Instituto Tecnológico de Costa Rica

Proyecto 4

Programación Imperativa lenguaje C

Jesus Araya Chaves 2021106062

Curso: Lenguajes de Programación

Número de grupo: 20

Profesor: Maria Auxiliadora Mora

Primer semestre del 2023

Descripción del sistema

En esta sección se describirán brevemente las estructuras y funciones del programa, se describen de manera breve ya que en la documentación interna del código tienen una descripción más amplia:

1. `struct Node`: Representa un único nodo en el tablero de juego. Cada nodo tiene un número de vértice único y es propiedad de un jugador (0 si está vacío, 1 para el Jugador 1 y 2 para el Jugador 2).
2. `struct ListNode`: Representa un nodo en la lista de adyacencia. Cada nodo tiene un puntero a su Nodo correspondiente en el grafo, y un puntero al siguiente ListNode en la lista de adyacencia.
3. `struct ListNodePath`: Representa un nodo en la lista de posibles caminos. Cada nodo contiene un valor (número de vértice) y un puntero al siguiente ListNodePath.
4. `struct Graph`: Representa el tablero de juego, compuesto por todos los Nodos, sus listas de adyacencia y posibles caminos. Incluye propiedades para el tamaño de la cuadrícula y el número total de vértices (nodos).
5. `createNode(int vertex, int player)`: Crea un nuevo Nodo con un número de vértice dado y lo asigna a un jugador.
6. `createListNode(struct Node* node)`: Crea un nuevo ListNode que apunta a un Nodo dado.
7. `addEdge(struct Graph* graph, int src, int dest)`: Añade un borde no dirigido entre dos nodos, es decir, añade cada nodo a la lista de adyacencia del otro.
8. `initializeBoard(int gridSize)`: Inicializa el tablero de juego con un tamaño de cuadrícula dado. Genera dos cuadrículas conectadas por un único nodo. Cada celda de la cuadrícula es un vértice del grafo. Esta función también establece las listas de adyacencia para todos los nodos y define la colocación inicial de las piezas del jugador en el tablero.
9. `printBoard`: Imprime el tablero de juego en la consola. El tablero se imprime en dos partes para representar dos cuadrículas para dos jugadores. Cada nodo se asigna a un color según el jugador al que pertenece. Un nodo de conexión especial se imprime entre las dos cuadrículas.
10. `validPath`: Verifica si un nodo está en una ruta específica en el gráfico. Si el nodo está en el camino, devuelve verdadero, de lo contrario, devuelve falso.
11. `possiblePlays`: Imprime todos los movimientos posibles para un jugador en el juego. Los movimientos posibles se basan en la posición actual de las piezas del jugador y las posiciones de las piezas del oponente.
12. `movePiece`: Mueve una pieza de un jugador de una posición a otra en el tablero de juego. La función tiene en cuenta varios factores, como si la casilla de destino está vacía, si el destino está adyacente a la posición actual y si hay una pieza del oponente en el destino.
13. `freeMemory`: Desasigna la memoria utilizada para el tablero y todas las estructuras asociadas.

Diseño de componentes

En esta sección se describirán algunos componentes utilizados para la creación del juego:

Tablero

El tablero del juego está representado por una estructura de datos de tipo grafo compuesta por nodos conectados por aristas. Cada nodo representa una celda en la cuadrícula del juego y puede pertenecer a uno de los jugadores o estar vacío. El grafo está compuesto de dos cuadrículas, cada una de tamaño `gridSize x gridSize`, conectadas por un solo nodo central. Las conexiones entre nodos, que representan las celdas directamente alcanzables entre sí, se manejan utilizando listas de adyacencia. Además, existe una lista de posibles rutas que representan posibles trayectorias para que una pieza se desplace. Todo el grafo, incluyendo los nodos, sus listas de adyacencia, y las posibles rutas, está encapsulado dentro de una estructura `Graph`.

Manejo de jugadores

Cada jugador tiene nueve piezas, y el objetivo es agotar las piezas del oponente, cada nodo en el grafo tiene un número que dice a qué jugador pertenece el nodo.

Al principio, el jugador que iniciará el juego se selecciona mediante una entrada del usuario.

En cada turno, se le pide al jugador actual que seleccione una pieza para mover e indique el destino de la pieza. Si el movimiento es válido, se realiza y el turno cambia al otro jugador. Si el movimiento es inválido, el jugador recibe un mensaje de error y debe intentar de nuevo.

El programa continuará cambiando turnos entre los jugadores hasta que uno de ellos se quede sin piezas.

