Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación


**Curso:**

Lenguajes de Programación

Tarea Programada 04

Tema: Programación imperativa lenguaje C

**Profesora:**

María Auxiliadora Mora Cross

**Estudiante:**

Kevin Cubillo Chacón - 2021123138

Alajuela Junio, 2023

# Contenido

## Descripción del problema:

El proyecto busca desarrollar un programa en lenguaje C que implemente el juego Lau-kata-kati. Este juego de estrategia es similar a las Damas chinas, pero con reglas simplificadas y un tablero con forma de mariposa. El objetivo del juego es capturar las fichas del oponente o bloquearlas para evitar que se muevan.

El tablero de juego es representado como un grafo bidireccional, donde cada nodo del grafo representa una posición en el tablero y las aristas del grafo representan las conexiones entre las posiciones. El tablero está formado por triángulos de diferentes profundidades, donde la profundidad se refiere al número de filas desde el punto central del tablero hacia los lados. Por ejemplo, un triángulo de profundidad 3 tendría 3 filas a cada lado del punto central. La estructura de datos debe ser implementada únicamente con listas enlazadas, no está permitido usar arreglos o bloques de memoria.

Cada jugador tiene un conjunto de 9 fichas, que se colocan en la base del triángulo del tablero correspondiente a su lado. Los jugadores alternan turnos y pueden elegir quién comienza y la ubicación de sus fichas en el lado izquierdo o derecho del tablero. Durante su turno, un jugador puede mover una ficha a una posición adyacente vacía siguiendo las conexiones del tablero. También se permiten capturas, donde una ficha enemiga adyacente puede ser saltada y capturada si hay una posición vacía al otro lado de la ficha enemiga.

El juego continúa hasta que uno de los jugadores capture todas las fichas del oponente o bloquee todas sus fichas para que no puedan moverse. En caso de que ningún jugador pueda realizar movimientos legales, se produce un punto muerto y el jugador con más fichas restantes es declarado ganador. Si ambos jugadores tienen la misma cantidad de fichas al final del juego, se considera un empate.

## Diseño de componentes:

El programa implementa el juego "Lau-Kata-Kati" y utiliza los siguientes componentes de diseño:

1. Estructuras de datos:
    - **struct Cell**: Representa una celda en el tablero del juego (nodo). Contiene un campo **token** que almacena el jugador que posee la celda (1 para jugador X, 2 para jugador O) y un puntero **next** que apunta a la siguiente celda en el tablero.
    - **struct Board**: Representa el tablero del juego. Contiene un campo **numCells** que indica el número total de celdas en el tablero y un puntero **root** que apunta a la primera celda del tablero (lista enlazada).
2. Mantenimiento del estado del juego:
    - **createBoard()**: Crea y devuelve un nuevo tablero del juego. Inicializa las celdas del tablero y asigna las fichas iniciales a los jugadores.
    - **moveToken()**: Mueve una ficha desde una casilla de origen a una casilla de destino en el tablero. Actualiza el estado del tablero en función del movimiento realizado.
    - **hasValidMoves()**: Verifica si un jugador tiene movimientos válidos disponibles en el tablero. Comprueba todas las celdas ocupadas por el jugador y verifica si hay movimientos válidos desde esas celdas.
    - **checkWinner()**: Verifica si hay un ganador en el juego. Comprueba el número de fichas de cada jugador y si alguno de ellos no tiene movimientos válidos disponibles. Devuelve un código de ganador (1 para jugador X, 2 para jugador O, 3 si hay empate y 0 si no hay ganador).

3. Jugadores:
    - **currentPlayer**: Representa el jugador actual que está realizando un movimiento en el juego. Puede ser el jugador X (1) o el jugador O (2).
    - **topPlayer**: Representa el jugador que juega en el lado superior del tablero. Puede ser el jugador X (1) o el jugador O (2).

El programa sigue un ciclo de juego en el que los jugadores realizan movimientos alternativamente hasta que haya un ganador o un empate. Al final del juego, se muestra el tablero final y se anuncia el resultado (ganador o empate).

