Mini práctica Matrices: Escapa del labertinto

Definición del problema

Desarrollar un programa que sea capaz de recorrer un laberinto siguiendo un camino de forma autónoma, desde la casilla de inicio hasta la de salida. El programa deberá funcionar para cualquier configuración de lanerinto, pero se partirá de este mapa de ejemplo:

```
......ZZ..
AA++..++++
..++....++
..++..++++
..+++++++..
```

El programa deberá, en primer lugar, encontrar el punto de inicio del camino: "AA" (podría estár en cualquier punto del mapa). A continuación, deberá encontrar la sucesión de "++" que determinan el camino hasta llegar a la casilla "ZZ". Los movimientos que permiten seguir el camino siempre serán en línea recta o haciendo giros de 90 grados (arriba/abajo, izquierda/derecha), nunca en diagonal.

La matriz a utilizar será la siguiente de 5 x 5:

```
matriz = [  ["..", "..", "..", "ZZ", ".."],
            ["AA", "++", "..", "++", "++"],
            ["..", "++", "..", "..", "++"],
            ["..", "++", "..", "++", "++"],
            ["..", "++", "++", "++", ".."]]
```

Estructura del programa

Se deberá seguir oblogatoriamente las siguientes funciones:

```
def crear_mapa() -> list:
    """
    Crea el mapa que representa el labertinto
    ... completa la doc ...
    """
    matriz = [  ["..", "..", "..", "ZZ", ".."],
                ["AA", "++", "..", "++", "++"],
                ["..", "++", "..", "..", "++"],
                ["..", "++", "..", "++", "++"],
                ["..", "++", "++", "++", ".."]]
    return matriz

def imprimir_mapa(matriz:list) -> None:
    """
    Muestra la matriz por pantalla
    ... completa la doc ...
    """
    print(...)

def encontrar_el_inicio(matriz:list) -> tuple:
    """
    Devuelve las coordenadas x, y del inicio del laberinto
    ... completa la doc ...
    """
    return (x, y)

def is_dentro_del_mapa(matriz:list, fila:int, columna:int) -> bool:
    """
    Devuelve True si la siguiente celda a evaluar está dentro del mapa
    ... completa la doc ...
    """
    return True

def evaluar_celda(matriz:int, fila:int, columna:int) -> bool:
    """
    Devuelve True si la siguiente celda a evaluar forma parte del camino
    """
    return True
```

Como se muestra en el código main, la lógica principal del juego se encontrará en la función encontrar_la_salida, que recibirá la matriz creada. Desde esta función se podrán llamar al resto de las funciones que se necesiten.

```
def encontrar_la_salida(matriz:list) -> tuple:
```

El programa principal tendrá la estructura siguiente:

```
if __name__ == "__main__":
    matriz = crear_mapa()
    print("Laberinto a resolver:\n")
    imprimir_mapa(matriz)
    celda_final = encontrar_la_salida(matriz)
    print(f"La salida del mapa está en la posición {celda_final}")
`
```

Salida del programa

```
Laberinto a resolver:

......ZZ..
AA++..++++
..++....++
..++..++++
..+++++++..

Paso:  1 0
Paso:  1 1
Paso:  2 1
Paso:  3 1
Paso:  4 1
Paso:  4 2
Paso:  4 3
Paso:  3 3
Paso:  3 4
Paso:  2 4
Paso:  1 4
Paso:  1 3
La salida está en (0, 3)
```