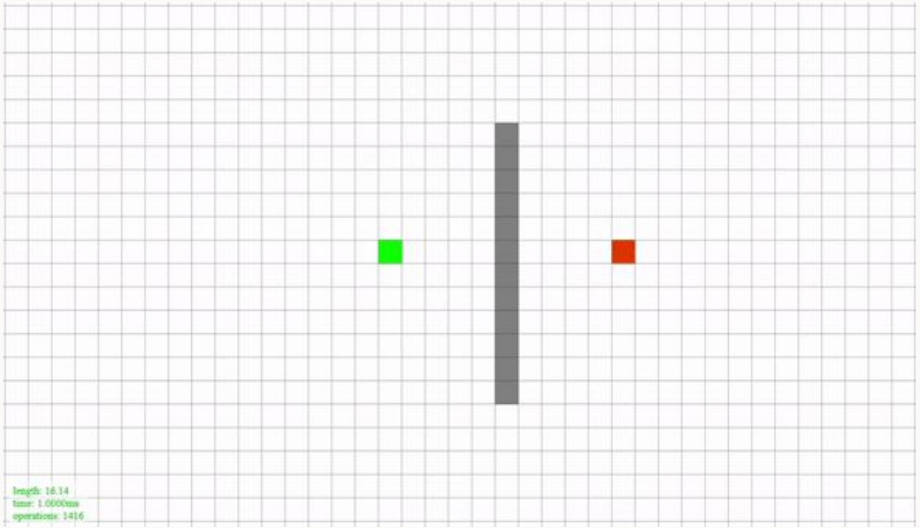
Ayudantia 5

A* y heurísticas

Por Daniel Toribio y Octavio Águila

15 de septiembre 2023

Recordemos Dijkstra...



```
for each nodo n ∈ grafo G:
     dist[n] ← inf
     prev[n] ← None
     Insert n a cola Q

dist[origen] <- 0
while Q ≠ ∅:
     u <- nodo en Q con menor dist[u]
     Extraer u de Q

     for each v ∈ Succ(u) ∩ Q:
          new_dist ← dist[u] + arista(u, v)
          If new_dist < dist[v]:
               dist[v] ← new_dist
               prev[v] ← u

return dist[], prev[]
```

Recordemos Dijkstra...

- Encuentra **caminos más cortos en un grafo ponderado** → encuentra soluciones **óptimas.**

- Pero ¿Qué sucede si el grafo de búsqueda es **MUY** grande?

Recordemos Dijkstra...

- Encuentra **caminos más cortos en un grafo ponderado** → encuentra soluciones **óptimas.**

- Pero ¿Qué sucede si el grafo de búsqueda es **MUY** grande?

Alto consumo de memoria

Exploración innecesaria

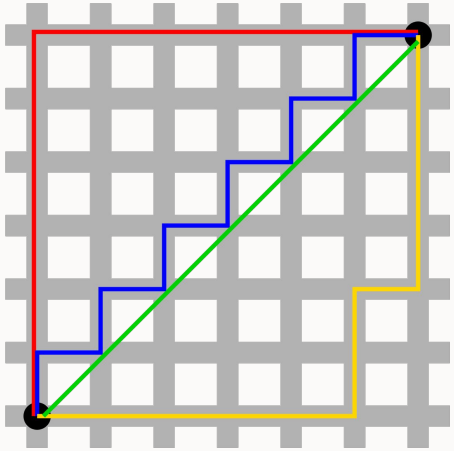
Gran tiempo de ejecución

- Queremos que nuestra búsqueda sea en **dirección al nodo objetivo**, evitando **revisar nodos innecesarios**. Y para ello necesitamos más información...