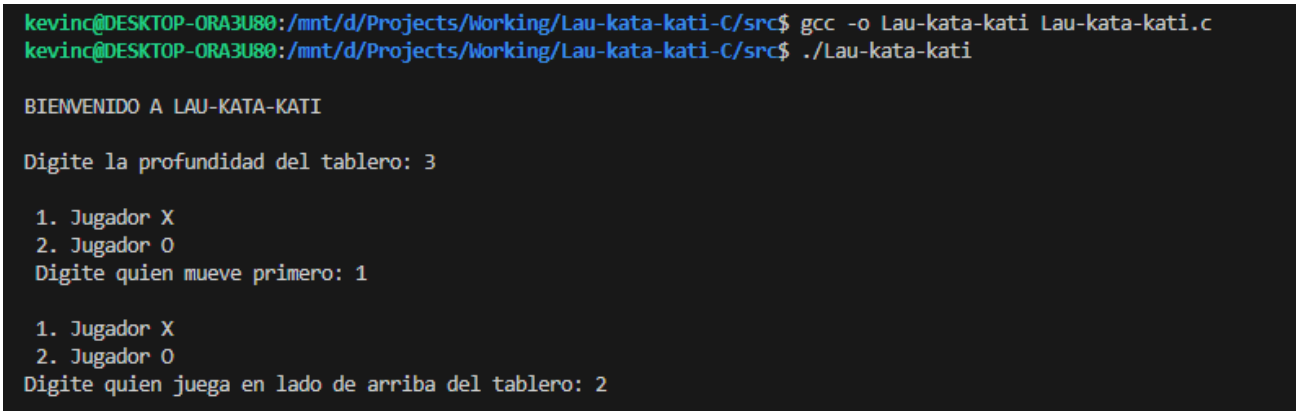
## Como ejecutar el código para jugar:

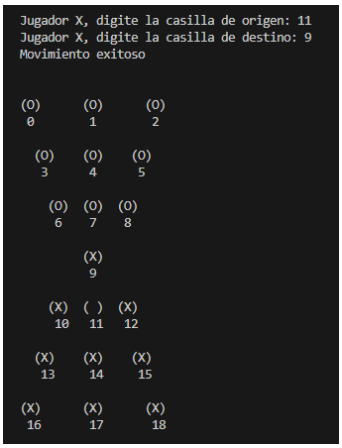
1. Compilación y ejecución:
    - Compila el programa ejecutando el siguiente comando en la terminal: **gcc -o Lau-kata-kati Lau-kata-kati.c**
    - En la terminal, ejecuta el programa escribiendo el siguiente comando: **./Lau-kata-kati**
    - Aparecerá un mensaje de bienvenida y se te pedirá ingresar ciertos valores para configurar el juego, como la profundidad del tablero, qué jugador mueve primero y qué jugador juega en el lado superior del tablero.
2. Juego:
    - Una vez configurado el juego, se mostrará el tablero inicial en la terminal.

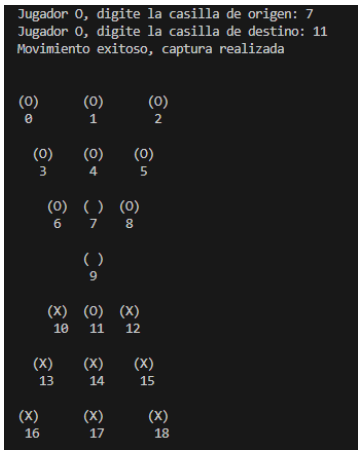
- El programa pedirá que se ingrese la casilla de origen y la casilla de destino para realizar un movimiento. Se indicará qué jugador debe realizar el movimiento.
- Ingresa los números de las casillas de origen y destino según las indicaciones.
- El programa verificará si el movimiento es válido y actualizará el estado del tablero en consecuencia.
- El turno pasará al siguiente jugador, y se pedirá el ingreso de los movimientos en cada turno hasta que haya un ganador o un empate.

