**Chess Simulator**

**Análisis** *Descripción detallada del problema a resolver*

Nuestra aplicación es una aplicación lúdica que tiene como propósito simular una partida de la vida real de un juego de ajedrez de dos personas. A diferencia de un juego normal, este simulador tendrá las reglas incluidas, es decir que no te dejará mover una pieza si es que esta pieza no debe moverse de esa manera, por ejemplo, un peón no puede moverse más de una casilla.

Esta simulación de ajedrez nos permite evitar cualquier trampa ya que solo le da una oportunidad de mover una pieza a cada jugador en su respectivo turno. Además, en caso de estar aprendiendo o enseñando a jugar ajedrez, es muy útil poder regresar movimientos por lo cual este programa nos permitirá regresar los movimientos que hemos hecho, esto es de gran utilidad ya que de esta manera puedes conocer los diferentes caminos, técnicas y jugadas para adquirir práctica y mejorar en este deporte.

**Diseño** *Descripción de las Estructuras de Datos que se emplearán y la explicación de cómo estas estructuras resuelven el problema*

Para la elaboración del proyecto decidimos usar cómo estructura de datos una pila ligada.

Una pila es una estructura de datos lineal, de entradas ordenadas, sus entradas son del mismo tipo de dato y el ingreso de los datos es de tipo LIFO (Last In, First Out). Una pila ligada tiene la ventaja de que la información no se guarda en localidades consecutivas de memoria, cada elemento de una pila ligada está conformada por dos partes, la parte de la información y el enlace con el elemento anterior. Con esta pila se pueden realizar operaciones, cómo:

* Insertar (push): Se inserta un dato al final de la pila.
* Extraer (pop): Se elimina el último dato de la pila.
* Consultar (peek): Muestra el último dato de la pila.

En nuestro caso la pila ligada la usamos para poder guardar los movimientos que se hacen en el tablero, es necesario usar una pila ya que si estamos en el movimiento 9 y queremos regresar al 7 primero tendríamos que ver el último movimiento hecho, extraerlo, acomodar el tablero y repetir hasta llegar al movimiento 7.

**Analysis**

This is a ludic app that has as a purpose to simulate a real-life chess game for two people. The difference between a normal chess game and this simulator is that this program has the rules integrated, in other words it has the validations in order to respect each piece’s movements, for example, a pawn cannot move more than one square.

This chess simulator avoids any type of cheating because the program will just let us move a piece if it’s our turn. Furthermore, this app can be useful in order to teach and learn because the simulator will let us reverse our moves. This is really handy due to the facility of knowing the different ruts, techniques and moves we can do in order to get practice and be better at this sport.

**Design**

For this project elaboration we decided to implement the linked stack as a data structure. A stack is a linear data structure of ordered inputs, its inputs must be the same data type and it is structured as LIFO (Last In, First Out). A linked stack has as an advantage that its information isn’t storage in different consecutive memory localities, the information is linked with the previous data. With this stack we can do operations as:

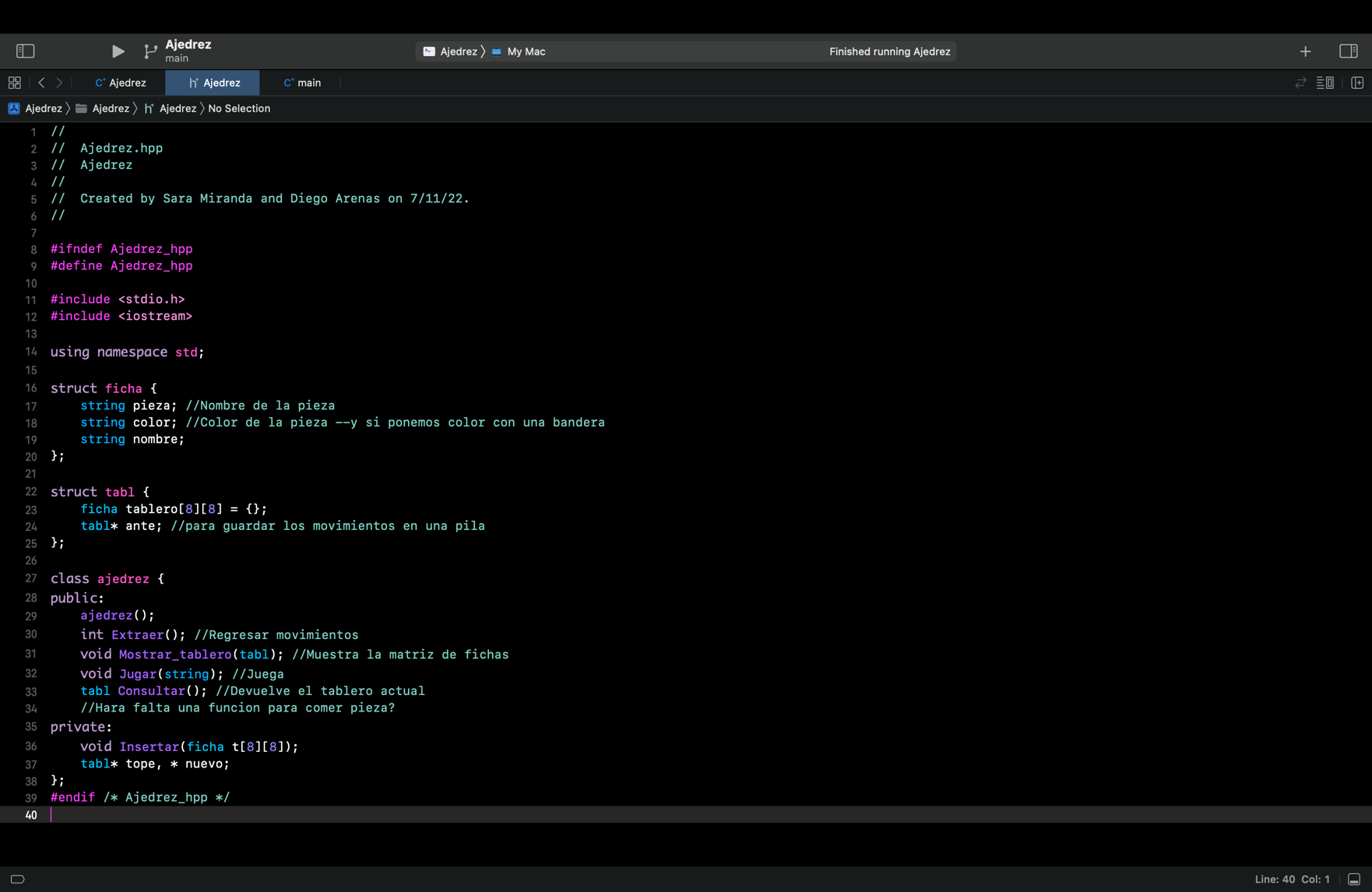
* Insert (push): We insert a data at the end of the stack
* Extract (pop): We delete the last data in the stack
* Consult (peek): It shows the last data in the stack