Función Heurística

- Función que estima la cercanía de un nodo ***s*** a un nodo objetivo **G.**
- Con ella podemos discriminar qué alternativa de nodo es **mejor** que otro, priorizando así revisar caminos más **prometedores.**
- A diferencia del costo, una heurística proporciona una mejor aproximación del problema a partir de **información limitada o parcial.**

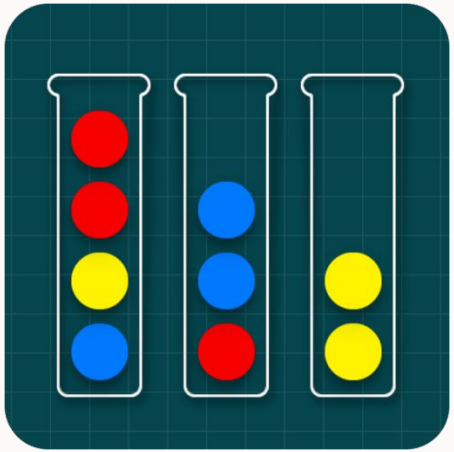
Ejemplos clásicos de Heurísticas



Distancia Manhattan

Distancia Euclidiana

Número de piezas fuera de lugar



Cantidad de bolitas que no tienen el mismo color

Admisibilidad

Una función heurística *h* se dice *admisible* si para todo estado *s:*

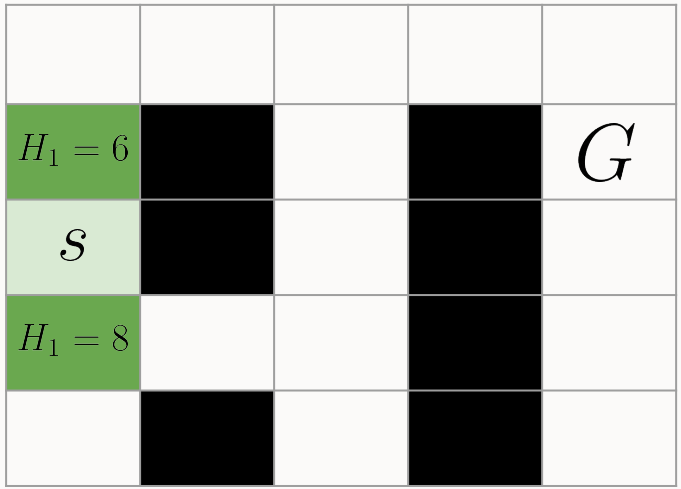
$$h(s) \leq h^*(s)$$

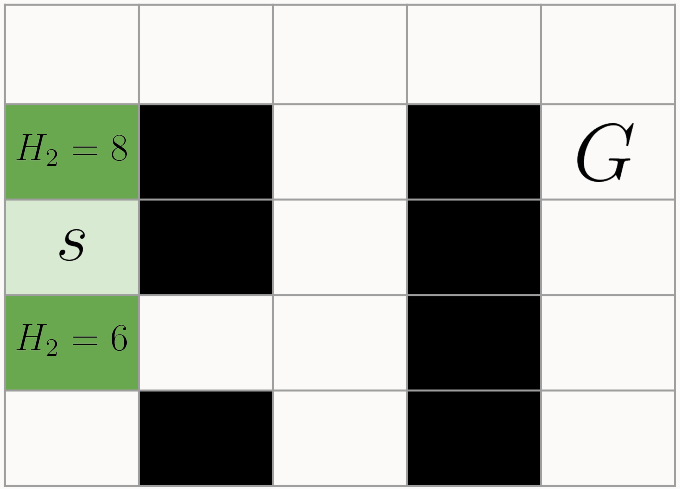
Donde *h*(s)* es el costo de un camino óptimo desde *s* a un estado objetivo.

Nuestra función es admisible si nunca sobreestima respecto a un camino óptimo

Admisibilidad

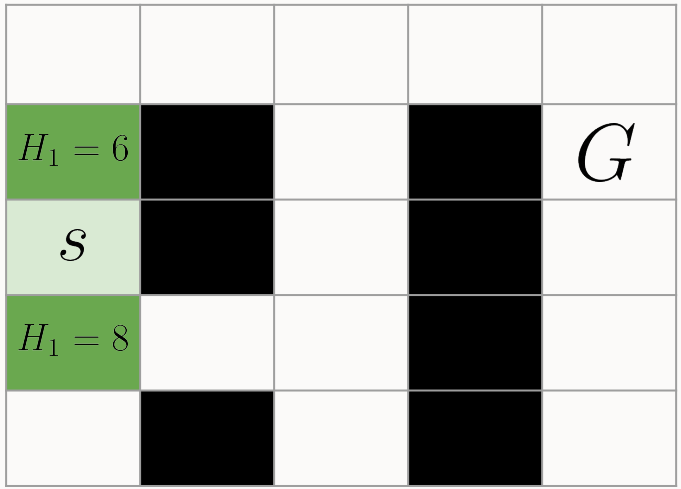
Sea el costo de moverse entre casillas igual a 1. ¿Cuál heurística es admisible?



