



Universidad Nacional Autónoma
de México

Facultad de Estudios Superiores
Aragón

Laberinto
Estructura de Datos

Prof. Hernández Cabrera Jesús

Grupo: 1306

Angeles Mermejo Octavio Emiliano

Código Array2D :

```
1 package fes.aragon.clases;
2
3 3 usages
4 public class Array2D {
5     4 usages
6     private int[][] red;
7     3 usages
8     private int filas, columnas;
9
10    1 usage
11    public Array2D(int filas, int columnas) {
12        this.filas = filas;
13        this.columnas = columnas;
14        red = new int[filas][columnas];
15    }
16
17    1 usage
18    public int get(int fila, int columna) {
19        return red[fila][columna];
20    }
```

```
21
22    public void set(int fila, int columna, int value) {
23        red[fila][columna] = value;
24    }
25
26    1 usage
27    public int getFilas() {
28        return filas;
29    }
30
31    1 usage
32    public int getColumnas() {
33        return columnas;
34    }
35
36    1 usage
37    public void print() {
38        for (int i = 0; i < filas; i++) {
39            for (int j = 0; j < columnas; j++) {
40                System.out.print(red[i][j] + " ");
41            }
42            System.out.println();
43        }
```

```
44    }
45
46    1 usage
47    public int getColumnas() {
48        return columnas;
49    }
50
51    1 usage
52    public void print() {
53        for (int i = 0; i < filas; i++) {
54            for (int j = 0; j < columnas; j++) {
55                System.out.print(red[i][j] + " ");
56            }
57            System.out.println();
58        }
59    }
60 }
```

Código Back tracking:

```
Array2D.java  Backtracking.java x  Pila.java  Laberinto.java  Main.java
1  package fes.aragon.clases;
2
3  3 usages
4  public class Backtracking {
5      10 usages
6      private Laberinto laberinto;
7      6 usages
8      private Pila pila;
9
10     // Movimientos: Izquierda, Arriba, Derecha, Abajo
11     1 usage
12     private final int[][] MOVIMIENTOS = {
13         {0, -1}, // Izquierda
14         {-1, 0}, // Arriba
15         {0, 1}, // Derecha
16         {1, 0}  // Abajo
17     };
18
19     1 usage
20     public Backtracking(Laberinto laberinto) {
21         this.laberinto = laberinto;
22     }
23 }
```

```
Array2D.java  Backtracking.java x  Pila.java  Laberinto.java  Main.java
17     pila = new Pila();
18 }
19
20 1 usage
21 public boolean resolver() {
22     int entradaFila = laberinto.getEntradaFila();
23     int entradaCol = laberinto.getEntradaCol();
24     pila.push(new int[]{entradaFila, entradaCol});
25
26     while (!pila.isEmpty()) {
27         int[] actual = pila.peek();
28         int fila = actual[0];
29         int col = actual[1];
30
31         if (fila == laberinto.getSalidaFila() && col == laberinto.getSalidaCol()) {
32             return true; // Laberinto resuelto
33         }
34
35         boolean movimientoRealizado = false;
36         for (int[] mov : MOVIMIENTOS) {
37             int nFila = fila + mov[0];
38             int nCol = col + mov[1];
39             if (nFila < 0 || nFila > laberinto.getSalidaFila() || nCol < 0 || nCol > laberinto.getSalidaCol()) {
40                 continue;
41             }
42             if (laberinto.isCeldaLibre(nFila, nCol)) {
43                 pila.push(new int[]{nFila, nCol});
44                 movimientoRealizado = true;
45             }
46         }
47         if (!movimientoRealizado) {
48             pila.pop();
49         }
50     }
51     return false;
52 }
```

```
37         int nuevaFila = fila + mov[0];
38         int nuevaCol = col + mov[1];
39
40         if (esMovimientoValido(nuevaFila, nuevaCol)) {
41             laberinto.getLaberinto().set(fila, col, value: 2); // Marcar como visitado
42             pila.push(new int[]{nuevaFila, nuevaCol});
43             movimientoRealizado = true;
44             break;
45         }
46     }
47
48     if (!movimientoRealizado) {
49         laberinto.getLaberinto().set(fila, col, value: 3); // Marcar sin más opciones
50         pila.pop(); // Retroceder
51     }
52 }
53
54 return false; // No se encontró solución
55 }
56
```