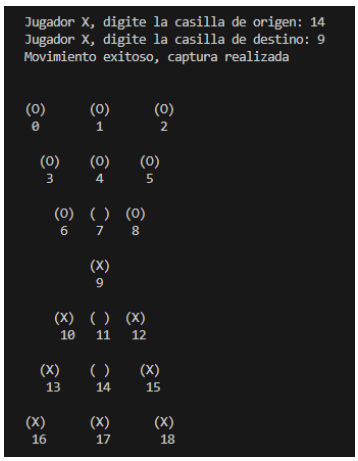
3. Finalización:
    - Una vez que el juego haya terminado (ya sea porque hay un ganador o un empate), se mostrará el tablero final en la terminal.
    - El programa anunciará el resultado del juego, indicando qué jugador
    - ha ganado o si ha habido un empate.

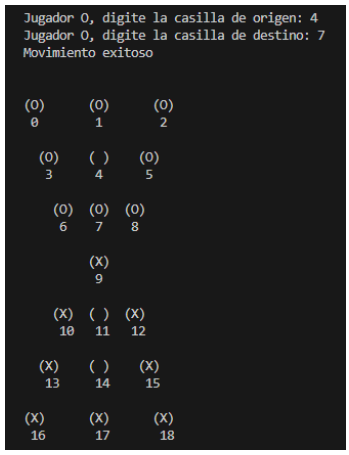
## Capturas de una corrida:

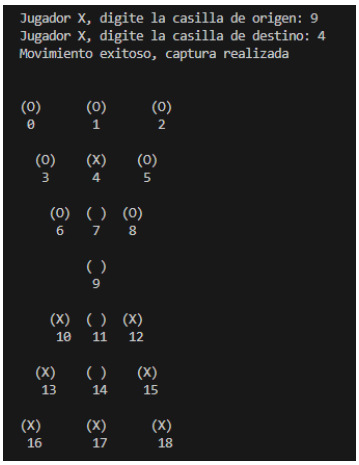


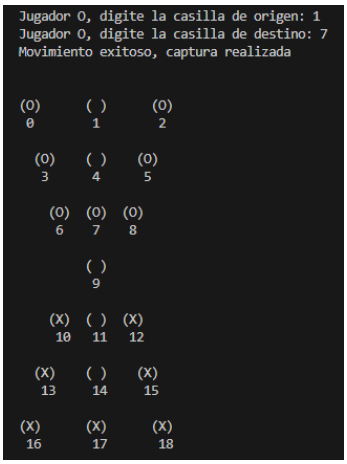

Jugador X, digite la casilla de origen: 11
Jugador X, digite la casilla de destino: 9
Movimiento exitoso
(0) (0) (0)
0 1 2
(0) (0) (0)
3 4 5
(0) (0) (0)
6 7 8
(X)
9
(X) ( ) (X)
10 11 12
(X) (X) (X)
13 14 15
(X) (X) (X)
16 17 18

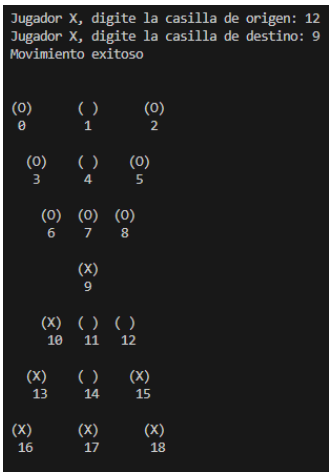

Jugador O, digite la casilla de origen: 7
Jugador O, digite la casilla de destino: 11
Movimiento exitoso, captura realizada
(0) (0) (0)
0 1 2
(0) (0) (0)
3 4 5
(0) ( ) (0)
6 7 8
( )
9
(X) (0) (X)
10 11 12
(X) (X) (X)
13 14 15
(X) (X) (X)
16 17 18

