UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INFORMÁTICA

Inteligencia Artificial

Informe: Tarea 1

Nombre: Diego Joaquín Andrés Venegas Anabalón.

Matricula: 2021433473.

Profesor: Julio Erasmo Godoy Del Campo

Ayudante: Felipe Alejandro Cerda Saavedra.

Fecha de Entrega: Domingo, 29 de abril del 2024.

Introducción:

Un laberinto saltarín se define como una grilla de m por n de números saltarines, una celda inicial (en un círculo arriba), y una celda de destino (marcada “0”). En base al número saltarín de cada celda, un movimiento corresponde a moverse esa cantidad exacta de celdas ya sea de forma horizontal o vertical, en línea recta. No está permitido moverse de manera diagonal ni cambiar de dirección a medio camino. Sólo se permiten movimientos en que el número de celdas a mover no sobrepasa alguno de los límites del laberinto. El objetivo del laberinto saltarín es encontrar el camino más corto, es decir, la menor cantidad de movimientos desde la celda inicial hasta la celda de destino.

Dado que en esta tarea se nos solicitó implementar Búsqueda de Costo Uniforme (UCS) y en este caso, tenemos que todo movimiento para el agente tiene igual costo, tendremos el mismo comportamiento que un BFS, puesto que UCS siempre elige el camino con el costo más bajo. Por este mismo motivo, en el informe veremos la mención de BFS en vez de UCS, aunque cabe aclarar que no son lo mismo, solamente se hará dado que en esta instancia tendrán el mismo comportamiento y el UCS es una variante del BFS.

Implementación:

En esta implementación, dado que buscamos encontrar la mejor solución, es decir, el camino más corto entre la celda inicial y la celda de destino, se implementaron dos algoritmos de búsqueda, DFS y BFS. En el caso de DFS, se implementó de modo que se recorran todas las posibles rutas, puesto que no podemos saber en qué momento encontramos la solución óptima, al contrario que en BFS, que, al ser un recorrido en anchura, la primera solución encontrada será óptima.

Ejecutar con:

```
python main.py --filename FILENAME --m M --n N --modo MODO
```

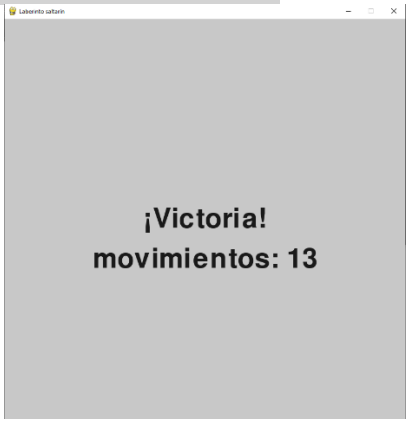
- **FILENAME:** Un string para el argumento *filename*. Es el nombre del archivo de texto que contiene el laberinto. El archivo debe estar en la carpeta base *"TareaIA-LaberintoSaltarin"*.
- **M:** Un entero para el argumento *m*. Es el número de filas del laberinto.
- **N:** Un entero para el argumento *n*. Es el número de columnas del laberinto.
- **MODO:** Un entero para el argumento *modo*. Es el modo de ejecución del programa. A continuación, los argumentos validos:
  - 0: El usuario podrá jugar con las teclas de flechas o wasd.
  - 1: Se ejecutará el laberinto, pero será resuelto por DFS.
  - 2: Se ejecutará el laberinto, pero será resuelto por BFS.
  - 3: No ejecuta el laberinto, pero compara a ambos métodos.

Si se ejecuta el programa con el argumento *filename* no se podrán utilizar los argumentos *m*, *n* y viceversa, además, si utilizamos *m* y *n*, el laberinto es creado de manera totalmente aleatoria.

Ejecución del código:

Probando con el laberinto proporcionado y ejecución manual del juego:

```
python main.py --filename entrada.txt --modo 0
```



Output en CMD de DFS y BFS con el laberinto proporcionado:

```
python main.py --filename entrada.txt --modo 1
```

