1. Declaración de Constantes y Variables:

```
private static final int FILAS = 8;
private static final int COLUMNAS = 8;
private static final int MINAS = 10;

private char[][] tableroVisible;
private char[][] tableroReal;
private boolean gameOver;
```

- Se definen constantes para las dimensiones del tablero y la cantidad de minas.

- Se declaran matrices para representar el tablero visible y el tablero real.
- Se declara una variable booleana `gameOver` para controlar el estado del juego.

2. Constructor 'Buscaminas':

```
public Buscaminas() {

    tableroVisible = new char[FILAS][COLUMNAS];
    tableroReal = new char[FILAS][COLUMNAS];
    gameOver = false;

    // Inicializar los tableros y colocar las minas
    inicializarTablero();
    colocarMinas();
    actualizarTableroVisible();
}
```

- **Inicializa las matrices y la variable `gameOver`.**

- Llama a los métodos `inicializarTablero()`, `colocarMinas()`, y `actualizarTableroVisible()`.

3. Método 'inicializarTablero':

```
private void inicializarTablero() {

    for (int i = 0; i < FILAS; i++) {
        for (int j = 0; j < COLUMNAS; j++) {
            tableroVisible[i][j] = '-';
            tableroReal[i][j] = ' ';
        }
    }
}
```

- **Inicializa las matrices `tableroVisible` y `tableroReal` con caracteres predeterminados.**

4. Método 'colocarMinas':

```
private void colocarMinas() {

    Random rand = new Random();
    int minasColocadas = 0;

    while (minasColocadas < MINAS) {
```

```
        int fila = rand.nextInt(FILAS);
        int columna = rand.nextInt(COLUMNAS);

        if (tableroReal[fila][columna] != '*') {
            tableroReal[fila][columna] = '*';
            minasColocadas++;
        }
    }
}
```

- **Utiliza un bucle `while` para colocar minas aleatoriamente en el `tableroReal`.**

5. Método 'actualizarTableroVisible':

```
private void actualizarTableroVisible() {

    for (int i = 0; i < FILAS; i++) {
        for (int j = 0; j < COLUMNAS; j++) {
            if (tableroVisible[i][j] == '-') {
                System.out.print(" -");
            } else {
                System.out.print(" " + tableroVisible[i][j]);
            }
        }
        System.out.println();
    }
    System.out.println();
}
```

- **Imprime en la consola el estado actual del `tableroVisible` con espacios o caracteres ocultos.**

6. Método 'revelarTablero':

```
private void revelarTablero() {

    for (int i = 0; i < FILAS; i++) {
        for (int j = 0; j < COLUMNAS; j++) {
            tableroVisible[i][j] = tableroReal[i][j];
        }
    }
    actualizarTableroVisible();
}
```

- **Copia el contenido del `tableroReal` al `tableroVisible` para revelar todas las celdas en caso de derrota.**

- Llama a `actualizarTableroVisible()` para imprimir el tablero revelado.

7. Método 'jugar':

```
private void jugar() {
    Scanner scanner = new Scanner(System.in);

    while (!gameOver) {
        System.out.print("Fila: ");
        int fila = scanner.nextInt();
        System.out.print("Columna: ");
        int columna = scanner.nextInt();
```

```
        // Verificar si la posición ingresada es válida
        if (fila < 0 || fila >= FILAS || columna < 0 || columna >= COLUMNAS) {
            System.out.println("Posición no válida. Inténtalo de nuevo.");
            continue;
        }

        // Verificar si se ha encontrado una mina
        if (tableroReal[fila][columna] == '*') {
            System.out.println("¡Has perdido! Una mina explotó.");
            revelarTablero();
            gameOver = true;
        } else {
            // Contar minas alrededor y actualizar el tablero visible
            int minasAlrededor = contarMinasAlrededor(fila, columna);
            tableroVisible[fila][columna] = (char) (minasAlrededor + '0');
            actualizarTableroVisible();

            // Expandir si no hay minas alrededor
            if (minasAlrededor == 0) {
                expandir(fila, columna);
            }

            // Verificar si se ha ganado el juego
            if (esVictoria()) {
                System.out.println("¡Felicidades! Has ganado.");
                gameOver = true;
            }
        }
    }

    scanner.close();
}
```

- **Utiliza un bucle `while` para permitir que el jugador realice movimientos hasta que el juego termine.**

- Solicita al jugador ingresar fila y columna.
- Verifica si la posición es válida y si se ha encontrado una mina.
- Actualiza el `tableroVisible`, lo muestra y realiza acciones adicionales según la situación.

8. Método ‘contarMinasAlrededor’:

```
private int contarMinasAlrededor(int fila, int columna) {
    int minasAlrededor = 0;

    for (int i = -1; i <= 1; i++) {
        for (int j = -1; j <= 1; j++) {
            int nuevaFila = fila + i;
            int nuevaColumna = columna + j;

            // Verificar límites del tablero
            if (nuevaFila >= 0 && nuevaFila < FILAS && nuevaColumna >= 0 &&
nuevaColumna < COLUMNAS) {
                if (tableroReal[nuevaFila][nuevaColumna] == '*') {
                    minasAlrededor++;
                }
            }
        }
    }
```

```
    return minasAlrededor;
}
```

- **Cuenta la cantidad de minas alrededor de una celda dada.**

9. Método ‘expandir’:

```
private void expandir(int fila, int columna) {
    for (int i = -1; i <= 1; i++) {
        for (int j = -1; j <= 1; j++) {
            int nuevaFila = fila + i;
            int nuevaColumna = columna + j;

            // Verificar límites del tablero y si la celda es no revelada
            if (nuevaFila >= 0 && nuevaFila < FILAS && nuevaColumna >= 0 &&
nuevaColumna < COLUMNAS) {
                if (tableroVisible[nuevaFila][nuevaColumna] == '-') {
                    int minasAlrededor = contarMinasAlrededor(nuevaFila,
nuevaColumna);
                    tableroVisible[nuevaFila][nuevaColumna] = (char)
(minasAlrededor + '0');

                    // Expandir recursivamente si no hay minas alrededor
                    if (minasAlrededor == 0) {
                        expandir(nuevaFila, nuevaColumna);
                    }
                }
            }
        }
    }
}
```

- **Expande el tablero revelando celdas vacías adyacentes de manera recursiva.**

10. Método ‘esVictoria’:

```
private boolean esVictoria() {
    for (int i = 0; i < FILAS; i++) {
        for (int j = 0; j < COLUMNAS; j++) {
            // Verificar si hay celdas no reveladas que no contienen minas
            if (tableroVisible[i][j] == '-' && tableroReal[i][j] != '*') {
                return false;
            }
        }
    }
    return true;
}
```

- **Verifica si todas las celdas no reveladas en el `tableroVisible` que no contienen minas han sido reveladas, indicando que se ha ganado el juego.**

11. Método ‘main’:

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

- **Crea una instancia de la clase `Buscaminas` y llama al método `jugar()` para comenzar el juego.**