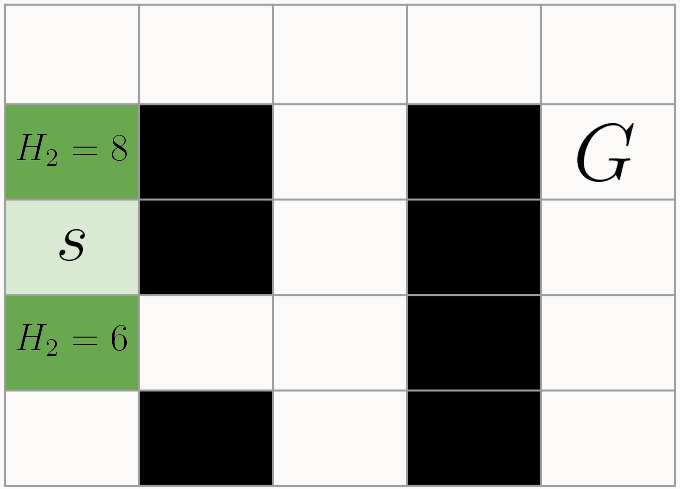


Admisibilidad

Sea el costo de moverse entre casillas igual a 1. ¿Cuál heurística es admisible?



¡H_1 es admisible!



H_2 no es admisible ya que $H_2 = 8 > 6 = h^*$

Consistencia

Una función heurística *h* se dice *consistente* si y solo si:

- $h(s) = 0$, para todo $s \in G$
- $h(s) \leq c(s, s') + h(s')$ para todo vecino s' de s

O de manera equivalente $h(s) - h(s') \leq c(s, s')$

Nuestra función es consistente si **nunca sobreestima en los estados objetivos** ni el **costo es mayor que la variación de la heurística** entre dos estados vecinos.

Consistencia

Sea el costo de moverse entre casillas igual a 1. ¿Son estas heurísticas consistentes?

$H_1 = 10$	$H_1 = 8$	$H_1 = 6$	$H_1 = 4$	$H_1 = 2$
$H_1 = 12$		$H_1 = 8$		$H_1 = 0$
S		$H_1 = 10$		$H_1 = 2$
$H_1 = 16$	$H_1 = 14$	$H_1 = 12$		$H_1 = 4$
$H_1 = 18$		$H_1 = 14$		$H_1 = 6$

G

$H_2 = 5$	$H_2 = 4$	$H_2 = 3$	$H_2 = 2$	$H_2 = 1$
$H_2 = 4$		$H_2 = 2$		$H_2 = 0$
S		$H_2 = 3$		$H_2 = 1$
$H_2 = 6$	$H_2 = 5$	$H_2 = 4$		$H_2 = 2$
$H_2 = 7$		$H_2 = 5$		$H_2 = 3$

G

Consistencia

Sea el costo de moverse entre casillas igual a 1. ¿Son estas heurísticas consistentes?

$H_1 = 10$	$H_1 = 8$	$H_1 = 6$	$H_1 = 4$	$H_1 = 2$
$H_1 = 12$		$H_1 = 8$		$H_1 = 0$
S		$H_1 = 10$		$H_1 = 2$
$H_1 = 16$	$H_1 = 14$	$H_1 = 12$		$H_1 = 4$
$H_1 = 18$		$H_1 = 14$		$H_1 = 6$

G

¡H_1 no es consistente!

$H_2 = 5$	$H_2 = 4$	$H_2 = 3$	$H_2 = 2$	$H_2 = 1$
$H_2 = 4$		$H_2 = 2$		$H_2 = 0$
S		$H_2 = 3$		$H_2 = 1$
$H_2 = 6$	$H_2 = 5$	$H_2 = 4$		$H_2 = 2$
$H_2 = 7$		$H_2 = 5$		$H_2 = 3$

G

H_2 si lo es :)

Importante

Teorema

Si una heurística *h* es consistente, entonces *h* es admisible.

