

EMILIO - ESTRELLA - CLAUDIA

PROYECTO  
PROGRAMACIÓN  
PIEZAS AJEDRÉZ

01 - PRESENTACIÓN AJEDREZ

# 01 /

## IMPRESIÓN DE TABLERO

```
// ----- MÉTODOS ----- //
// --> MÉTODO DE SELECCIÓN/ASIGNACIÓN DE LETRA

public static void letras() {
    columna=0;
    switch (letra) {
        case "a":
            columna = 1;
            break;
        case "b":
            columna = 2;
            break;
        case "c":
            columna = 3;
            break;
        case "d":
            columna = 4;
            break;
        case "e":
            columna = 5;
            break;
        case "f":
            columna = 6;
            break;
        case "g":
            columna = 7;
            break;
        case "h":
            columna = 8;
            break;
    }
}
```

```
drez.java > ...
import java.util.Scanner;
public class Ajedrez {

    // ----- VARIABLES PUBLIC ----- //

    public static String[][] tablero = {
        {" ", " ", " ", " ", " ", " ", " ", " "},
        {" ", "a1", "b1", "c1", "d1", "e1", "f1", "g1", "h1", " "},
        {" ", "a2", "b2", "c2", "d2", "e2", "f2", "g2", "h2", " "},
        {" ", "a3", "b3", "c3", "d3", "e3", "f3", "g3", "h3", " "},
        {" ", "a4", "b4", "c4", "d4", "e4", "f4", "g4", "h4", " "},
        {" ", "a5", "b5", "c5", "d5", "e5", "f5", "g5", "h5", " "},
        {" ", "a6", "b6", "c6", "d6", "e6", "f6", "g6", "h6", " "},
        {" ", "a7", "b7", "c7", "d7", "e7", "f7", "g7", "h7", " "},
        {" ", "a8", "b8", "c8", "d8", "e8", "f8", "g8", "h8", " "}
    };
    public static String letra;
    public static String pieza;
    public static String respuesta;
    public static String planilla;
    public static int fila;
    public static int columna;
    public static int fila_copy;
}
```

1 - Creación de método para la impresión del tablero de ajedrez y declaración de variables públicas globales.

2 - Método de asignación de letra (selección de primer carácter del nombre de cada una de las piezas).

0 2 /

## E L E C C I Ó N D E P O S I C I Ó N Y A S I G N A C I Ó N P A R T E N E G R A

```
// --> MÉTODO DE ELEGIR LA POSICION EN EL TABLERO

public static void posicion() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Ahora dime su posicion inicial.");
    System.out.println("Dime una letra valida en minuscula: ");
    letra = sc.nextLine();
    System.out.println("Dime un numero valido: ");
    fila = sc.nextInt();
    letras();
}

// --> MÉTODO DE ASIGNACIÓN DE TABLERO - PARTE BLANCA

public static void tableroBlancas() {
    System.out.println("Este es tu tablero de ajedrez para las piezas blancas.");
    for (int i = tablero.length - 1; i >= 1; i--) {
        System.out.println(" ");
        for (int j = 1; j < tablero.length; j++) {
            System.out.print(tablero[i][j] + " ");
        }
    }
    System.out.println('\n');
}
```

1 - Creación del método que nos permite elegir la posición que tomará la pieza del jugador dentro del tablero.

2 - Creación del método para la selección por parte del jugador de la parte blanca del tablero.

## 03 / ELECCIÓN DE PIEZAS BLANCAS

```
// --> MÉTODO DE ELECCIÓN DE PIEZAS - BLANCAS

public static void eleccionpiezasblancas() {
    System.out.println("Elige que pieza quieres usar: P=Peón, C=Caballo, A=Alfil, T=Torre, D=Dama, R=Rey ");
    Scanner piezas = new Scanner(System.in);
    pieza = piezas.nextLine();
    if (pieza.matches("[Pp]")) {
        System.out.println("Has elegido el Peón.");
        peonblanco();
    }
    else if (pieza.matches("[Cc]")) {
        System.out.println("Has elegido el Caballo.");
        caballo();
    } else if (pieza.matches("[Aa]")) {
        System.out.println("Has elegido el Alfil.");
        alfil();
    } else if (pieza.matches("[Tt]")) {
        System.out.println("Has elegido la Torre.");
        torre();
    } else if (pieza.matches("[Dd]")) {
        System.out.println("Has elegido la Dama.");
        dama();
    } else if (pieza.matches("[Rr]")) {
        System.out.println("Has elegido el Rey.");
        rey();
    } else {
        System.out.println("No has puesto una pieza válida.");
    }
}
```

1 - Creación del método con el que conseguiremos hacer la elección de piezas dentro de nuestra parte **blanca** del tablero.