1 usage

```
52 }
53
54 return false; // No se encontró solución
55 }
56
57 private boolean esMovimientoValido(int fila, int col) {
58     if (fila < 0 || col < 0 || fila >= laberinto.getLaberinto().getFilas() || col >=
59         laberinto.getLaberinto().getColumnas()) {
60         return false; // Fuera de límites
61     }
62
63     return laberinto.getLaberinto().get(fila, col) == 0; // Verificar si es un pasillo
64 }
65 }
66
```

Código Pila:

```
1 package fes.aragon.clases;
2
3 import java.util.LinkedList;
4
5 public class Pila {
6     private LinkedList<int[]> stack;
7
8     public Pila() {
9         stack = new LinkedList<>();
10    }
11
12    public void push(int[] position) {
13        stack.push(position);
14    }
15
16    public int[] pop() {
```

```
17        return stack.pop();
18    }
19
20    public boolean isEmpty() {
21        return stack.isEmpty();
22    }
23
24    public int[] peek() {
25        return stack.peek();
26    }
27
28    public int size() {
29        return stack.size();
30    }
31 }
32
```

Código Laberinto:

```
1 package fes.aragon.clases;
2
3 public class Laberinto {
4     4 usages
5     private Array2D laberinto;
6     2 usages
7     private int entradaFila, entradaCol, salidaFila, salidaCol;
8
9     1 usage
10    public Laberinto(int filas, int columnas, int entradaFila,
11    int entradaCol, int salidaFila, int salidaCol) {
12        laberinto = new Array2D(filas, columnas);
13        this.entradaFila = entradaFila;
14        this.entradaCol = entradaCol;
15        this.salidaFila = salidaFila;
16        this.salidaCol = salidaCol;
17    }
18
19    1 usage
20    public void configurarLaberinto(int[][] config) {
```

```
17         for (int i = 0; i < config.length; i++) {
18             for (int j = 0; j < config[0].length; j++) {
19                 laberinto.set(i, j, config[i][j]);
20             }
21         }
22     }
23
24     1 usage
25     public int getEntradaFila() {
26         return entradaFila;
27     }
28
29     1 usage
30     public int getEntradaCol() {
31         return entradaCol;
32     }
33
34     1 usage
35     public int getSalidaFila() {
36         return salidaFila;
37     }
```

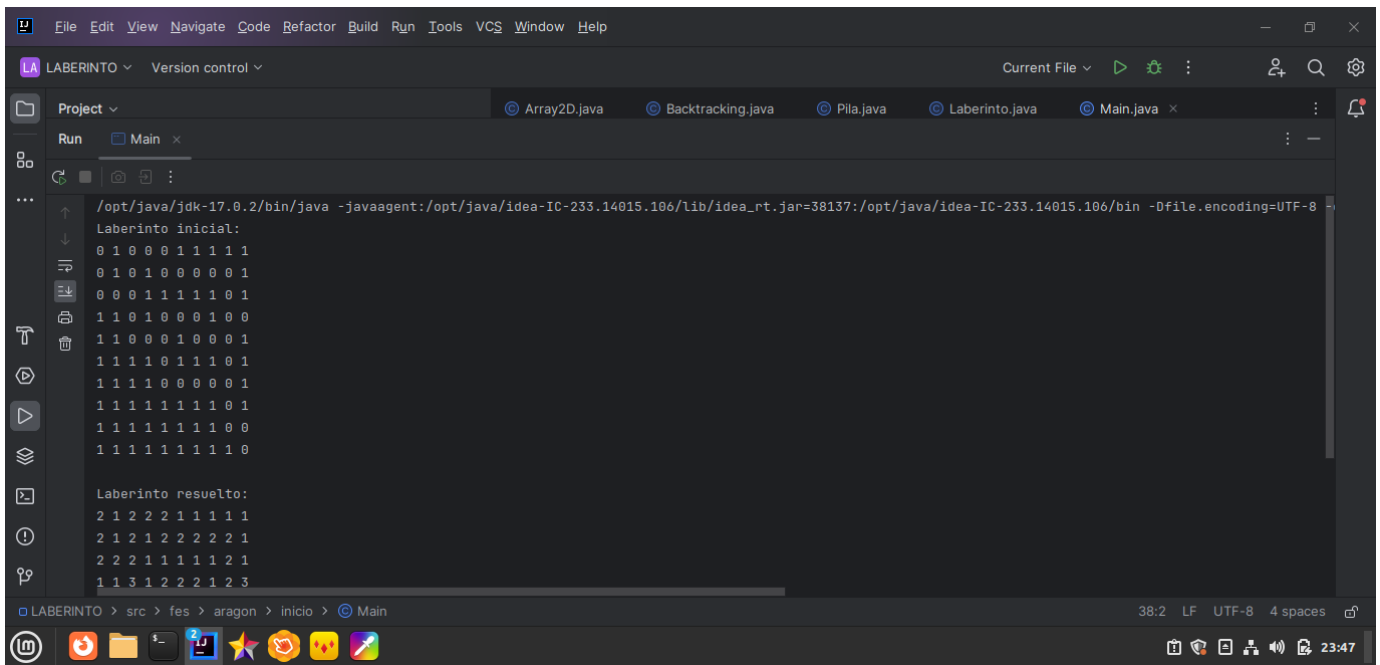
```
34     }
35
36     1 usage
37     public int getSalidaCol() {
38         return salidaCol;
39     }
40
41     5 usages
42     public Array2D getLaberinto() {
43         return laberinto;
44     }
45
46     2 usages
47     public void imprimirLaberinto() {
48         laberinto.print();
49     }
50 }
```

