Instituto Politécnico Nacional

Escuela Superior de Cómputo

Unidad académica: Análisis y Diseño de Algoritmos

Actividad:

"Contador de rutas"

Equipo:

Solares Velasco Arturo Misael

Solis Lugo Mayra

Grupo: 3CV1

Profesor: García Floriano Andrés

Fecha:

21 de Mayo de 2024

Reporte de Práctica: Contar Rutas en un Laberinto

Introducción

En esta práctica, se desarrolló una función en Python que calcula la cantidad de rutas posibles desde la esquina superior izquierda hasta la esquina inferior derecha de un laberinto representado por una matriz. La matriz contiene ceros y unos, donde los ceros representan caminos libres y los unos representan obstáculos. Este reporte detalla la implementación de la función `contar_rutas` y su aplicación en un ejemplo de uso.

Descripción de la Función

La función `contar_rutas` utiliza programación dinámica para contar el número de rutas posibles en un laberinto dado. A continuación, se describe el código y su funcionamiento:

python
Copy code

```
def contar_rutas(laberinto):
    m = len(laberinto)
    n = len(laberinto[0])

    if laberinto[0][0] == 1 or laberinto[m-1][n-1] == 1:
        return 0

    dp = [[0] * n for _ in range(m)]
    dp[0][0] = 1

    for i in range(m):
        for j in range(n):
            if laberinto[i][j] == 1:
                dp[i][j] = 0
            else:
                if i > 0:
                    dp[i][j] += dp[i-1][j]
                if j > 0:
                    dp[i][j] += dp[i][j-1]

    return dp[m-1][n-1]
```

Análisis de la Implementación

1. **Inicialización de Variables**:
    - `m` y `n` se asignan al número de filas y columnas del laberinto respectivamente.
    - Si la celda inicial (`laberinto[0][0]`) o la celda final (`laberinto[m-1][n-1]`) contienen un obstáculo (valor 1), la función retorna 0 inmediatamente, ya que no hay rutas posibles.
2. **Creación de la Matriz `dp`**:
    - Se inicializa una matriz `dp` de tamaño `m x n` con ceros.
    - `dp[0][0]` se establece en 1, ya que hay una única forma de estar en la celda inicial.
3. **Relleno de la Matriz `dp`**:
    - Se iteran las celdas del laberinto utilizando dos bucles anidados.
    - Si una celda contiene un obstáculo (`laberinto[i][j] == 1`), `dp[i][j]` se establece en 0.
    - Si una celda es accesible, se suma el número de formas de llegar desde la celda superior (`dp[i-1][j]`) y desde la celda a la izquierda (`dp[i][j-1]`).
4. **Resultado Final**:
    - La función retorna el valor en `dp[m-1][n-1]`, que representa el número de rutas posibles para llegar a la esquina inferior derecha del laberinto.

Ejemplo de Uso

Se proporciona un ejemplo de uso de la función con un laberinto de 3x3:

python
Copy code

```
laberinto = [
    [0, 0, 0],
    [0, 1, 0],
    [0, 0, 0]
]

print(f"Rutas posibles: {contar_rutas(laberinto)}")
```

Salida esperada:

yaml

Copy code

```
Rutas posibles: 2
```

Este resultado indica que hay dos rutas posibles desde la esquina superior izquierda hasta la esquina inferior derecha del laberinto dado.

Conclusión

La función `contar_rutas` es una implementación eficiente que utiliza programación dinámica para resolver el problema de contar rutas en un laberinto. La solución es óptima en términos de tiempo y espacio, ya que solo requiere una matriz adicional del mismo tamaño que el laberinto original. Esta práctica ilustra cómo abordar problemas de caminos y rutas en estructuras matriciales utilizando técnicas avanzadas de programación.