## 04 / MOVIMIENTOS DE PIEZA - PEÓN BLANCO

```
// --> MÉTODO DE MOVIMIENTOS DE PEÓN - BLANCO

public static void peonblanco() {
    char inicial = 'P';
    Scanner peon = new Scanner(System.in);
    do {
        try {
            posicion();
            planilla = inicial + tablero[fila][columna];
            if (fila <= 1 || fila >= 8) {
                System.out.println("Error el peón no puede estar en esa posición.");
            }
            else {
                if (columna < 1 || columna > 8) {
                    System.out.println("Error el peón no puede estar en esa posición.");
                }
                else if (fila < 7) {
                    System.out.println("Tu posición actual es: " + tablero[fila][columna]);
                    System.out.print("El peón blanco puede moverse a: ");
                    System.out.print(tablero[fila + 1][columna] + " ");
                    System.out.print(tablero[fila + 2][columna] + " ");
                    System.out.print('\n');
                    System.out.println("La planilla actual es: " + planilla);
                }
                if (fila == 7) {
                    System.out.println("Tu posición actual es: " + tablero[fila][columna]);
                    System.out.print("El peón blanco puede moverse a: ");
                    System.out.print(tablero[fila + 1][columna] + " ");
                    System.out.print('\n');
                    System.out.println("La planilla actual es: " + planilla);
                }
            }
        catch (Exception b) {
            System.out.println("Error el peón no puede estar en esa posición.");
        }
        System.out.print("¿Quieres volver a mover el peón? si/no: ");
        respuesta = peon.next();
        while (!respuesta.matches("si|no")) {
            System.out.print("No has puesto si o no");
            System.out.print('\n');
            System.out.print("¿Quieres volver a mover el peón? si/no: ");
            respuesta = peon.next();
        }
    }
}
```

```

}
while (respuesta.matches("si"));
if (respuesta.matches("no")) {
    Ajedrez.main(null);
}

// --> MÉTODO DE SELECCIÓN/ASIGNACIÓN DE TABLERO - PARTE NEGRA

public static void tableronegras() {
    System.out.println("Este es tu tablero de ajedrez para las piezas negras.");
    for (int i = 1; i < tablero.length; i++) {
        System.out.println(" ");
        for (int j = tablero.length - 1; j >= 1; j--) {
            System.out.print(tablero[i][j] + " ");
        }
    }
    System.out.println('\n');
}
```

1 - Creación del método con el que vamos a conseguir que nos aparezcan por pantalla, todos aquellos movimientos que va a poder realizar el peón según la posición introducida por el usuario (en este caso, sería si el usuario hubiese seleccionado moverse dentro de la parte blanca del tablero).

2 - Creación del método para la selección por parte del jugador de la parte blanca del tablero.

# 05 /

## MOVIMIENTOS DE PIEZA - PEÓN NEGRO

```
// --> MÉTODO DE MOVIMIENTOS DE PEÓN - NEGRO

public static void peonnegro() {
    char inicial = 'P';
    do {
        Scanner peon = new Scanner(System.in);
        try {
            posicion();
            planilla = inicial + tablero[fila][columna];
            if (fila <= 1 || fila >= 8 || columna < 1 || columna > 8) {
                System.out.println("Error el peón no puede estar en esa posición.");
            } else if (fila < 6) {
                System.out.println("Tu posición actual es: " + tablero[fila][columna]);
                System.out.print("El peón puede moverse a: ");
                System.out.print(tablero[fila - 1][columna] + " ");
                System.out.print('\n');
                System.out.println("La planilla actual es: " + planilla);
            } else {
                System.out.println("Tu posición actual es: " + tablero[fila][columna]);
                System.out.print("El peón negro puede moverse a: ");
                System.out.print(tablero[fila - 1][columna] + " ");
                System.out.print(tablero[fila - 2][columna] + " ");
                System.out.print('\n');
                System.out.println("La planilla actual es: " + planilla);
            }
        } catch (Exception c) {
            System.out.println("Error el peón no puede estar en esa posición.");
        }
        System.out.print("¿Quieres volver a mover el peón? si/no: ");
        respuesta = peon.next();
        while (!respuesta.matches("si|no")) {
            System.out.print("No has puesto si o no");
            System.out.print('\n');
            System.out.print("¿Quieres volver a mover el peón? si/no: ");
            respuesta = peon.next();
        }
    } while (respuesta.matches("si"));
    if (respuesta.matches("no")) {
        Ajedrez.main(null);
    }
}
```

1 - Creación del método con el que vamos a conseguir que nos aparezcan por pantalla, todos aquellos movimientos que va a poder realizar el peón según la posición introducida por el usuario (en este caso, sería si el usuario hubiese seleccionado moverse dentro de la parte negra del tablero).