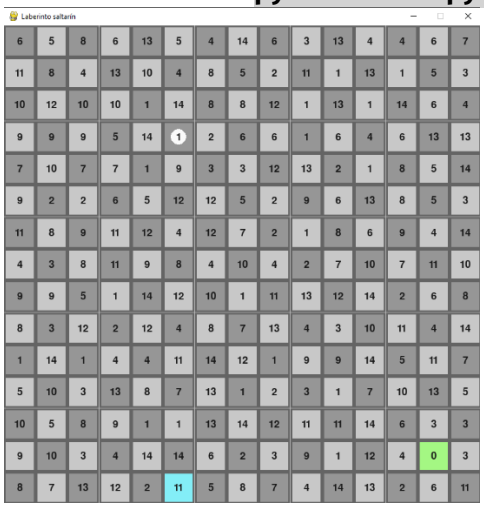
```
Búsqueda DFS
Camino mas corto: ['abajo', 'derecha', 'izquierda', 'arriba', 'abajo', 'izquierda', 'derecha', 'arriba', 'izquierda', 'izquierda', 'derecha', 'abajo', 'arriba']
Largo del camino: 13
Casillas visitadas: 50
```

```
python main.py --filename entrada.txt --modo 2
```

```
Búsqueda BFS
Camino mas corto: ['abajo', 'derecha', 'izquierda', 'arriba', 'abajo', 'izquierda', 'derecha', 'arriba', 'izquierda', 'izquierda', 'derecha', 'abajo', 'arriba']
Largo del camino: 13
Casillas visitadas: 45
```

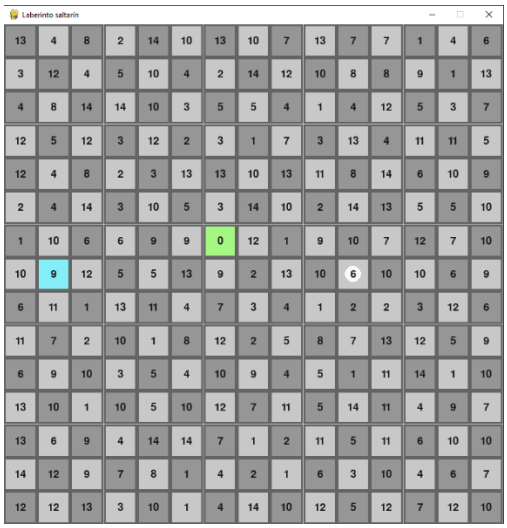
Ejecución en mapas de 15x15 en mapas aleatorizados:

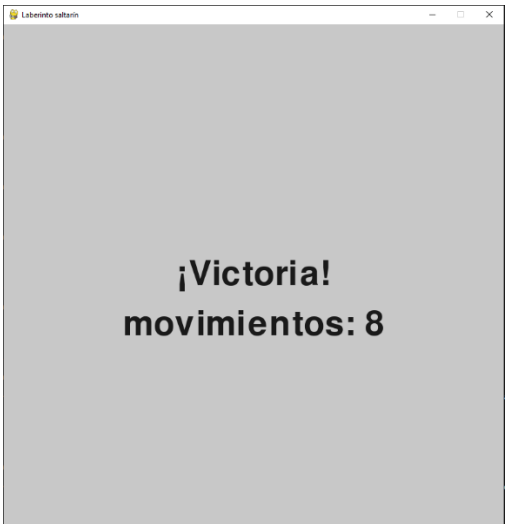
```
python main.py -m 15 -n 15 -modo 1
```




```
Búsqueda DFS
Camino mas corto: ['izquierda', 'arriba', 'abajo', 'derecha', 'izquierda', 'abajo', 'derecha', 'derecha', 'izquierda', 'abajo', 'arriba', 'abajo', 'izquierda', 'abajo', 'derecha', 'izquierda', 'arriba', 'arriba', 'derecha', 'izquierda',
'abajo', 'izquierda', 'derecha', 'izquierda', 'izquierda', 'derecha', 'abajo', 'abajo', 'derecha', 'abajo']
Largo del camino: 30
Casillas visitadas: 356
```

```
python main.py -m 15 -n 15 -modo 2
```






```
Búsqueda BFS
Camino mas corto: ['abajo', 'izquierda', 'derecha', 'arriba', 'izquierda', 'derecha', 'abajo', 'abajo']
Largo del camino: 8
Casillas visitadas: 106
```

Tres Comparaciones de DFS y BFS en mapas aleatorizados de 30x30:

```
python main.py -m 30 -n 30 -modo 3
```

```
No hay solución
```

```
Largo del camino: 218
Casillas visitadas por DFS: 1415
Casillas visitadas por BFS: 1268
```

```
Largo del camino: 15
Casillas visitadas por DFS: 1388
Casillas visitadas por BFS: 213
```

Requerimientos:

- Python 3.8.2
- Pygame 2.5.2