```
public abstract class Pieza {
	protected String color;
	protected String tipo;
	protected int fila;
	protected int columna;

	public Pieza(String color, String tipo, int fila, int columna) {
	this.color = color;
	this.tipo = tipo;
	this.fila = fila;
	this.columna = columna;
	}

	public String getColor() {
	return color;
	}

	public String getTipo() {
	return tipo;
	}

	public int getFila() {
	return fila;
	}

	public int getColumna() {
	return columna;
	}

	public void setPosicion(int fila, int columna) {
	this.fila = fila;
	this.columna = columna;
	}

	public abstract boolean puedeMover(int nuevaFila, int nuevaColumna, Pieza[][]
tablero);

	@Override
	public String toString() {
	return tipo.substring(0, 1).toUpperCase() + color.charAt(0);
	}
}
```

```
public class PiezaReina extends Pieza {

	public PiezaReina(String color, int fila, int columna) {
	super(color, "Reina", fila, columna);
	}

	@Override
	public boolean puedeMover(int nuevaFila, int nuevaColumna, Pieza[][] tablero) {
	int deltaFila = Math.abs(nuevaFila - this.fila);
	int deltaColumna = Math.abs(nuevaColumna - this.columna);

	// Movimiento de Reina: en línea recta (horizontal, vertical o diagonal)
	if (deltaFila == 0 && deltaColumna == 0) {
	return false; // No se mueve a la misma casilla
	}
	if (deltaFila == deltaColumna || nuevaFila == this.fila || nuevaColumna ==
this.columna) {
	// Verificar si hay piezas en el camino
	return !hayPiezaEnCamino(nuevaFila, nuevaColumna, tablero);
	}
	return false;
	}

	private boolean hayPiezaEnCamino(int nuevaFila, int nuevaColumna, Pieza[][]
tablero) {
	int dirFila = Integer.compare(nuevaFila, this.fila);
	int dirColumna = Integer.compare(nuevaColumna, this.columna);

	int filaActual = this.fila + dirFila;
	int columnaActual = this.columna + dirColumna;

	while (filaActual != nuevaFila || columnaActual != nuevaColumna) {
	if (tablero[filaActual][columnaActual] != null) {
		return true; // Hay una pieza en el camino
	}
	filaActual += dirFila;
	columnaActual += dirColumna;
	}
	return false;
	}
}
```

```
public class Tablero {
	private Pieza[][] tablero;

	public Tablero() {
	tablero = new Pieza[8][8];
	inicializarTablero();
	}

	public void inicializarTablero() {
	// Colocamos las piezas en sus posiciones iniciales.
	for (int i = 0; i < 8; i++) {
	tablero[1][i] = new Pieza("Blanco", "Peón", 1, i);
	tablero[6][i] = new Pieza("Negro", "Peón", 6, i);
	}

	tablero[0][0] = new Pieza("Blanco", "Torre", 0, 0);
	tablero[0][7] = new Pieza("Blanco", "Torre", 0, 7);
	tablero[7][0] = new Pieza("Negro", "Torre", 7, 0);
	tablero[7][7] = new Pieza("Negro", "Torre", 7, 7);

	tablero[0][1] = new Pieza("Blanco", "Caballo", 0, 1);
	tablero[0][6] = new Pieza("Blanco", "Caballo", 0, 6);
	tablero[7][1] = new Pieza("Negro", "Caballo", 7, 1);
	tablero[7][6] = new Pieza("Negro", "Caballo", 7, 6);

	tablero[0][2] = new Pieza("Blanco", "Alfil", 0, 2);
	tablero[0][5] = new Pieza("Blanco", "Alfil", 0, 5);
	tablero[7][2] = new Pieza("Negro", "Alfil", 7, 2);
	tablero[7][5] = new Pieza("Negro", "Alfil", 7, 5);

	tablero[0][3] = new PiezaReina("Blanco", 0, 3);
	tablero[7][3] = new PiezaReina("Negro", 7, 3);

	tablero[0][4] = new Pieza("Blanco", "Rey", 0, 4);
	tablero[7][4] = new Pieza("Negro", "Rey", 7, 4);
	}

	public Pieza getPieza(int fila, int columna) {
	return tablero[fila][columna];
	}

	public void moverPieza(int filaInicial, int columnaInicial, int filaDestino, int
columnaDestino) {
	Pieza pieza = tablero[filaInicial][columnaInicial];
	if (pieza != null && pieza.puedeMover(filaDestino, columnaDestino, tablero)) {
	tablero[filaDestino][columnaDestino] = pieza;
	tablero[filaInicial][columnaInicial] = null;
	pieza.setPosicion(filaDestino, columnaDestino);
```

```
        }
        }

        public void imprimirTablero() {
        for (int i = 0; i < 8; i++) {
        for (int j = 0; j < 8; j++) {
                if (tablero[i][j] != null) {
                System.out.print(tablero[i][j] + " ");
                } else {
                System.out.print("-- ");
                }
        }
        System.out.println();
        }
        }
}
```

```
import java.util.Scanner;

public class JuegoAjedrez {
	private Tablero tablero;
	private boolean turnoBlanco;

	public JuegoAjedrez() {
	tablero = new Tablero();
	turnoBlanco = true; // Empieza el turno de las piezas blancas
	}

	public void jugar() {
	Scanner scanner = new Scanner(System.in);
	while (true) {
	tablero.imprimirTablero();
	String colorTurno = turnoBlanco ? "Blanco" : "Negro";
	System.out.println("Turno de las piezas " + colorTurno);

	System.out.print("Introduce la fila de la pieza a mover: ");
	int filaInicial = scanner.nextInt();
	System.out.print("Introduce la columna de la pieza a mover: ");
	int columnaInicial = scanner.nextInt();

	System.out.print("Introduce la fila de destino: ");
	int filaDestino = scanner.nextInt();
	System.out.print("Introduce la columna de destino: ");
	int columnaDestino = scanner.nextInt();

	// Verificar que el movimiento es correcto y mover la pieza
	if (tablero.getPieza(filaInicial, columnaInicial) != null &&
		tablero.getPieza(filaInicial, columnaInicial).getColor().equals(colorTurno)) {
		tablero.moverPieza(filaInicial, columnaInicial, filaDestino, columnaDestino);
		turnoBlanco = !turnoBlanco; // Cambiar el turno
	} else {
		System.out.println("Movimiento no válido, intenta de nuevo.");
	}
	}
	}

	public static void main(String[] args) {
	JuegoAjedrez juego = new JuegoAjedrez();
	juego.jugar();
	}
}
```