# 06 /

## MOVIMIENTOS DE PIEZA - TORRE

```
// --> MÉTODO DE MOVIMIENTOS DE TORRE

public static void torre() {
    char inicial = 'T';
    Scanner torre = new Scanner(System.in);
    do {
        try {
            posicion();
            planilla = inicial + tablero[fila][columna];
            if (fila == 0 || columna == 0 || fila > 8 || columna > 8) {
                System.out.println("Error la torre no puede estar en esa posicion.");
            } else {
                System.out.println("Tu posicion actual es: " + tablero[fila][columna]);
                System.out.print("La torre puede moverse a: ");
                for (int i = fila + 1; i < tablero.length; i++) {
                    System.out.print(tablero[i][columna] + " ");
                }
                for (int j = columna + 1; j < tablero.length; j++) {
                    System.out.print(tablero[fila][j] + " ");
                }
                for (int i = fila - 1; i >= 1; i--) {
                    System.out.print(tablero[i][columna] + " ");
                }
                for (int j = columna - 1; j >= 1; j--) {
                    System.out.print(tablero[fila][j] + " ");
                }
                System.out.print('\n');
                System.out.println("Tu planilla actual es: " + planilla);
            }
        } catch (Exception d) {
            System.out.println("Error la torre no puede estar en esa posicion.");
        }
        System.out.print("¿Quieres volver a mover la torre? si/no: ");
        respuesta = torre.next();
        while (!respuesta.matches("si|no")) {
            System.out.print("No has puesto si o no");
            System.out.print('\n');
            System.out.print("¿Quieres volver a mover la torre? si/no: ");
            respuesta = torre.next();
        }
    } while (respuesta.matches("si"));
    if (respuesta.matches("no")) {
        Ajedrez.main(null);
    }
}
```

1 - Creación del método con el que conseguiremos que nos aparezcan por pantalla, todos aquellos movimientos que va ha poder realizar la torre según la posición introducida por el usuario (en el caso de la torre, sus movimientos no tendrán diferencia entre la parte negra o blanca del tablero, por eso, solo existe un único tipo de torre).