Código Main:

```
1 package fes.aragon.inicio;
2
3 import fes.aragon.clases.Backtracking;
4 import fes.aragon.clases.Laberinto;
5
6 public class Main {
7     public static void main(String[] args) {
8         int[][] configuracionLaberinto = {
9             {0, 1, 0, 0, 0, 1, 1, 1, 1, 1},
10            {0, 1, 0, 1, 0, 0, 0, 0, 0, 1},
11            {0, 0, 0, 1, 1, 1, 1, 1, 0, 1},
12            {1, 1, 0, 1, 0, 0, 0, 1, 0, 0},
13            {1, 1, 0, 0, 0, 1, 0, 0, 0, 1},
14            {1, 1, 1, 1, 0, 1, 1, 1, 0, 1},
15            {1, 1, 1, 1, 0, 0, 0, 0, 0, 1},
16            {1, 1, 1, 1, 1, 1, 1, 1, 0, 1},
17            {1, 1, 1, 1, 1, 1, 1, 1, 0, 0},
18            {1, 1, 1, 1, 1, 1, 1, 1, 1, 0}
19        };
20
21        Laberinto laberinto = new Laberinto
```

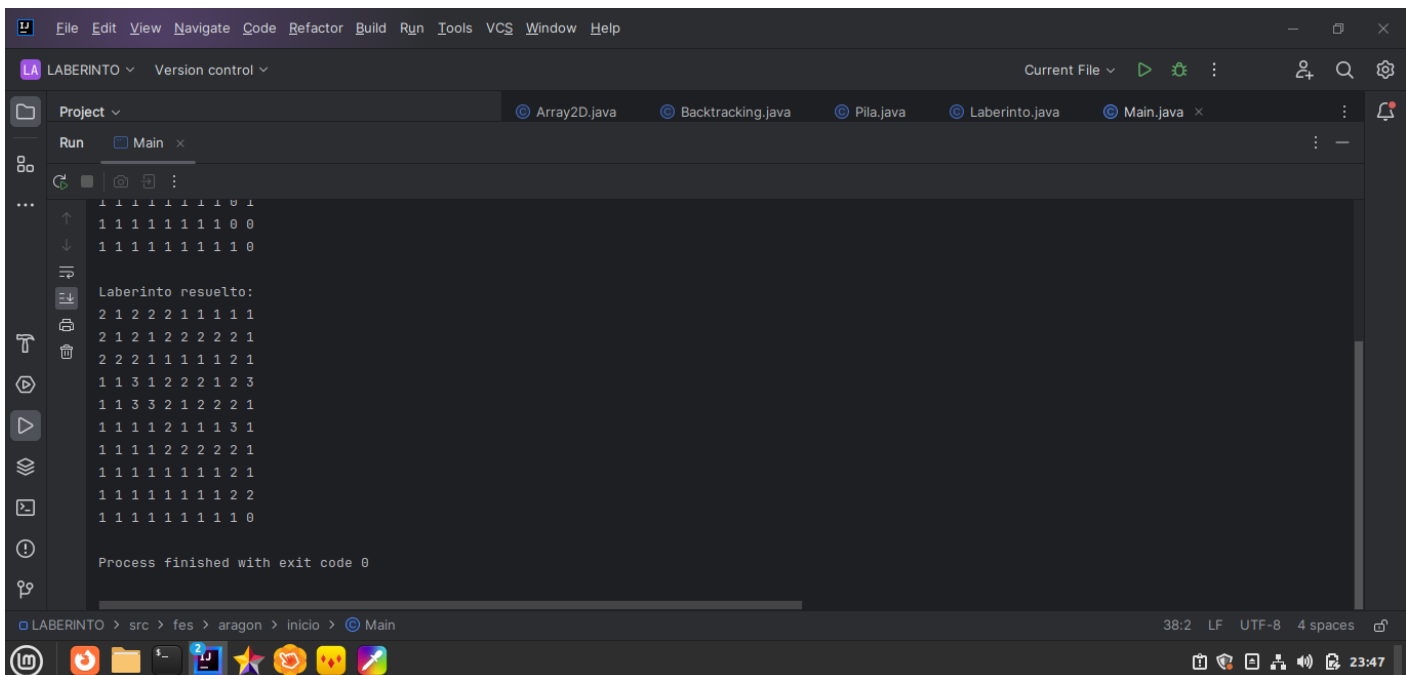
```
20
21        Laberinto laberinto = new Laberinto
22            ( filas: 10,  columnas: 10,  entradaFila: 0,  entradaCol: 0,  salidaFila: 9,  salidaCol: 9);
23        laberinto.configurarLaberinto(configuracionLaberinto);
24
25        System.out.println("Laberinto inicial:");
26        laberinto.imprimirLaberinto();
27
28        Backtracking backtracking = new Backtracking(laberinto);
29
