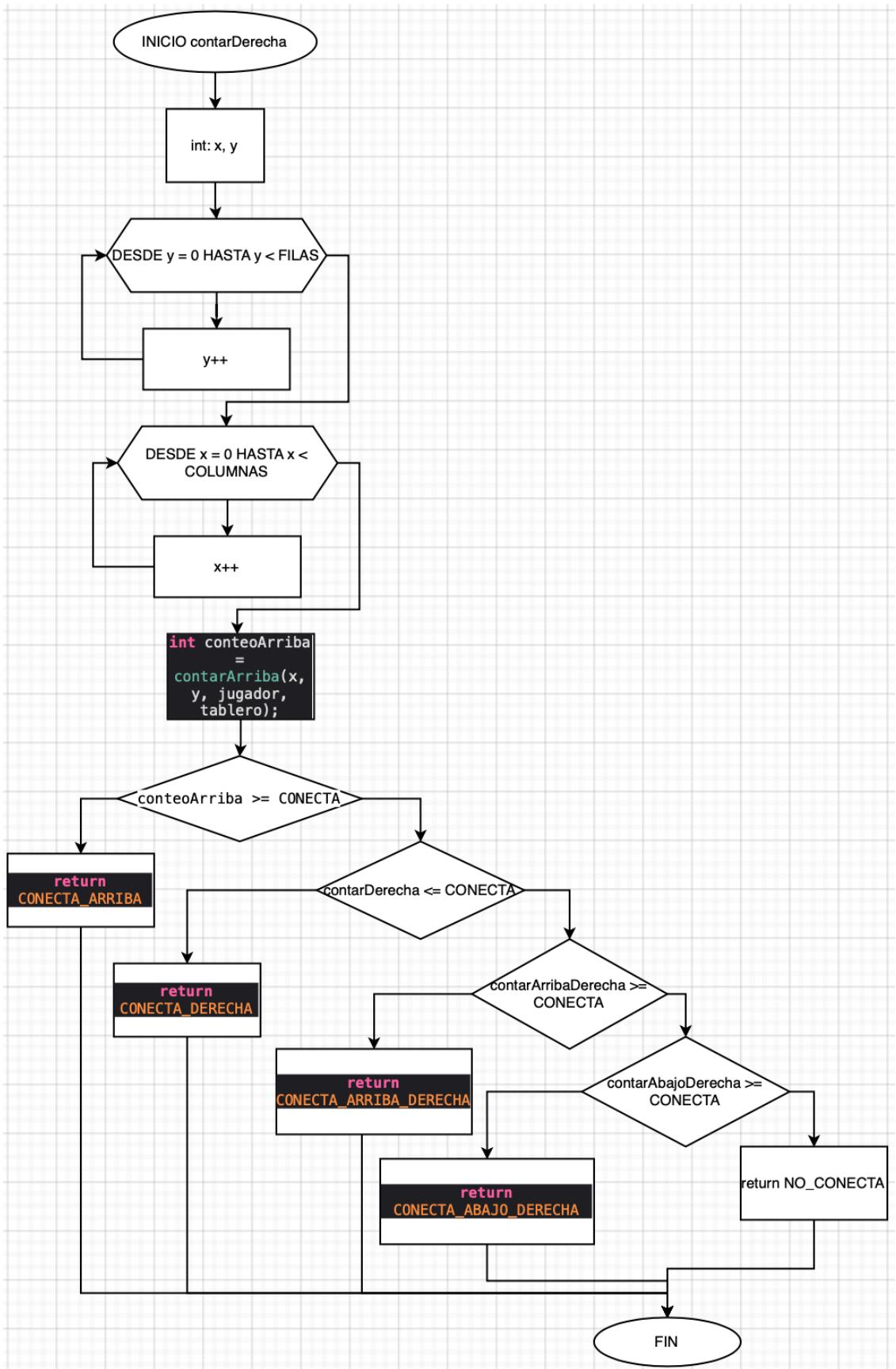


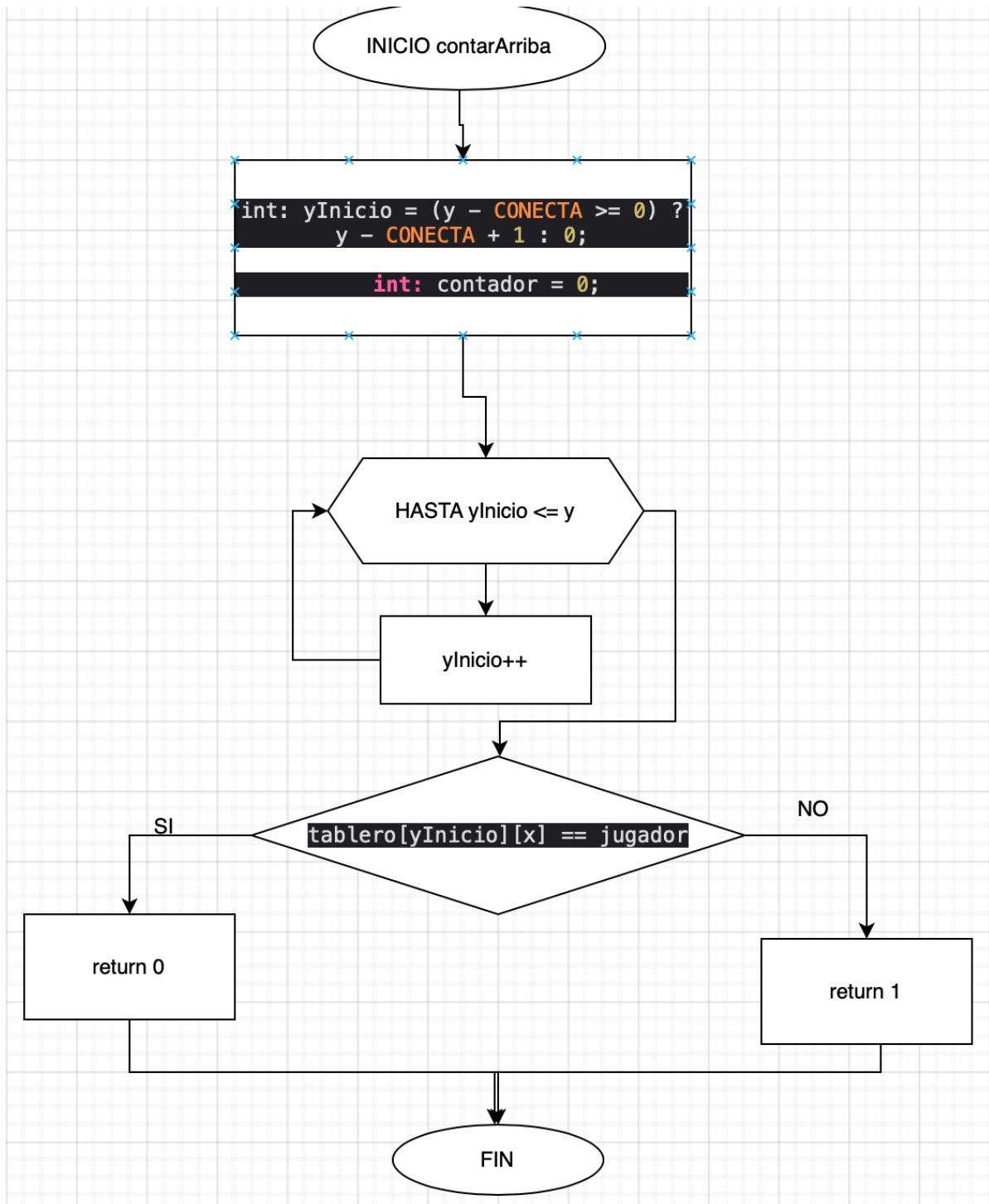