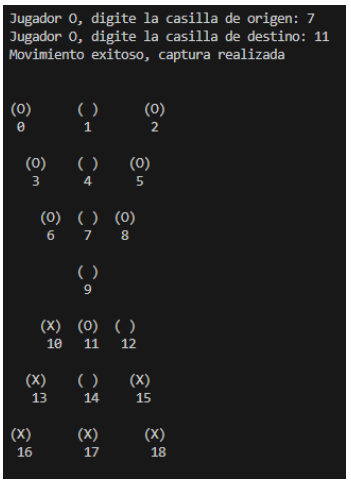

Jugador X, digite la casilla de origen: 14
Jugador X, digite la casilla de destino: 9
Movimiento exitoso, captura realizada
(0) (0) (0)
0 1 2
(0) (0) (0)
3 4 5
(0) ( ) (0)
6 7 8
(X)
9
(X) ( ) (X)
10 11 12
(X) ( ) (X)
13 14 15
(X) (X) (X)
16 17 18

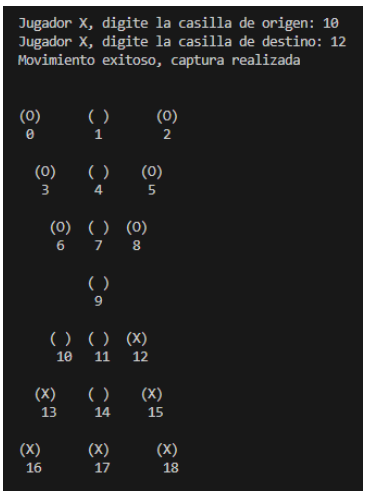

Jugador O, digite la casilla de origen: 4
Jugador O, digite la casilla de destino: 7
Movimiento exitoso
(0) (0) (0)
0 1 2
(0) ( ) (0)
3 4 5
(0) (0) (0)
6 7 8
(X)
9
(X) ( ) (X)
10 11 12
(X) ( ) (X)
13 14 15
(X) (X) (X)
16 17 18

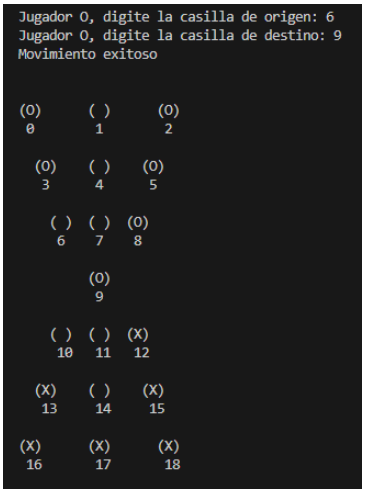

Jugador X, digite la casilla de origen: 9
Jugador X, digite la casilla de destino: 4
Movimiento exitoso, captura realizada
(0) (0) (0)
0 1 2
(0) (X) (0)
3 4 5
(0) ( ) (0)
6 7 8
( )
9
(X) ( ) (X)
10 11 12
(X) ( ) (X)
13 14 15
(X) (X) (X)
16 17 18

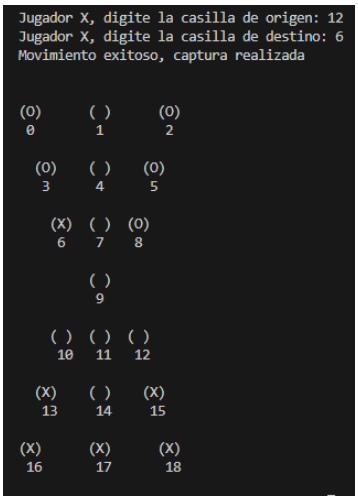

Jugador O, digite la casilla de origen: 1
Jugador O, digite la casilla de destino: 7
Movimiento exitoso, captura realizada
(0) ( ) (0)
0 1 2
(0) ( ) (0)
3 4 5
(0) (0) (0)
6 7 8
( )
9
(X) ( ) (X)
10 11 12
(X) ( ) (X)
13 14 15
(X) (X) (X)
16 17 18