30        if (backtracking.resolver()) {
31            System.out.println("\nLaberinto resuelto:");
32        } else {
33            System.out.println("\nNo se encontró solución:");
34        }
35
36        laberinto.imprimirLaberinto();
37    }
38 }
```

Código Resultado:



```
LABERINTO Version control
Project Run Main x
/opt/java/jdk-17.0.2/bin/java -javaagent:/opt/java/idea-IC-233.14015.106/lib/idea_rt.jar=38137:/opt/java/idea-IC-233.14015.106/bin -Dfile.encoding=UTF-8
Laberinto inicial:
0 1 0 0 0 1 1 1 1 1 1
0 1 0 1 0 0 0 0 0 1
0 0 0 1 1 1 1 1 0 1
1 1 0 1 0 0 0 1 0 0
1 1 0 0 0 1 0 0 0 1
1 1 1 1 0 1 1 1 0 1
1 1 1 1 0 0 0 0 0 1
1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 0 0
1 1 1 1 1 1 1 1 1 0

Laberinto resuelto:
2 1 2 2 2 1 1 1 1 1
2 1 2 1 2 2 2 2 2 1
2 2 2 1 1 1 1 1 2 1
1 1 3 1 2 2 2 1 2 3
```



```
LABERINTO Version control
Project Run Main x
1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 0 0
1 1 1 1 1 1 1 1 1 0

Laberinto resuelto:
2 1 2 2 2 1 1 1 1 1
2 1 2 1 2 2 2 2 2 1
2 2 2 1 1 1 1 1 2 1
1 1 3 1 2 2 2 1 2 3
1 1 3 3 2 1 2 2 2 1
1 1 1 1 2 1 1 1 3 1
1 1 1 1 2 2 2 2 2 1
1 1 1 1 1 1 1 1 2 1
1 1 1 1 1 1 1 1 2 2
1 1 1 1 1 1 1 1 1 0

Process finished with exit code 0
```