Como se mantiene el estado del juego

El estado del juego se mantiene a través de una estructura de grafo, donde cada nodo representa una casilla del tablero y almacena información sobre la pieza en esa casilla. Cuando se realiza un movimiento, la función `movePiece()` cambia el estado del nodo de origen y destino. Si una pieza es movida, su nodo correspondiente se actualiza al jugador adecuado. Si una pieza es capturada, el nodo correspondiente se vacía, lo que indica que ya no hay ninguna pieza en esa casilla. Las actualizaciones del estado se reflejan visualmente en el tablero con la función `printBoard()`, la cual itera a través de cada nodo en el grafo, imprime un marcador de posición con un color específico que indica la pieza de un jugador. Los nodos vacíos se representan con el color blanco, las piezas del jugador 1 con rojo y las del jugador 2 con azul.

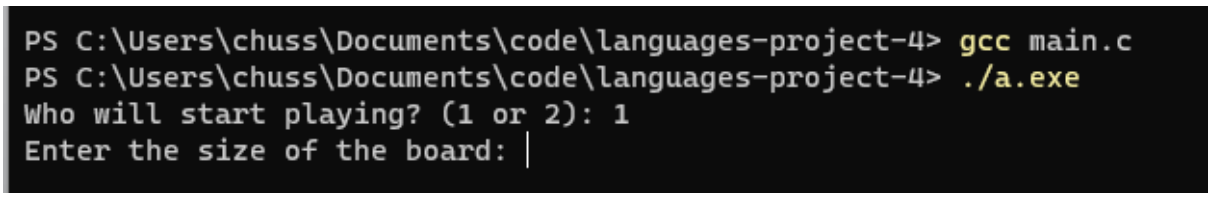
Ejecución del código

Para la ejecución del código se puede hacer de dos maneras, la primera es utilizando un IDE como CLion (programa utilizado para el desarrollo de esta tarea) o se puede utilizar el compilador de C, ejecutando:

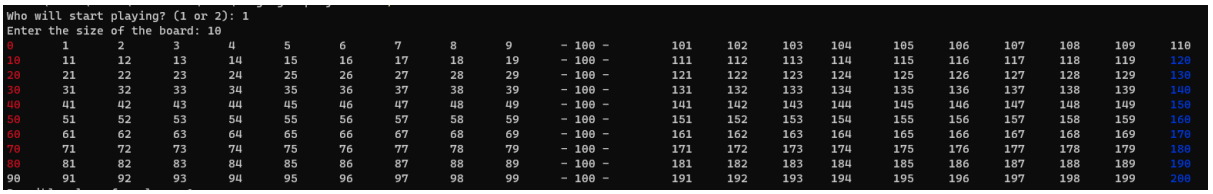
```
>gcc main.c
>./a.exe
```

Para estos pasos, se utilizará el compilador de C, ya que muestra mejores resultados gráficamente ejecutarlo desde la terminal:

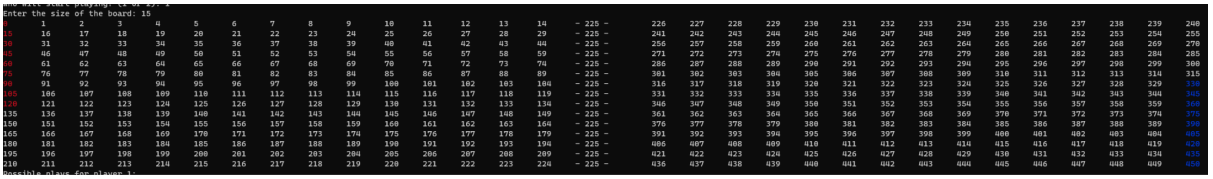
1. Compilar el programa con los comandos mencionados anteriormente.
2. Seleccionar el jugador que va a comenzar.



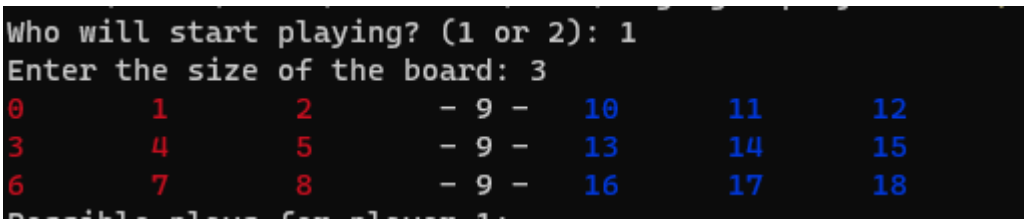
3. Ahora podemos observar que se pide el tamaño del tablero, para efectos de la documentación se creará un tablero de tamaño 3, pero de igual manera, se adjuntan tableros de más tamaño.



Tablero de tamaño 10.