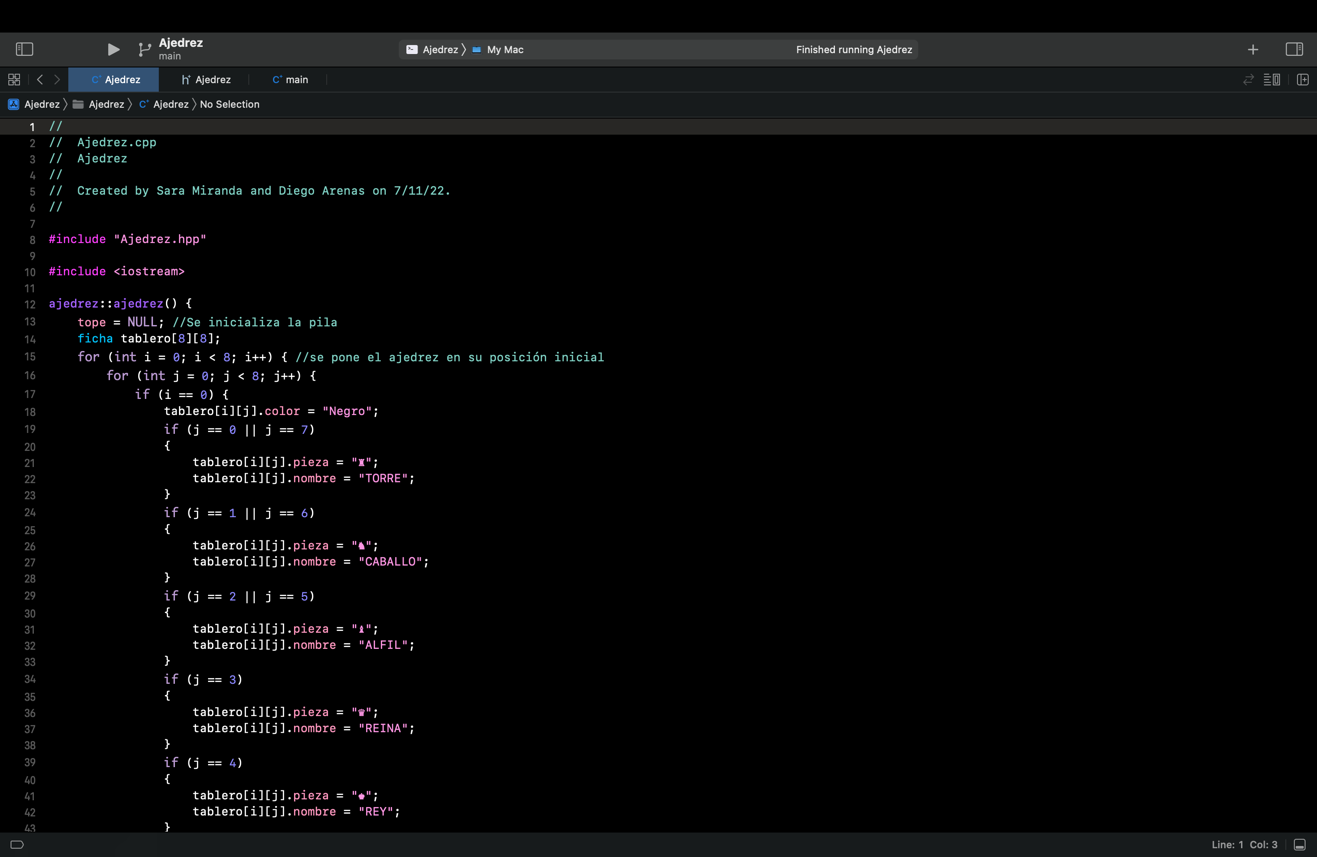
In our app we use a linked stack in order to save the moves the user had made on the board. It is necessary to use a stack because if we are in move number 9 and we want to go back to move number 7, first we should watch the last move done, extract it, order the board and repeat it until we get to the move we wanted to see.

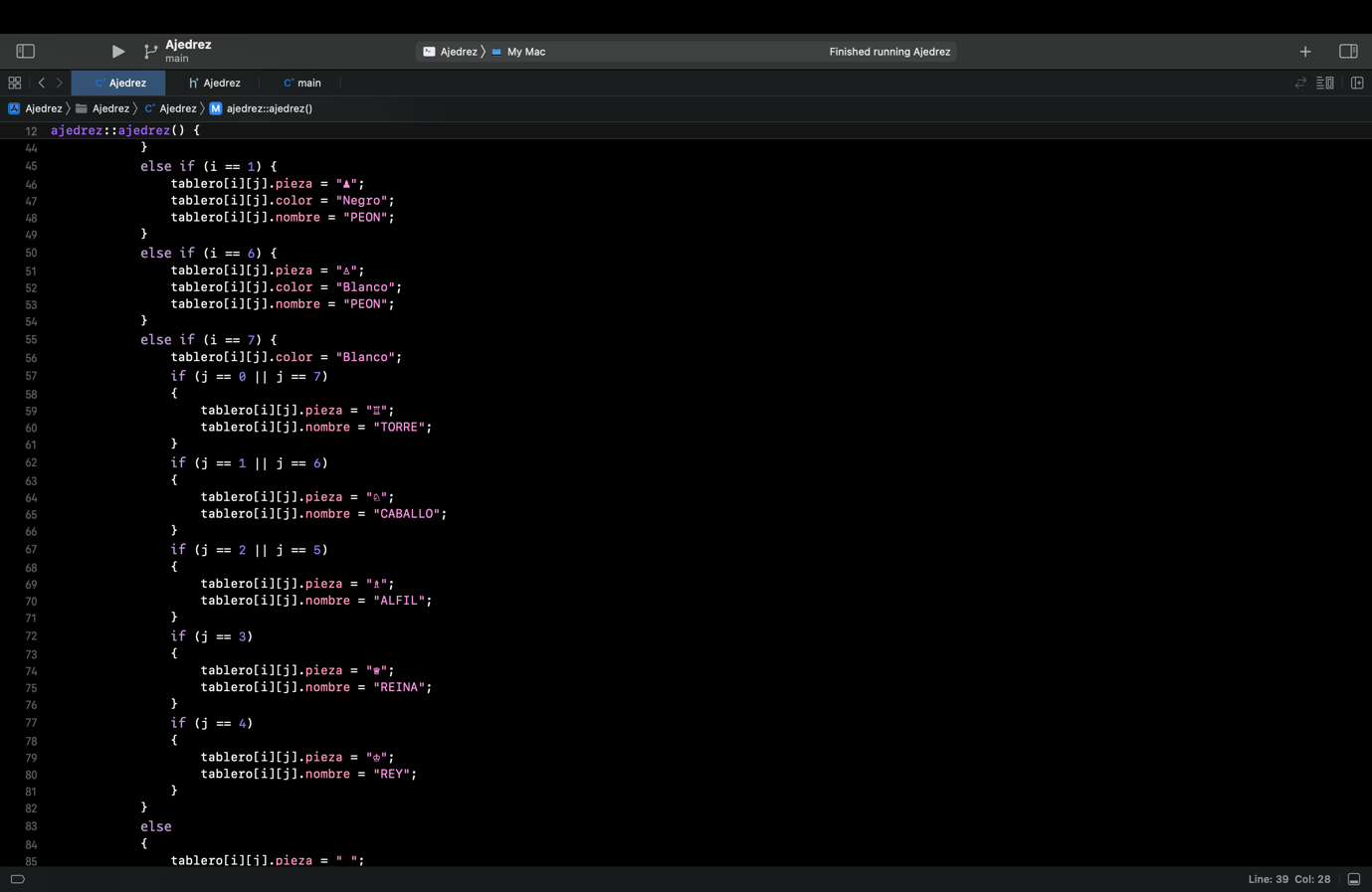
**Implementación***. Código completo de la aplicación* ***/***