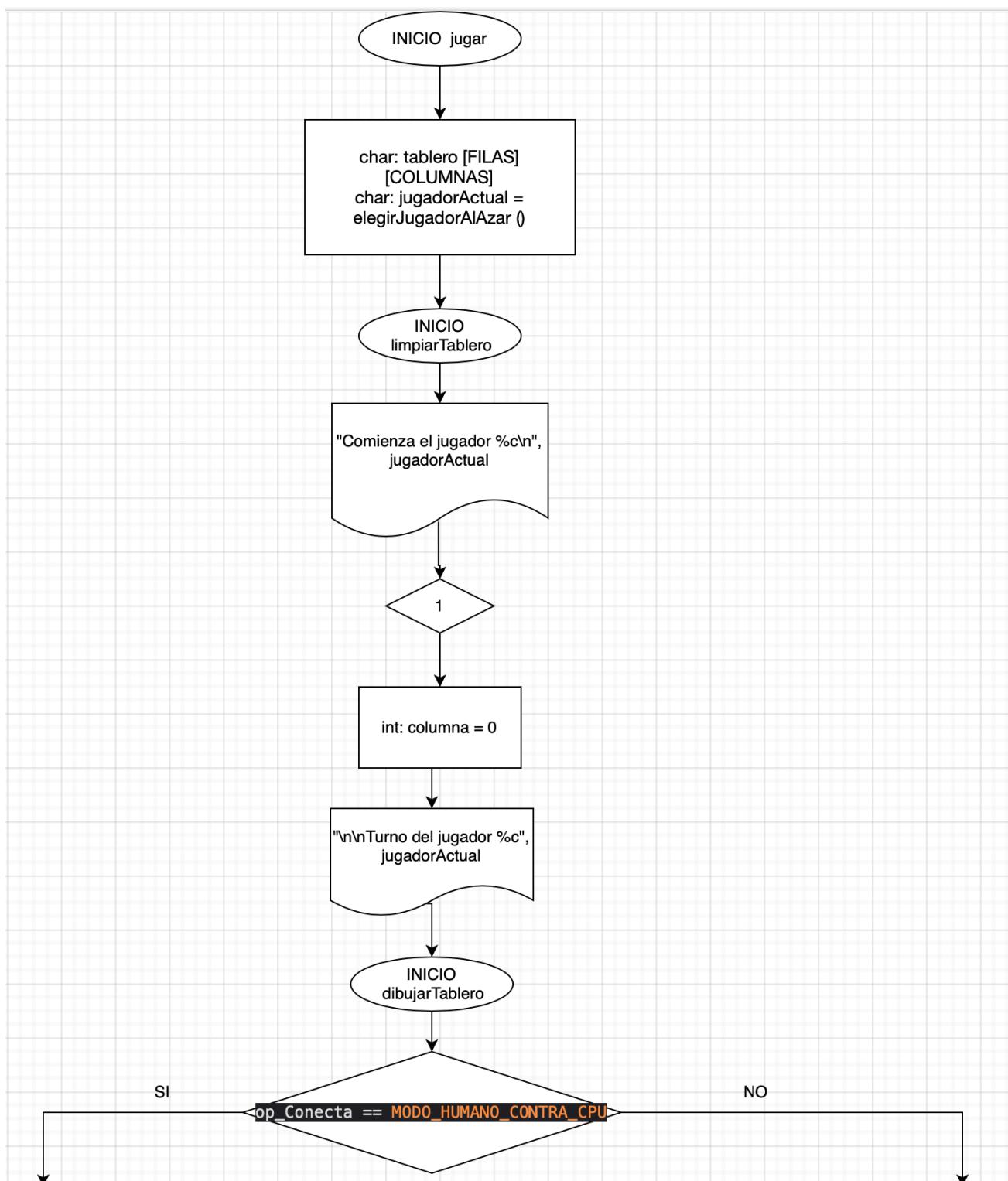