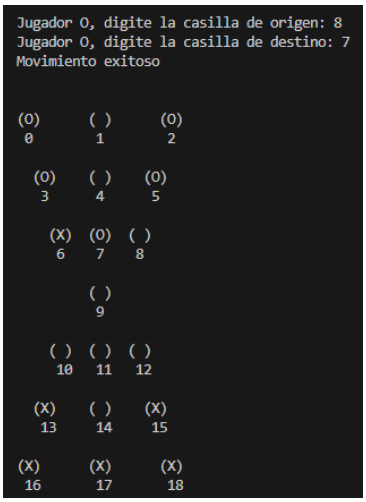

Jugador X, digite la casilla de origen: 12
Jugador X, digite la casilla de destino: 9
Movimiento exitoso
(0) ( ) (0)
0 1 2
(0) ( ) (0)
3 4 5
(0) (0) (0)
6 7 8
(X)
9
(X) ( ) ( )
10 11 12
(X) ( ) (X)
13 14 15
(X) (X) (X)
16 17 18


Jugador O, digite la casilla de origen: 7
Jugador O, digite la casilla de destino: 11
Movimiento exitoso, captura realizada
(0) ( ) (0)
0 1 2
(0) ( ) (0)
3 4 5
(0) ( ) (0)
6 7 8
( )
9
(X) (0) ( )
10 11 12
(X) ( ) (X)
13 14 15
(X) (X) (X)
16 17 18


Jugador X, digite la casilla de origen: 10
Jugador X, digite la casilla de destino: 12
Movimiento exitoso, captura realizada
(0) ( ) (0)
0 1 2
(0) ( ) (0)
3 4 5
(0) ( ) (0)
6 7 8
( )
9
( ) ( ) (X)
10 11 12
(X) ( ) (X)
13 14 15
(X) (X) (X)
16 17 18


Jugador O, digite la casilla de origen: 6
Jugador O, digite la casilla de destino: 9
Movimiento exitoso
(0) ( ) (0)
0 1 2
(0) ( ) (0)
3 4 5
( ) ( ) (0)
6 7 8
(0)
9
( ) ( ) (X)
10 11 12
(X) ( ) (X)
13 14 15
(X) (X) (X)
16 17 18


Jugador X, digite la casilla de origen: 12
Jugador X, digite la casilla de destino: 6
Movimiento exitoso, captura realizada
(0) ( ) (0)
0 1 2
(0) ( ) (0)
3 4 5
(X) ( ) (0)
6 7 8
( )
9
( ) ( ) ( )
10 11 12
(X) ( ) (X)
13 14 15
(X) (X) (X)
16 17 18