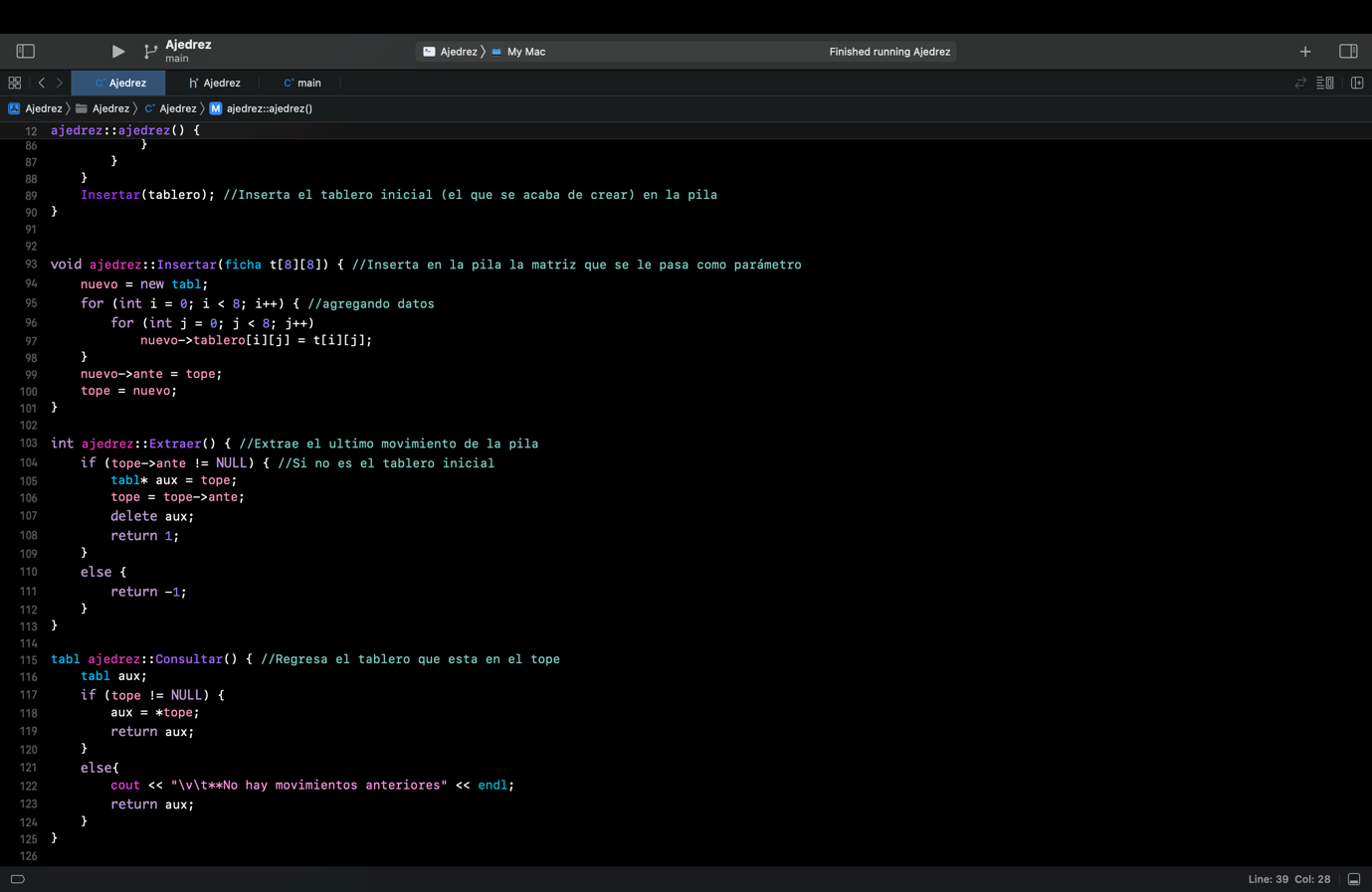
**Implementation***. Complete aplication code*