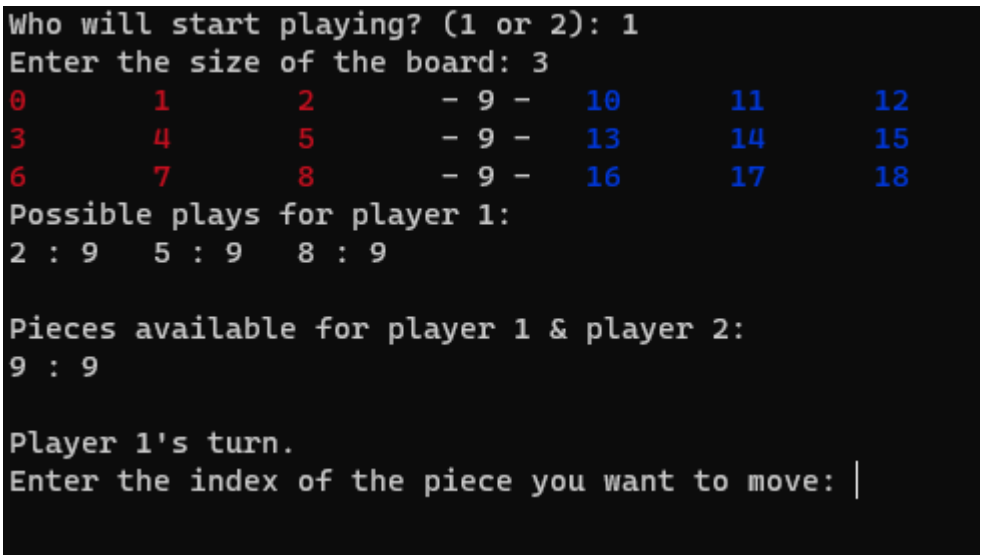


Tablero de tamaño 15.



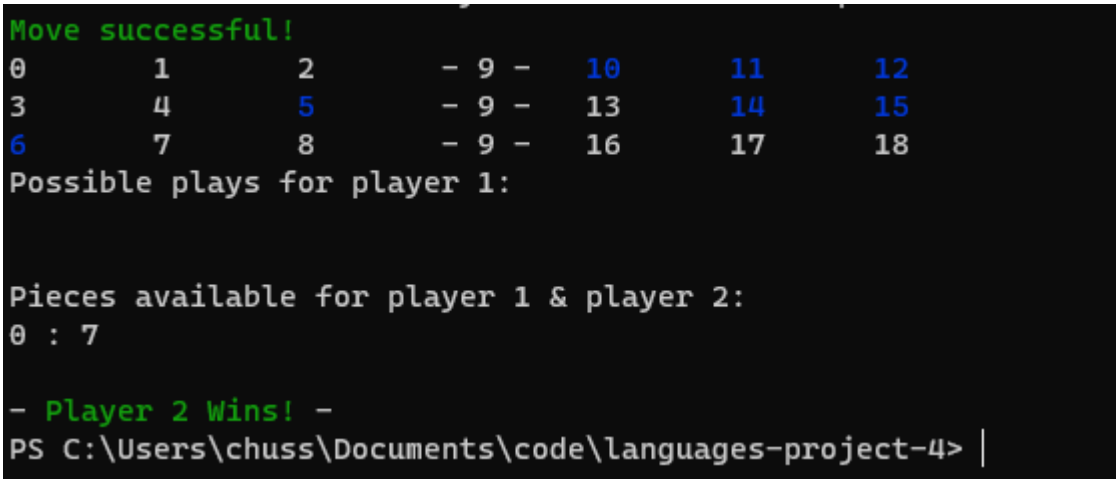
Tablero de tamaño 3.

4. Cuando se selecciona el tamaño, se despliega el tablero. A partir de este momento el juego comienza, al momento de desplegar el tablero también se muestran unos datos de ayuda, como por ejemplo, una lista de posibles jugadas para el jugador y la cantidad de fichas que cada jugador tiene.



5. Ahora, se pueden mover las fichas como el jugador desee.

6. Se puede observar que el juego tiene validaciones para que las fichas no se puedan mover a posiciones inválidas. Muestra un mensaje de error y permite al jugador volver a intentar.
7. Cuando un usuario se queda sin fichas, muestra un mensaje diciendo cual jugador ganó.


Move successful!
0 1 2 - 9 - 10 11 12
3 4 5 - 9 - 13 14 15
6 7 8 - 9 - 16 17 18
Possible plays for player 1:
Pieces available for player 1 & player 2:
0 : 7
- Player 2 Wins! -
PS C:\Users\chuss\Documents\code\languages-project-4>