Jugador O, digite la casilla de origen: 8
Jugador O, digite la casilla de destino: 7
Movimiento exitoso
(0) ( ) (0)
0 1 2
(0) ( ) (0)
3 4 5
(X) (0) ( )
6 7 8
( )
9
( ) ( ) ( )
10 11 12
(X) ( ) (X)
13 14 15
(X) (X) (X)
16 17 18

```
Jugador X, digite la casilla de origen: 6
Jugador X, digite la casilla de destino: 8
Movimiento exitoso, captura realizada


(O)      ( )      (O)
 0        1        2

  (O)     ( )     (O)
   3       4       5

    ( )   ( )   (X)
     6     7     8

          ( )
           9

    ( )   ( )   ( )
     10    11    12

  (X)     ( )     (X)
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador O, digite la casilla de origen: 2
Jugador O, digite la casilla de destino: 1
Movimiento exitoso


(O)      (O)      ( )
 0        1        2

  (O)     ( )     (O)
   3       4       5

    ( )   ( )   (X)
     6     7     8

          ( )
           9

    ( )   ( )   ( )
     10    11    12

  (X)     ( )     (X)
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador X, digite la casilla de origen: 8
Jugador X, digite la casilla de destino: 2
Movimiento exitoso, captura realizada


(O)      (O)      (X)
 0        1        2

  (O)     ( )     ( )
   3       4       5

    ( )   ( )   ( )
     6     7     8

          ( )
           9

    ( )   ( )   ( )
     10    11    12

  (X)     ( )     (X)
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador O, digite la casilla de origen: 3
Jugador O, digite la casilla de destino: 6
Movimiento exitoso


(O)      (O)      (X)
 0        1        2

  ( )     ( )     ( )
   3       4       5

    (O)   ( )   ( )
     6     7     8

          ( )
           9

    ( )   ( )   ( )
     10    11    12

  (X)     ( )     (X)
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador X, digite la casilla de origen: 15
Jugador X, digite la casilla de destino: 12
Movimiento exitoso


(O)      (O)      (X)
 0        1        2

  ( )     ( )     ( )
   3       4       5

    (O)   ( )   ( )
     6     7     8

          ( )
           9

    ( )   ( )   (X)
     10    11    12

  (X)     ( )     ( )
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador O, digite la casilla de origen: 1
Jugador O, digite la casilla de destino: 4
Movimiento exitoso


(O)      ( )      (X)
 0        1        2

  ( )     (O)     ( )
   3       4       5

    (O)   ( )   ( )
     6     7     8

          ( )
           9

    ( )   ( )   (X)
     10    11    12

  (X)     ( )     ( )
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador X, digite la casilla de origen: 2
Jugador X, digite la casilla de destino: 1
Movimiento exitoso


(O)      (X)      ( )
 0        1        2

  ( )     (O)     ( )
   3       4       5

    (O)   ( )   ( )
     6     7     8

          ( )
           9

    ( )   ( )   (X)
     10    11    12

  (X)     ( )     ( )
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador O, digite la casilla de origen: 6
Jugador O, digite la casilla de destino: 9
Movimiento exitoso


(O)      (X)      ( )
 0        1        2

  ( )     (O)     ( )
   3       4       5

    ( )   ( )   ( )
     6     7     8

          (O)
           9

    ( )   ( )   (X)
     10    11    12

  (X)     ( )     ( )
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador X, digite la casilla de origen: 1
Jugador X, digite la casilla de destino: 7
Movimiento exitoso, captura realizada


(O)      ( )      ( )
 0        1        2

  ( )     ( )     ( )
   3       4       5

    ( )   (X)   ( )
     6     7     8

          (O)
           9

    ( )   ( )   (X)
     10    11    12

  (X)     ( )     ( )
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador O, digite la casilla de origen: 0
Jugador O, digite la casilla de destino: 3
Movimiento exitoso


( )      ( )      ( )
 0        1        2

  (O)     ( )     ( )
   3       4       5

    ( )   (X)   ( )
     6     7     8

          (O)
           9

    ( )   ( )   (X)
     10    11    12

  (X)     ( )     ( )
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador X, digite la casilla de origen: 12
Jugador X, digite la casilla de destino: 6
Movimiento exitoso, captura realizada


( )      ( )      ( )
 0        1        2

  (O)     ( )     ( )
   3       4       5

    (X)   (X)   ( )
     6     7     8

          ( )
           9

    ( )   ( )   ( )
     10    11    12

  (X)     ( )     ( )
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador O, digite la casilla de origen: 3
Jugador O, digite la casilla de destino: 4
Movimiento exitoso


( )      ( )      ( )
 0        1        2

  ( )     (O)     ( )
   3       4       5

    (X)   (X)   ( )
     6     7     8

          ( )
           9

    ( )   ( )   ( )
     10    11    12

  (X)     ( )     ( )
   13      14      15

(X)       (X)       (X)
 16        17        18
```

```
Jugador X, digite la casilla de origen: 7
Jugador X, digite la casilla de destino: 1
Movimiento exitoso, captura realizada


( )      (X)      ( )
 0        1        2

  ( )     ( )     ( )
   3       4       5

    (X)   ( )   ( )
     6     7     8

          ( )
           9

    ( )   ( )   ( )
     10    11    12

  (X)     ( )     ( )
   13      14      15

(X)       (X)       (X)
 16        17        18

¡Ganó el jugador X!
kevinc@DESKTOP-ORA3U80:/mnt/d/Projects/Working/Lau-kata-kati-C/src$
```