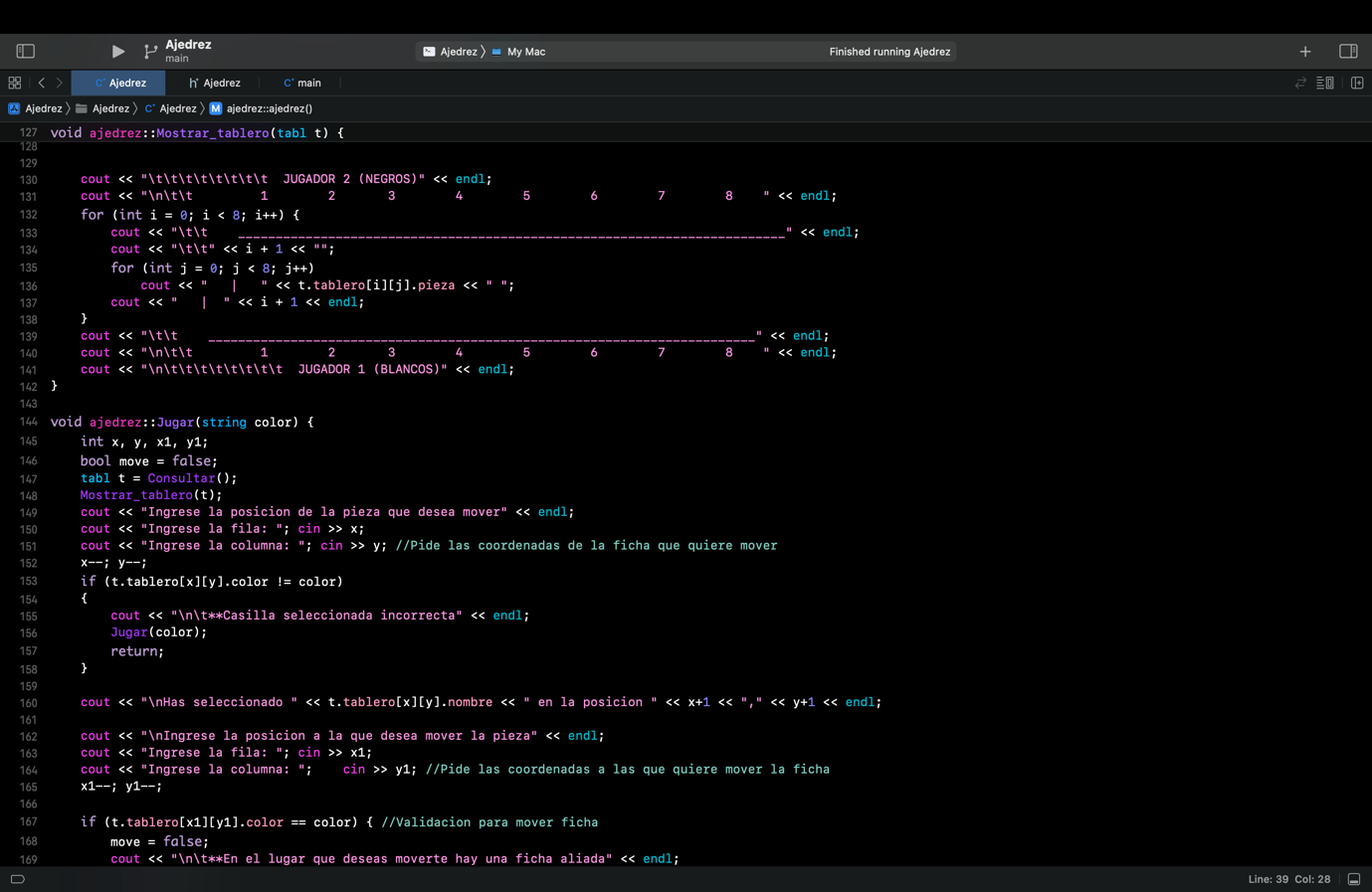
**Header *Ajedrez.h***

******

***Ajedrez.cpp***

******

******

******

***A screenshot of a computer

Description automatically generated***

***Text

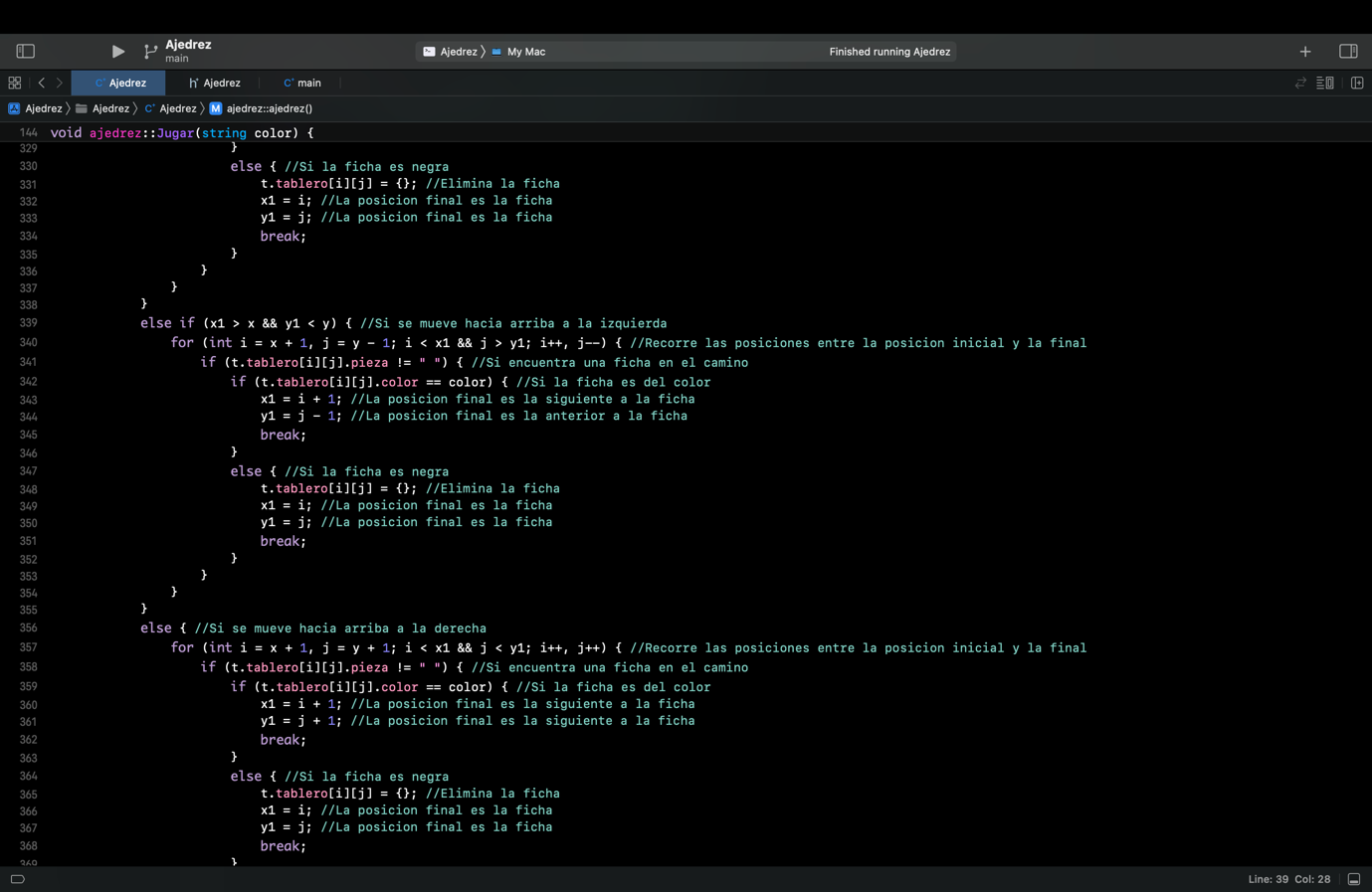
Description automatically generated***

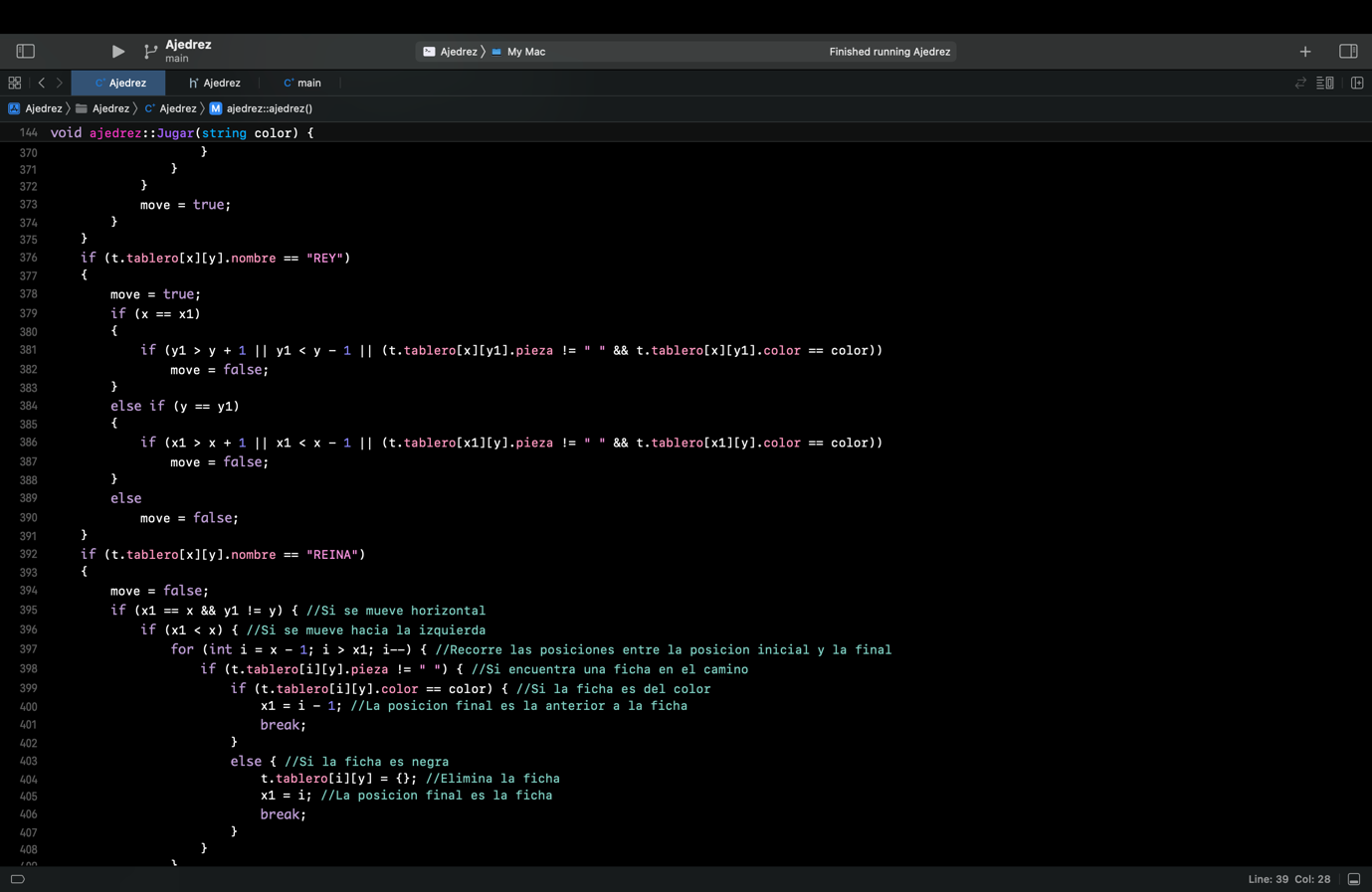
***Text

Description automatically generated***

*Text

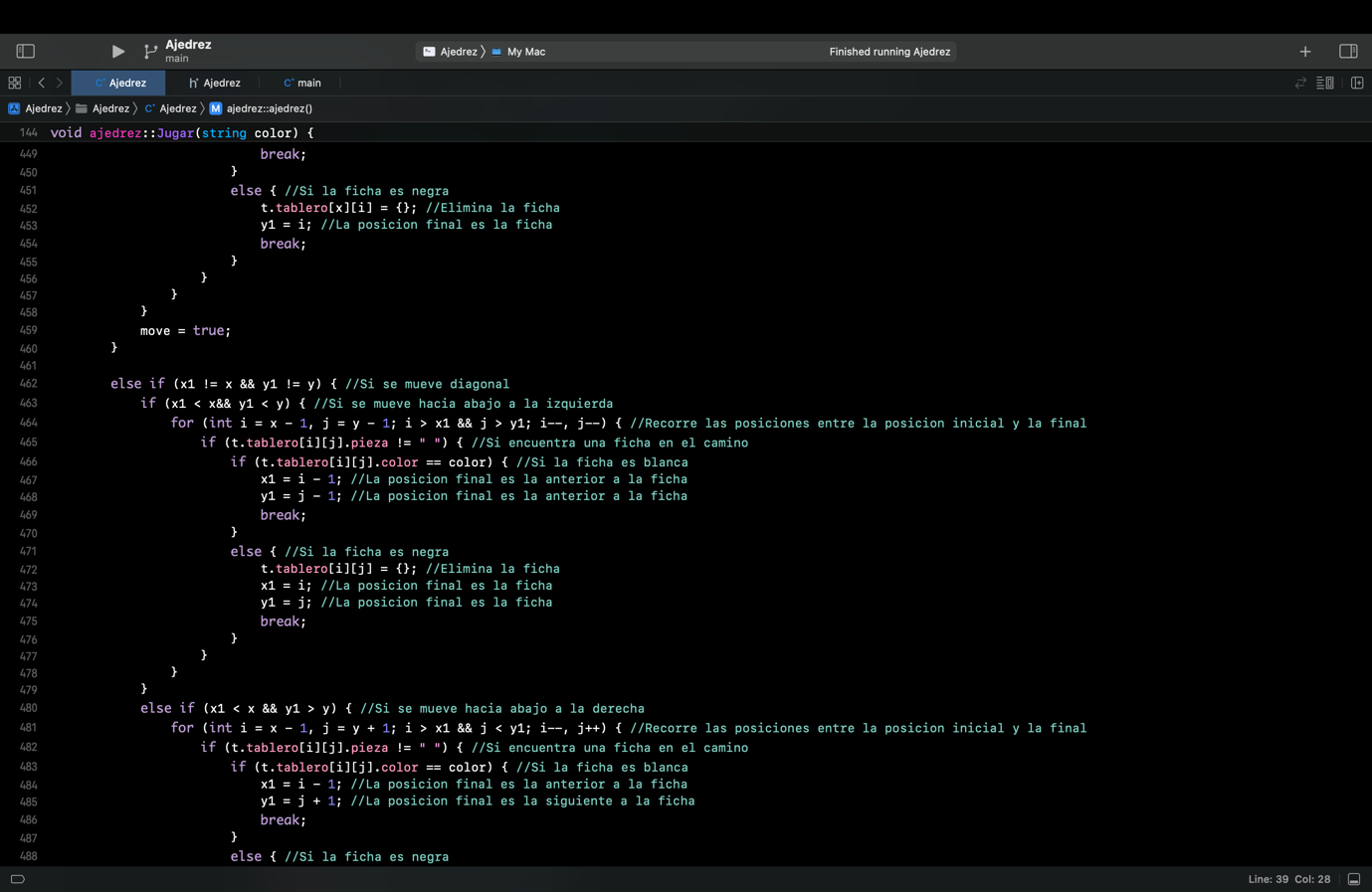
Description automatically generated*

****

**

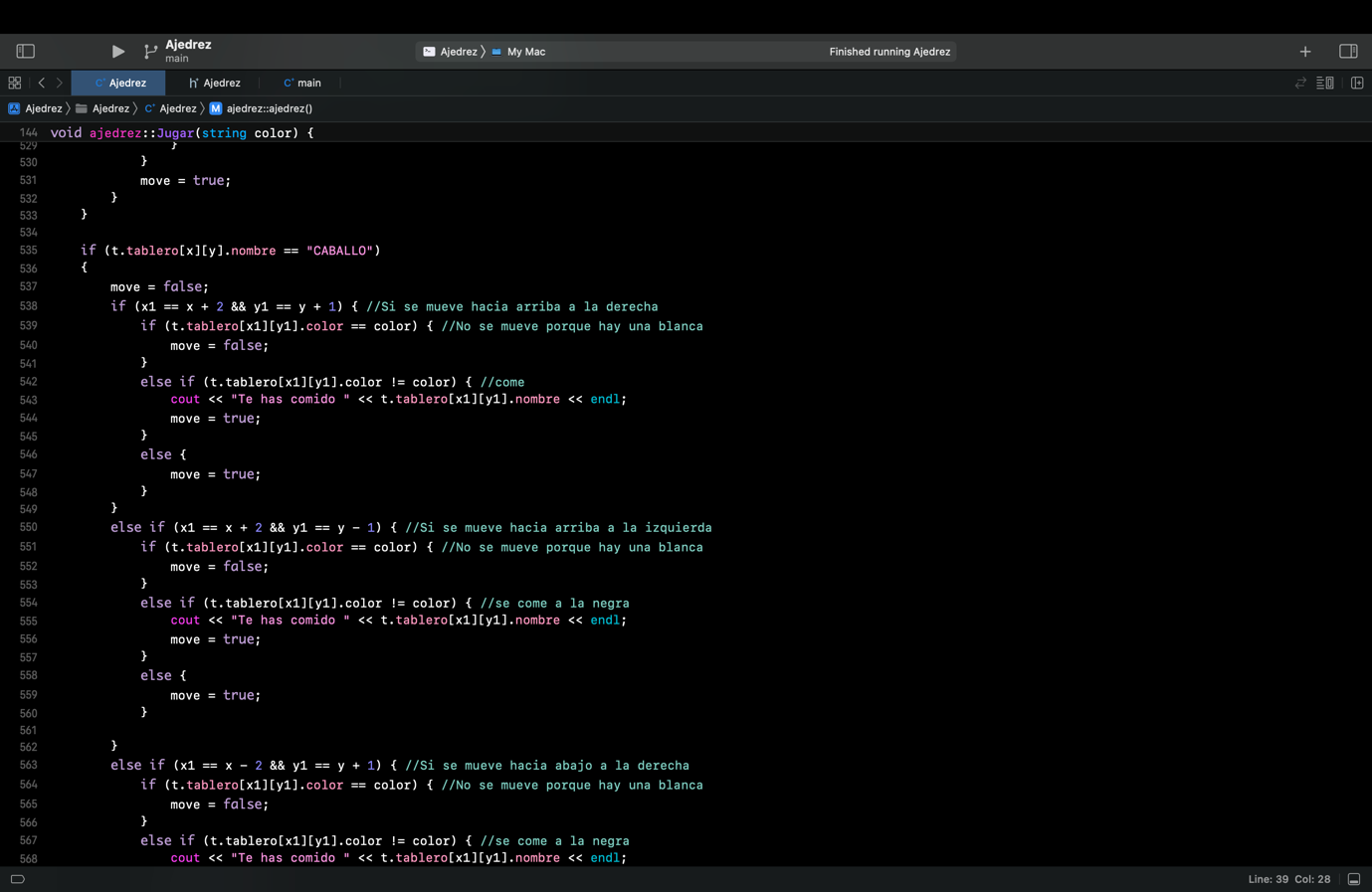
*Text

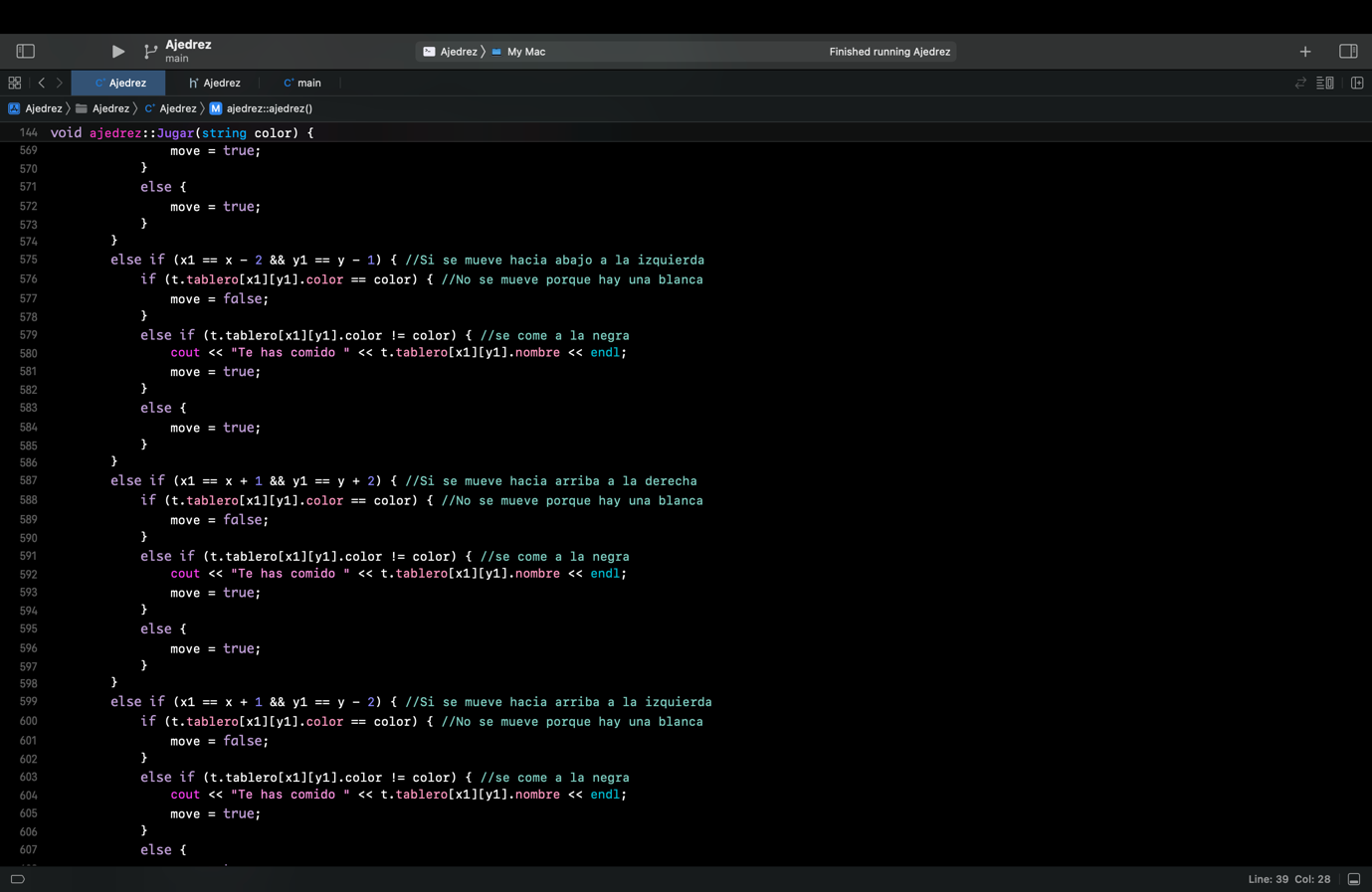
Description automatically generated*

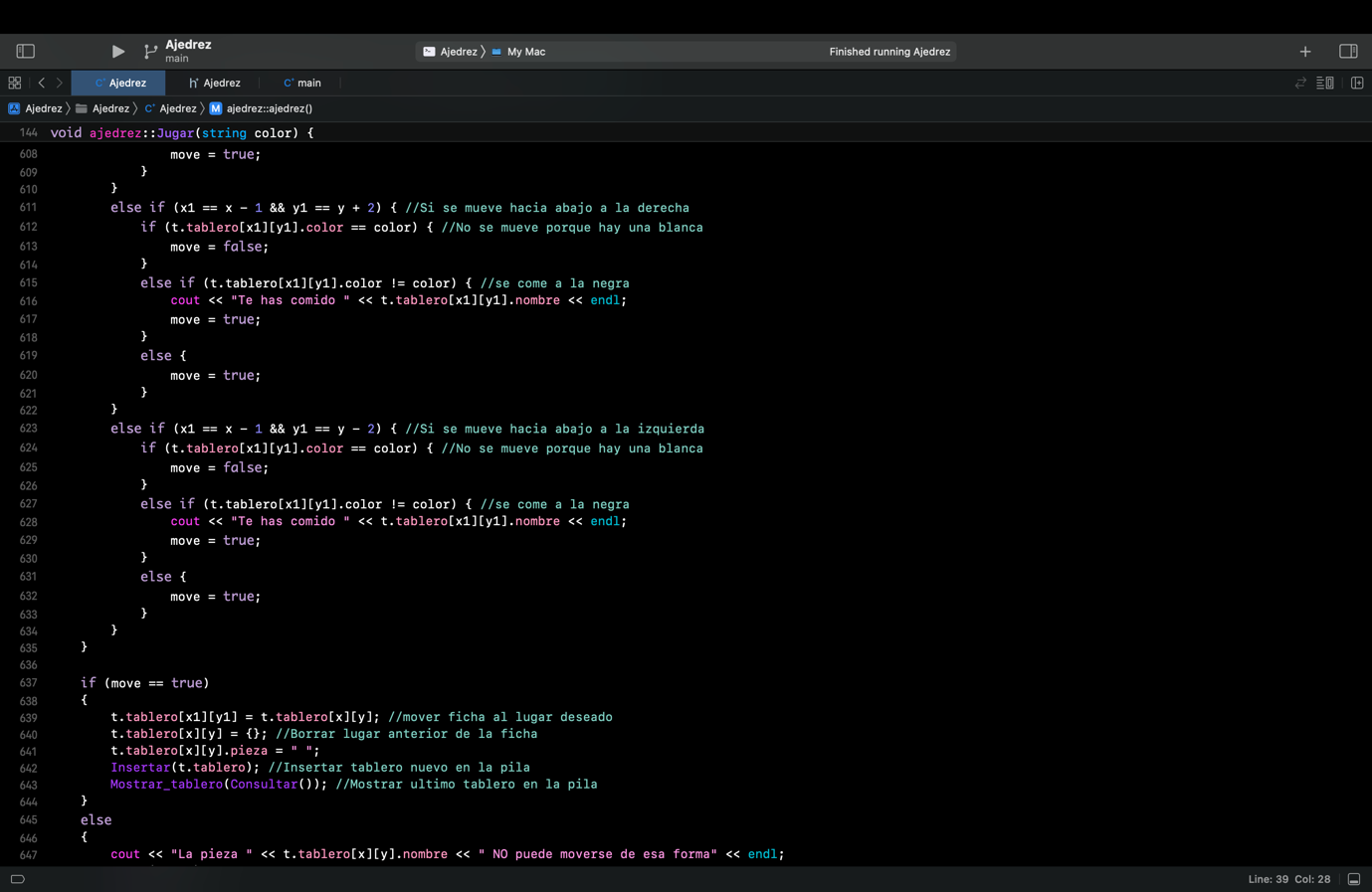
**

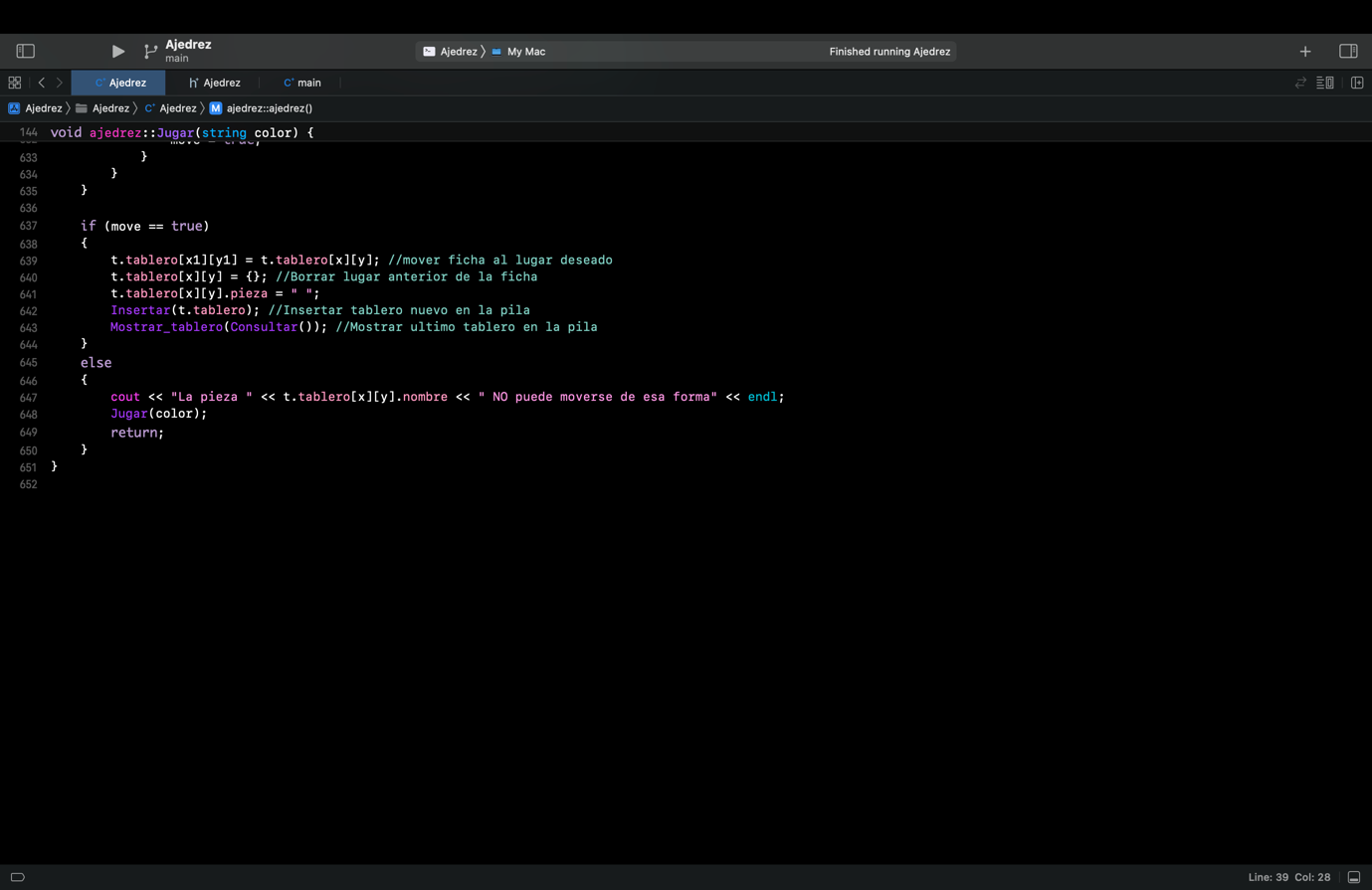
*Text

Description automatically generated*

**

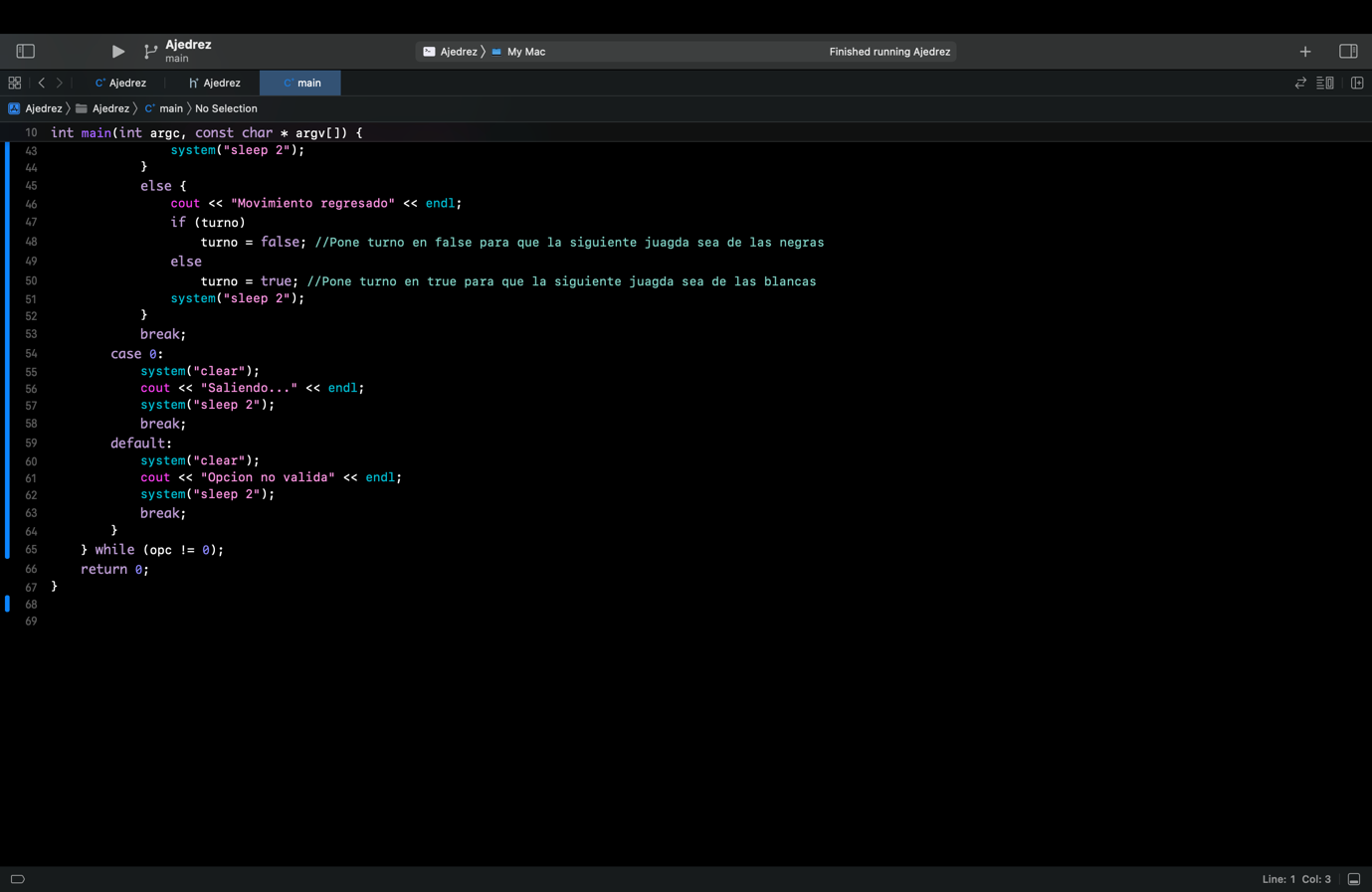
**

**

**

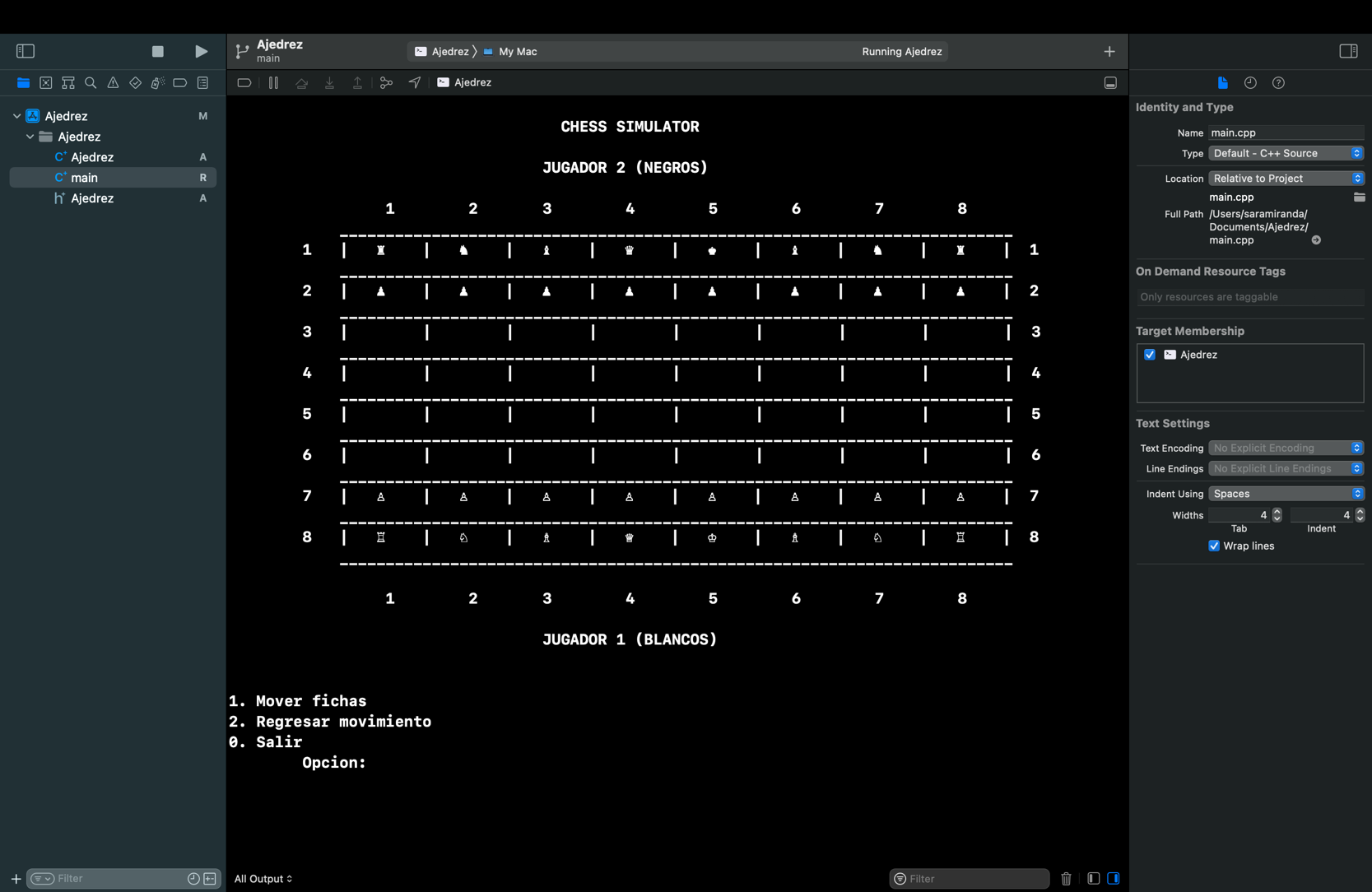