# 07 / MOVIMIENTOS DE PIEZA - CABALLO

```
// --> MÉTODO DE MOVIMIENTOS DE CABALLO

public static void caballo() {
    char inicial = 'C';
    do {
        Scanner caballo = new Scanner(System.in);
        try {
            System.out.println("Ahora dime su posicion inicial.");
            System.out.println("Dime una letra: ");
            letra = caballo.nextLine();
            System.out.println("Dime un numero: ");
            fila = caballo.nextInt();
            fila_copy = fila;
            letras();
            planilla = inicial + tablero[fila][columna];

            if (fila == 0 || columna == 0 || fila > 8 || columna > 8) {
                System.out.println("Error el caballo no puede estar en esa posicion.");
            } else {
                System.out.println("Tu posicion actual es: " + tablero[fila][columna]);
                System.out.println("El caballo puede moverse a: ");

                // ----- MOVIMIENTO 1 ---- // (x=x+1; y=y+2)
                try {
                    fila += 2;
                    columna += 1;
                    System.out.print(tablero[fila][columna] + " ");
                } catch (Exception error) {
                    System.out.print("");
                }
            }
        } catch (Exception error) {
            System.out.print("");
        }
    }
}

// ----- MOVIMIENTO 3 ---- // (x=x+2; y=y-1)
try {
    fila = fila_copy;
    fila -= 1;
    columna += 0; // Las columnas con cada movimiento tendrán un desplazamiento de 2
    System.out.print(tablero[fila][columna] + " ");
} catch (Exception error) {
    System.out.print("");
}

// ----- MOVIMIENTO 4 ---- // (x=x+1; y=y-2)
try {
    fila = fila_copy;
    fila -= 2;
    columna += -1; // Las columnas con cada movimiento tendrán un desplazamiento de 2
    System.out.print(tablero[fila][columna] + " ");
} catch (Exception error) {
    System.out.print("");
}

// ----- MOVIMIENTO 5 ---- // (x=x-1; y=y-2)
try {
    fila = fila_copy;
    fila -= 2;
    columna += -2; // Las columnas con cada movimiento tendrán un desplazamiento de 2
    System.out.print(tablero[fila][columna] + " ");
} catch (Exception error) {
    System.out.print("");
}

// ----- MOVIMIENTO 6 ---- // (x=x-2; y=y-1)
try {
    fila = fila_copy;
    fila -= 1;
    columna += -1; // Las columnas con cada movimiento tendrán un desplazamiento de 2
    System.out.print(tablero[fila][columna] + " ");
} catch (Exception error) {
    System.out.print("");
}

// ----- MOVIMIENTO 7 ---- // (x=x-2; y=y+1)
try {
    fila = fila_copy;
    fila += 1;
    columna += 0; // Las columnas con cada movimiento tendrán un desplazamiento de 2
    System.out.print(tablero[fila][columna] + " ");
} catch (Exception error) {
    System.out.print("");
}

// ----- MOVIMIENTO 8 ---- // (x=x-1; y=y+2)
try {
    fila = fila_copy;
    fila += 2;
    columna += 1; // Las columnas con cada movimiento tendrán un desplazamiento de 2
    System.out.print(tablero[fila][columna] + " ");
} catch (Exception error) {
    System.out.print("");
}

System.out.print('\n');
System.out.println("Tu planilla actual es: " + planilla);
} catch (Exception g) {
    System.out.println("Error el caballo no puede estar en esa posicion.");
}
System.out.print('\n');
System.out.print("¿Quieres volver a mover el caballo? si/no: ");
respuesta = caballo.next();
while (!respuesta.matches("si|no")) {
    System.out.print("No has puesto si o no");
    System.out.print('\n');
    System.out.print("¿Quieres volver a mover el caballo? si/no: ");
    respuesta = caballo.next();
}
while (respuesta.matches("si")) {
    if (respuesta.matches("no")) {
        Ajedrez.main(null);
    }
}
```

1 - Creación del método con el que conseguiremos que nos aparezcan por pantalla, todos aquellos movimientos que va ha poder realizar el caballo según la posición introducida por el usuario (en el caso del caballo, sus movimientos no tendrán diferencia entre la parte negra o blanca del tablero, por eso, solo existe un único tipo de caballo).

# 08 /

## MOVIMIENTOS DE PIEZA - REY

```
// --> MÉTODO DE MOVIMIENTOS DEL REY

public static void rey() {
    char inicial = 'R';
    Scanner rey = new Scanner(System.in);
    do {
        try {
            posicion();
            planilla = inicial + tablero[fila][columna];
            if (fila == 0 || columna == 0 || fila > 8 || columna > 8) {
                System.out.println("Error el rey no puede estar en esa posicion.");
            } else {
                System.out.println("Tu posicion actual es: " + tablero[fila][columna]);
                System.out.println("El rey puede moverse a: ");
                System.out.print(tablero[fila+1][columna] + " ");
                System.out.print(tablero[fila-1][columna] + " ");
                System.out.print(tablero[fila][columna+1] + " ");
                System.out.print(tablero[fila][columna-1] + " ");
                System.out.print('\n');
                System.out.println("Tu planilla actual es: " + planilla);
            }
        } catch (Exception g) {
            System.out.println("Error el rey no puede estar en esa posicion.");
        }
        System.out.print("¿Quieres volver a mover el rey? si/no: ");
        respuesta = rey.next();
        while (!respuesta.matches("si|no")) {
            System.out.print("No has puesto si o no");
            System.out.print('\n');
            System.out.print("¿Quieres volver a mover el rey? si/no: ");
            respuesta = rey.next();
        }
    }
    while (respuesta.matches("si"));
    if (respuesta.matches("no")) {
        Ajedrez.main(null);
    }
}
```

1 - Creación del método con el que conseguiremos que nos aparezcan por pantalla, todos aquellos movimientos que va ha poder realizar el rey según la posición que introduzca el usuario (en el caso del rey, sus movimientos no tendrán diferencia entre la parte negra o blanca del tablero).