A*

A* es un algoritmo utilizado en problemas de búsqueda, su característica distintiva es que utiliza heurística, utilizando la siguiente función de evaluación f(n) = g(n) + h(n)

- La rapidez y complejidad computacional está estrechamente relacionada a la heurística seleccionada.
- Siempre entrega soluciones óptimas si se usa con una **heurística admisible.**
- Dijkstra es A* con heurística cero.

Dijkstra

$$f(n) = g(n)$$

A*

$$f(n) = g(n) + h(n)$$

A*

$$f(n) = g(n) + h(n)$$

$g(n)$: Costo de un camino desde s_0 hasta un nodo *n*.

$g(n)$ **NO ES FUNCIÓN!**

Algoritmo de A*

Algoritmo A*

Input: Un problema de búsqueda $(\mathbf{S}, A, s_0, G)$

Output: Un nodo objetivo

1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
8. *Insertar v*

La extracción es del u con menor valor f(s) = g(s) + h(s)

Revisa si llego al nodo objetivo cuando se expande

Procedimiento de insertar v

Algoritmo de A*

Algoritmo A*

Input: Un problema de búsqueda $(\mathbf{S}, A, s_0, G)$

Output: Un nodo objetivo

1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
8. *Insertar v*

La extracción es del u con menor valor f(s) = g(s) + h(s)