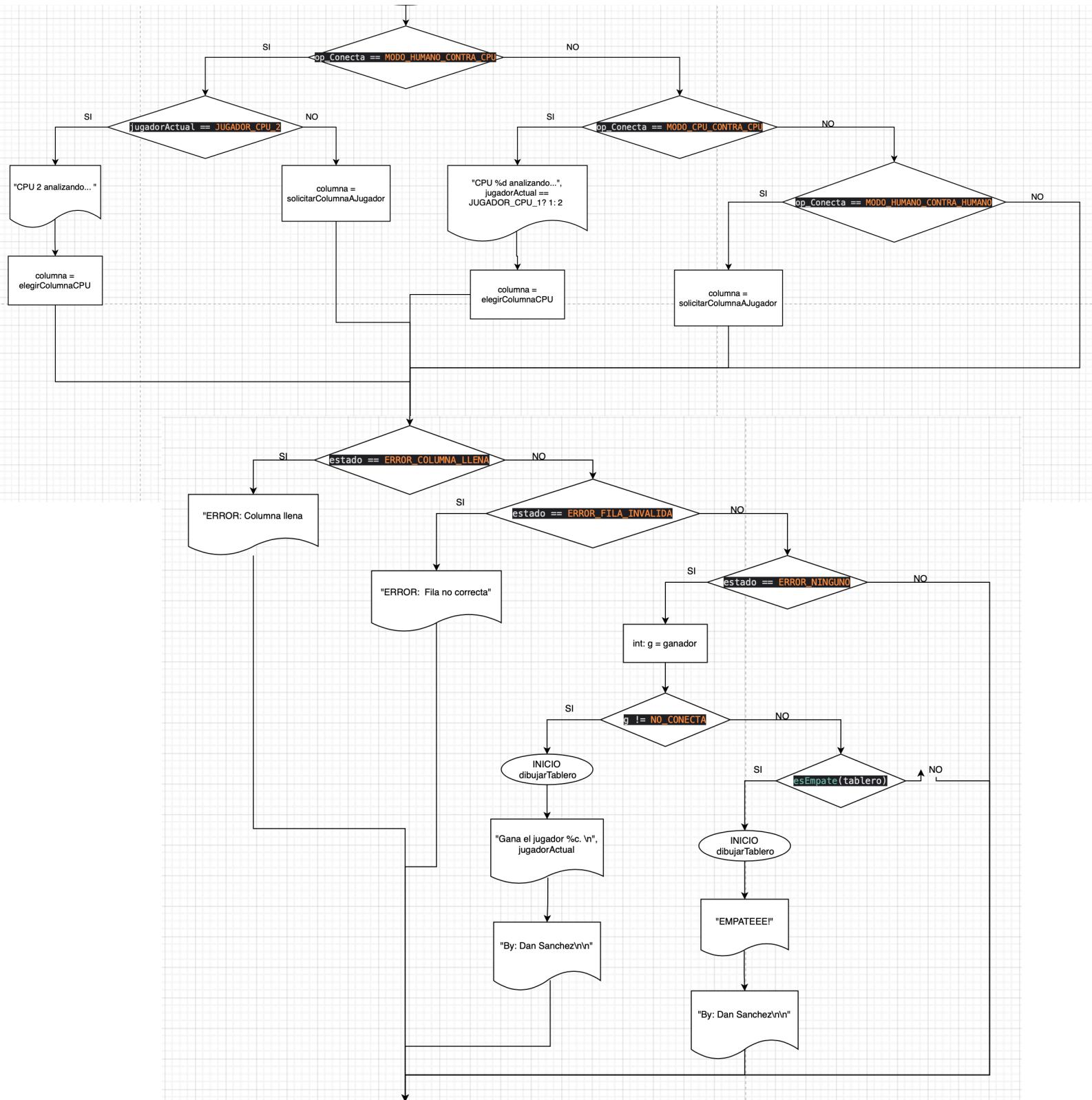








Esta es la continuación del diagrama de arriba



Esta es la continuación y fin del diagrama de las ultimas dos paginas

