

**Facultad de Ingeniería**

**Escuela de Ingeniería de Sistemas y Computación**

**IPOO**

**Facultad de Ingeniería**

**Escuela de Ingeniería de Sistemas y Computación**

**IPOO**

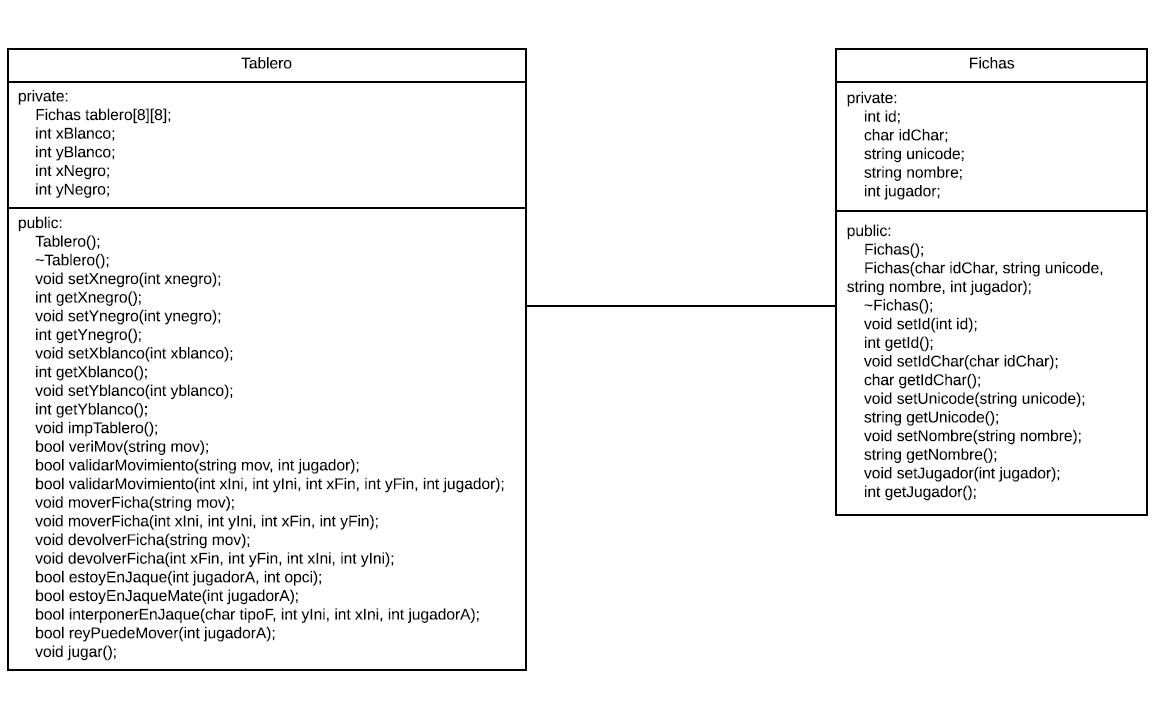
**Proyecto final**

**Manuel Diaz 1741652**

**Jeffrey Rios Muñoz 1744831**

**Juan Esteban Camargo Chacon 1741510**

Implementar por consola el juego Ajedrez, usando el paradigma orientado a objetos, en el lenguaje C++. Modele e implemente cada uno de los elementos del juego, tales como tablero, pieza, partida, y otros que consideren importante. El programa deberá permitir jugar una partida entre dos jugadores, permitir efectuar movimientos válidos y determinar cuando haya jaque mate.



**ANÁLISIS DEL PROBLEMA:**

**Objeto involucrado: Fichas**

**Atributos del Objeto:**

**privados:**

int id;

char idChar;

string unicode;

string nombre;

int jugador;

**Métodos del Objeto:**

**públicos:**

Fichas();

Fichas(char idChar, string unicode, string nombre, int jugador);

~Fichas();

void setId(int id);

int getId();

void setIdChar(char idChar);

char getIdChar();

void setUnicode(string unicode);

string getUnicode();

void setNombre(string nombre);

string getNombre();

void setJugador(int jugador);

int getJugador();

**Objeto involucrado: Tablero**

**Atributos del Objeto:**

**privados:**

Fichas tablero[8][8]; **// Tablero 8x8 del ajedrez**

int xBlanco; **// posicion x del rey blanco**

int yBlanco; **// posicion y del rey blanco**

int xNegro; **// posicion x del rey negro**

int yNegro; **// posicion y del rey negro**

**Métodos del Objeto:**

**públicos:**

Tablero();

~Tablero();

**// set and get**

void setXnegro(int xnegro);

int getXnegro();

void setYnegro(int ynegro);

int getYnegro();

void setXblanco(int xblanco);

int getXblanco();

void setYblanco(int yblanco);

int getYblanco();

**//Imprimir tablero**

void impTablero();

**//Verificaciones de movimientos**

bool veriMov(string mov); **//Verifica el formato de un movimiento (d4 a f5)**

bool validarMovimiento(string mov, int jugador**);// Valida que el movimiento sea correcto(Validacion para cada tipo de ficha)**

bool validarMovimiento(int xIni, int yIni, int xFin, int yFin, int jugador); **// Sobrecarga del metodo validarMovimiento.**

**//Modificaciones del tablero**

void moverFicha(string mov); **//Mueve una ficha de una posicion a otra sin ninguna restriccion.**

void moverFicha(int xIni, int yIni, int xFin, int yFin); **//Sobrecarga de metodo moverFicha**

void devolverFicha(string mov**); // recibiendo el mismo string que moverFicha, devuelve esta ficha a su posicion original**

**(esto para saber si el rey en una posicion queda en jaque) - Muevo el rey, verifico si estoy en jaque y devuelvo la ficha.**

void devolverFicha(int xFin, int yFin, int xIni, int yIni); **//Sobrecarga del metodo devolverFicha.**

**//Verificaciones de estado de juego**

bool estoyEnJaque(int jugadorA, int opci); **// verifica si un jugador esta en jaque. el atributo opci es para alternar los mensajes de salida.**

bool estoyEnJaqueMate(int jugadorA); **// Verifica si un jugador esta en jaque mate**

bool interponerEnJaque(char tipoF, int yIni, int xIni, int jugadorA); **// Dado un tipo de ficha enemiga, su posicion, verifico si alguna de mis fichas se la puede comer o interponerse en el camino a mi rey.**

bool reyPuedeMover(int jugadorA); **// verifico si el rey se puede mover seguro a una posicion.**

void jugar(); **// Metodo que maneja la logistica del juego**.