Revisa si llego al nodo objetivo cuando se expande

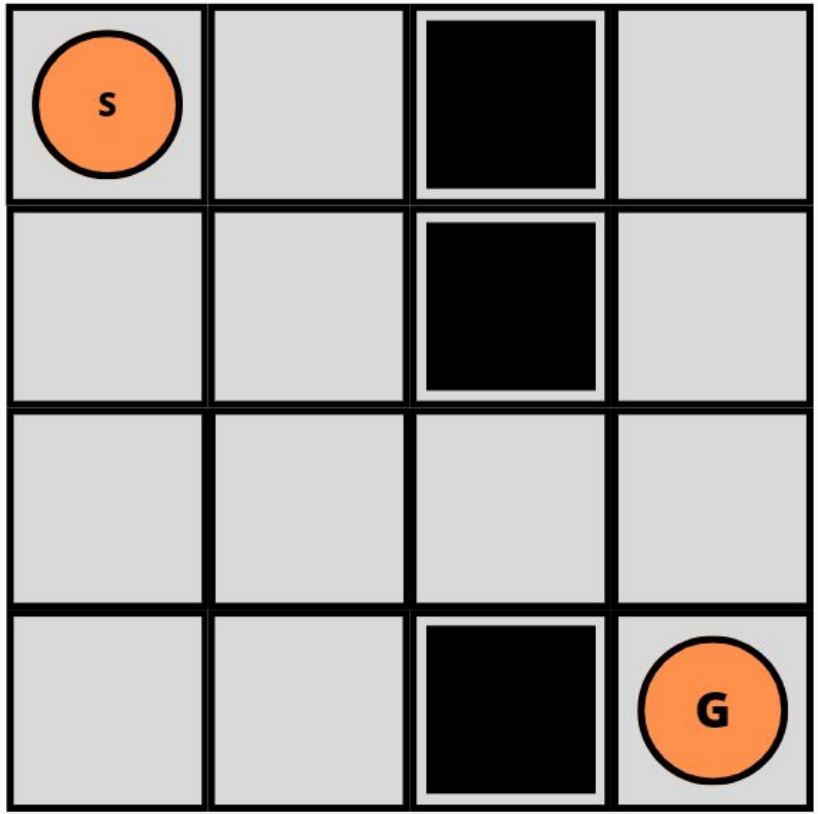
Procedimiento de insertar v

Procedimiento insertar v

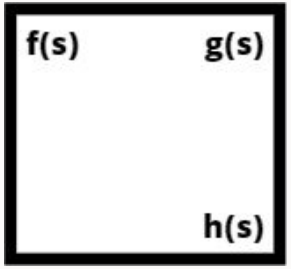
Insertar v en Open

1. $cost_v = g(u) + c(u, v)$ // el costo de llegar a v por u
2. **if** $cost_v \geq g(v)$ **return** // seguimos solo si $cost_v < g(v)$
3. $parent(v) \leftarrow u$
4. $g(v) \leftarrow cost_v$
5. $f(v) \leftarrow g(v) + h(v)$
6. **if** $v \in Open$ **then** Reordenar $Open$ // depende de la impl.
7. **else** Insertar v en $Open$