***Source.cpp***

******

******

**Prueba** *Imagen de algunas pantallas que ejemplifican la ejecución de la aplicación.*

**Proof:** *App Execution*

****

**Table

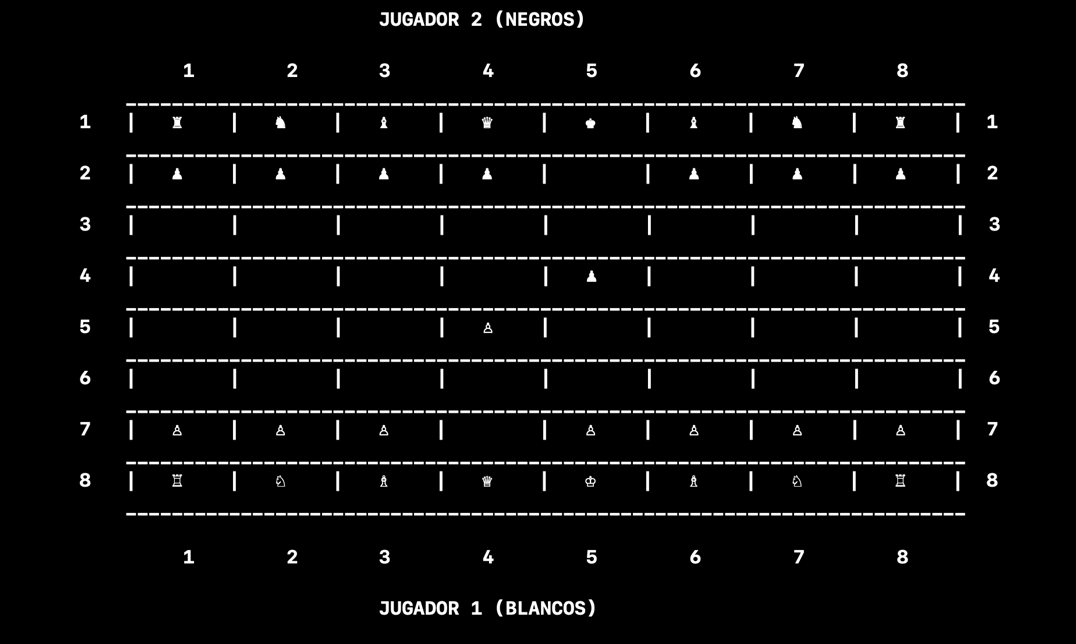
Description automatically generated**

**Calendar

Description automatically generated with medium confidence**

**A picture containing table

Description automatically generated**

****

**Calendar

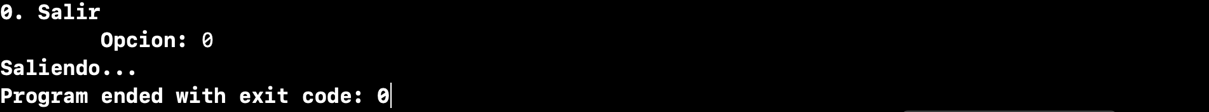
Description automatically generated**

**Calendar

Description automatically generated**

**Table

Description automatically generated with medium confidence**

****

**Guía del usuario**

**Objetivo:** Este manual tiene como finalidad dar a conocer a los pasos para utilizar el programa “Chess Simulator” para poder tener una experiencia amigable

**Requerimientos:**

* Tener cualquier compilador de C++
