Instituto Politécnico Nacional
Escuela Superior de Cómputo

Paradigmas de Programación

Actividad: Ejercicio 9

Profesor: García Floriano Andrés

Fecha: 22/05/2024

Alumno:

Pacheco Refugio Alan Ivan

Grupo: 3CV1

```python
class Robot:
    def __init__(self, rejilla):
        self.rejilla = rejilla
        self.filas = len(rejilla)
        self.columnas = len(rejilla[0])
        self.ruta = []

```

Se inicializa la clase Robot y los atributos de rejilla, filas, columnas y la ruta en donde haremos uso para el movimiento del robot

```python
    def es_movimiento_valido(self, x, y, visitado):
        return (0 <= x < self.filas and 0 <= y < self.columnas and
                not visitado[x][y] and self.rejilla[x][y] == 0)

    def encontrar_ruta(self):
        visitado = [[False for _ in range(self.columnas)] for _ in range(self.filas)]
        if self.dfs(0, 0, visitado):
            return self.ruta
        else:
            return None

    def dfs(self, x, y, visitado):
        if x == self.filas - 1 and y == self.columnas - 1:
            self.ruta.append((x, y))
            return True
```

Definimos la función que hará uso de la inicialización de las filas y columnas para verificar el método para un movimiento válido.

Posteriormente inicia un arreglo de encontrar ruta que encontrara la ruta y la definición “dfs” es un algoritmo de buqueda profunda en donde se marca la celda actual como una visita y la marca en una ruta haciendo un movimiento en dado caso que lo pueda hacer, de lo contrario

retrocede

```
            # Movimiento a la derechaaa
            if self.dfs(x, y + 1, visitado):
                return True
            # Mover para abajo
            if self.dfs(x + 1, y, visitado):
                return True

            # se retrocede si no hay una ruta adecuada
            self.ruta.pop()
            visitado[x][y] = False

        return False

# Ejemplo de uso
rejilla = [
    [0, 0, 1, 0],
    [0, 0, 1, 0],
    [1, 0, 0, 0],
    [0, 1, 0, 0]
]
```

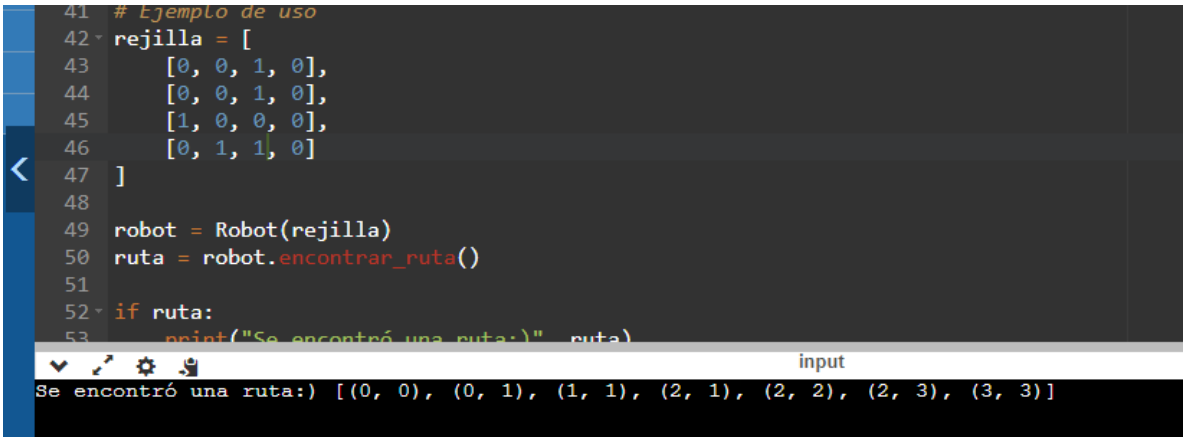
Posteriormente esa ruta es manejada por los movimientos de abajo y movimiento a la derecha almacenados por el agoritmo de búsqueda profunda “DFS” y retrocede en caso de noencontrar alguna. Después esta la matriz en donde se representa el mapa en donde se ejecutará la ruta

```
]

robot = Robot(rejilla)
ruta = robot.encontrar_ruta()

if ruta:
    print("Se encontró una ruta:)", ruta)
else:
    print("No se encontró una ruta :(")
```

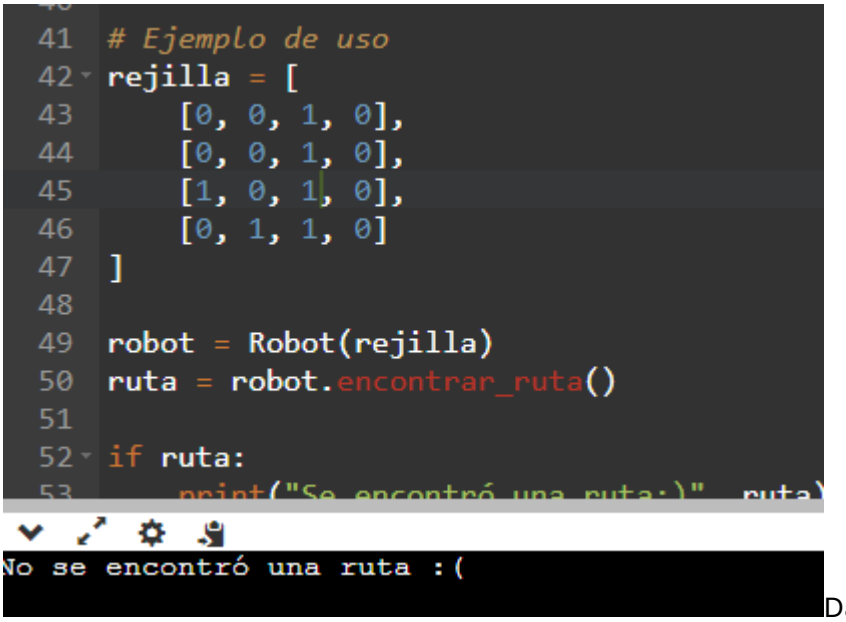
Posteriormente se imprime la ruta en caso de encontrarla y en caso contrario se imprime el mensaje que indica que no se pudo.

```
input
Se encontró una ruta:) [(0, 0), (0, 1), (1, 1), (2, 1), (2, 2), (2, 3), (3, 3)]
```

Dados esos valores de la matriz insertada de ejemplo se pudo averiguar la siguiente ruta de los pares ordenados.



Añadimos y modificamos la matriz viendo su comportamiento y obteniendo nuevos resultados



Dado que en la matriz rejilla hay una columna de unos no es posible encontrar la ruta correcta