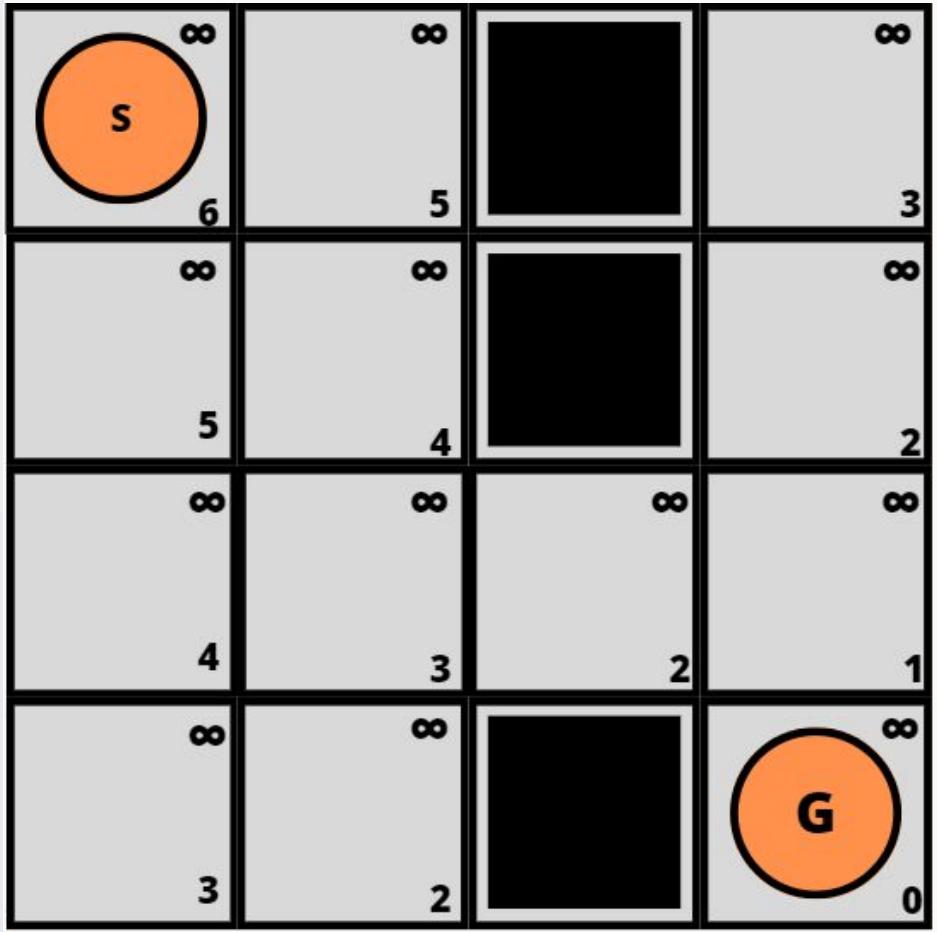
EJEMPLO



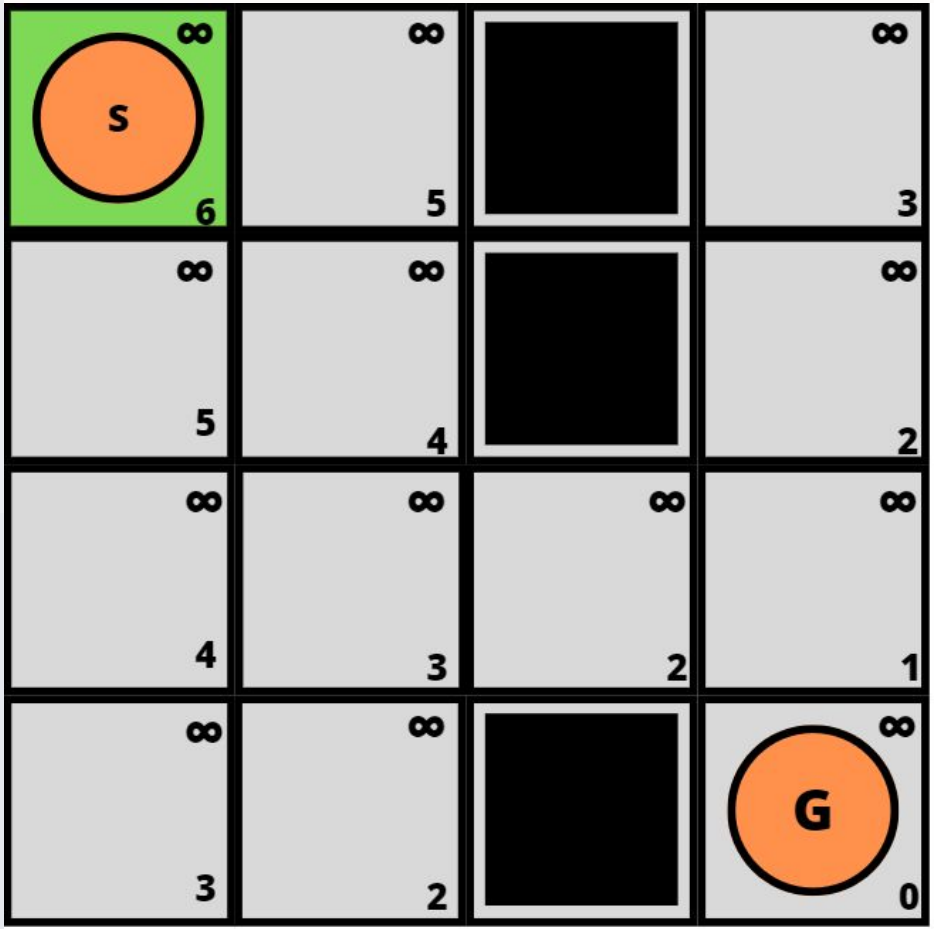
f(s) = g(s) + h(s)
g(s): largo del camino
h(s): distancia de manhattan



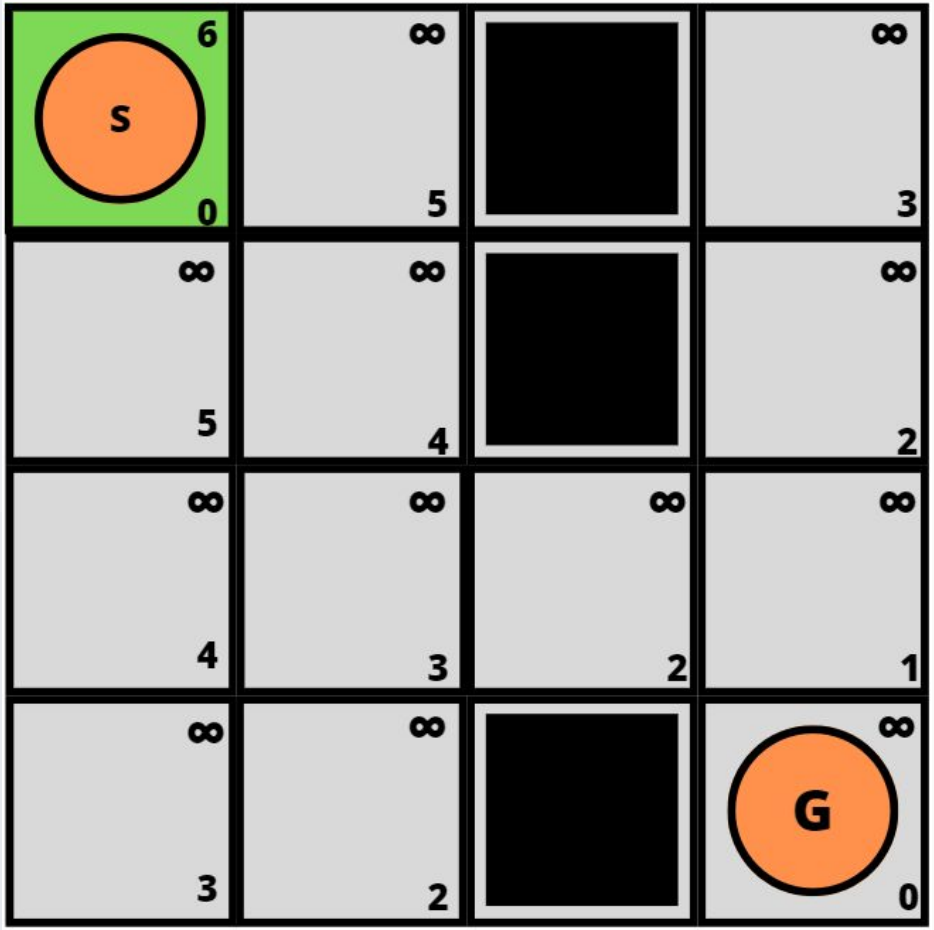
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