* Idealmente se requiere un sistema operativo Linux u iOS, de lo contrario algunas funcionalidades de la terminal no serán mostradas de la manera más eficiente. A pesar de ello el programa funcionará

**Ingreso al sistema**

* Opción 1: Con un IDE
  + Abre el programa
  + Click en *“Run Program”*
* Opción 2: Desde Terminal
  + Abre la terminal de tu dispositivo
  + Ingresa a la carpeta en donde se encuentre el programa, ejemplo

*$ cd Desktop/Ajedrez*

* + En caso de ser del sistema operativo Linux o iOS
    - Escribe los siguientes comandos

*$ g++ \*.cpp*

*$ ./a.out*

* + En caso de ser del sistema operativo Windows
    - Escribe el siguiente comando

*$ g++ -Wall -std=c++14 Source.cpp -o main.exe*

**Opciones del sistema**

***Seguir las instrucciones dentro del programa, en caso de no escribir la información correcta el programa te indicará que lo intentes de nuevo***

1. Mover pieza
   1. Ingrese la posición de la pieza que desea mover
   2. Ingrese la fila:
   3. Ingrese la columna:
   4. Ingrese la posición a la que desea mover la pieza
   5. Ingrese la fila:
   6. Ingrese la columna:
2. Regresar movimiento
3. Salir

**User’s Guide**

**Objective:** This guide has as a purpose to let the user know the steps to use the “Chess Simulator” program in order to have a friendly experience

**Requirements:**

* Download any C++ compilator
* Ideally it requires a Linux or iOS operating system, on the contrary some terminal functionalities will not be available nor be shown in the most efficient way. Nevertheless, the program will work as it is supposed to.

**System Login**

* Option 1: With a C++ IDE
  + Open the program
  + Click *“Run Program”*
* Option 2: From Terminal
  + Open your device’s terminal
  + Go into the folder which contains your program, for example

*$ cd Desktop/Ajedrez*

* + In the case it is a Linux or iOS operating system
    - Write the next commands

*$ g++ \*.cpp*

*$ ./a.out*

* + In the case it is a Windows operating system
    - Write the next commands

*$ g++ -Wall -std=c++14 Source.cpp -o main.exe*

**System Options**

***Follow up the program’s instructions. In case of not writing what is requested, the program will let you know***

1. Move piece
   1. Write the piece’s position you’d like to move
   2. Write the row:
   3. Write the column:
   4. Write the piece’s position you’d like to move to
   5. Write the row:
   6. Write the column:
2. Return movement
3. Exit