E M I L I O - E S T R E L L A - C L A U D I A

# 0 9 / M O V I M I E N T O S D E P I E Z A - D A M A

```
// --> MÉTODO DE MOVIMIENTOS DE DAMA

public static void dama() {
    char inicial = 'D';
    Scanner dama = new Scanner(System.in);
    do {
        try {
            posicion();
            planilla = inicial + tablero[fila][columna];
            if (fila == 0 || columna == 0 || fila > 8 || columna > 8) {
                System.out.println("Error la dama no puede estar en esa posicion.");
            } else {
                System.out.println("Tu posicion actual es: " + tablero[fila][columna]);
                System.out.println("La dama puede moverse a: ");
                for (int i = fila + 1; i < tablero.length; i++) {
                    System.out.print(tablero[i][columna] + " ");
                }
                for (int j = columna + 1; j < tablero.length; j++) {
                    System.out.print(tablero[fila][j] + " ");
                }
                for (int i = fila - 1; i >= 1; i--) {
                    System.out.print(tablero[i][columna] + " ");
                }
                for (int j = columna - 1; j >= 1; j--) {
                    System.out.print(tablero[fila][j] + " ");
                }
            }
            try {
                for (int i = fila + 1; i < tablero.length; i++) {
                    System.out.print(tablero[i][(columna + i) - fila] + " ");
                }
            } catch (Exception error3) {
                System.out.print("");
            }
            try {
                for (int i = columna - 1; i >= 1; i--) {
                    System.out.print(tablero[(columna - i) + fila][(i)] + " ");
                }
            } catch (Exception error3) {
                System.out.print("");
            }
        }
    }
}
```

```
try {
    for (int i = fila - 1; i >= 1; i--) {
        System.out.print(tablero[i][(columna - i) + fila] + " ");
    }
} catch (Exception error3) {
    System.out.print("");
}
try {
    for (int i = columna - 1; i >= 1; i--) {
        System.out.print(tablero[(fila + i) - columna][i] + " ");
    }
} catch (Exception error3) {
    System.out.print("");
}
System.out.print('\n');
System.out.println("Tu planilla actual es: " + planilla);
}
} catch (Exception h) {
    System.out.println("Error la dama no puede estar en esa posicion.");
}
System.out.print("¿Quieres volver a mover la dama? si/no: ");
respuesta = dama.next();
while (!respuesta.matches("no|n|NO|N|si|SI|S|yes|YES|Y|SÍ|sí")) {
    System.out.print("No has puesto si o no");
    System.out.print('\n');
    System.out.print("¿Quieres volver a mover la dama? si/no: ");
    respuesta = dama.next();
}
}
while (respuesta.matches("si")) ;
if (respuesta.matches("no")) {
    Ajedrez.main(null);
}
}
```

1 - Creación del método con el que conseguiremos que nos aparezcan por pantalla, todos aquellos movimientos que va ha poder realizar la dama según la posición que haya introducido el usuario (en el caso de la dama, sus movimientos no tendrán diferencia entre la parte negra o blanca del tablero).

# 10 /

## MÉTODO PRINCIPAL MAIN

```

596 // ----- MAIN -----
597
598 public static void main (String[] args){
599     int eleccion = 0;
600     do {
601         try {
602             Scanner sc = new Scanner(System.in);
603             System.out.println("-----Bienvenido al programa de Ajedrez-----");
604             System.out.println("-----Por Favor elige una opción-----");
605             System.out.println("0. Para salir del Programa.");
606             System.out.println("1. Para elegir las piezas blancas.");
607             System.out.println("2. Para elegir las piezas negras.");
608             eleccion = sc.nextInt();
609             switch (eleccion) {
610                 case 0:
611                     System.out.println("Adios, vuelve cuando quieras.");
612                     break;
613                 case 1:
614                 {
615                     tableroblanclas();
616                     elecciónpiezasblancas();
617                 }
618                 break;
619                 case 2:
620                 {
621                     tableronegras();
622                     elecciónpiezasnegras();
623                 }
624                 break;
625             }
626         } catch (Exception a) {
627             System.out.println("Error has puesto un caracter o varios caracteres, vuelve a elegir una opcion.");
628             System.out.println("Vuelve a intentarlo.");
629             Ajedrez.main(null); //Llamamos a la funcion principal para poder preguntar de nuevo por las opciones.
630         }
631     } while (eleccion < 0 || eleccion > 2);
632 }
633 }
```

1 - En este caso, dentro de nuestro método main. Al comienzo, nos aparecerán todos los comentarios de introducción al programa (la bienvenida y la opción para seleccionar el lado del tablero con sus respectivas fichas para jugar).



FIN

EMILIO - ESTRELLA - CLAUDIA