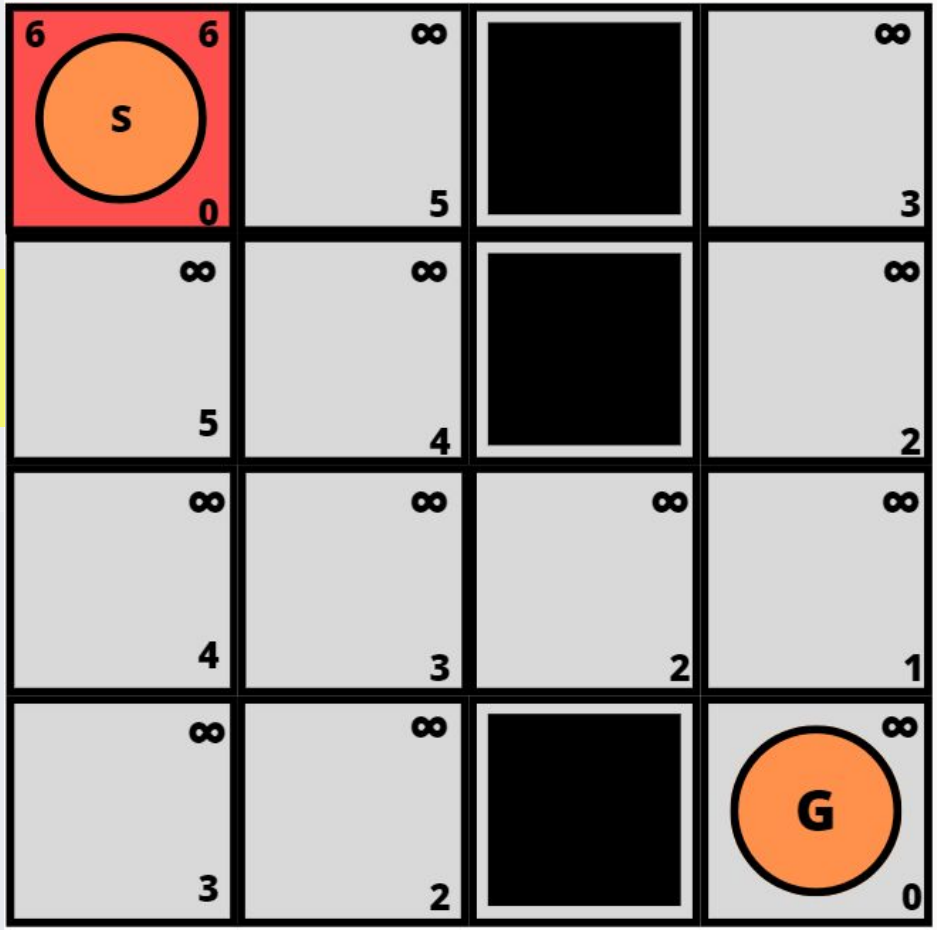
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



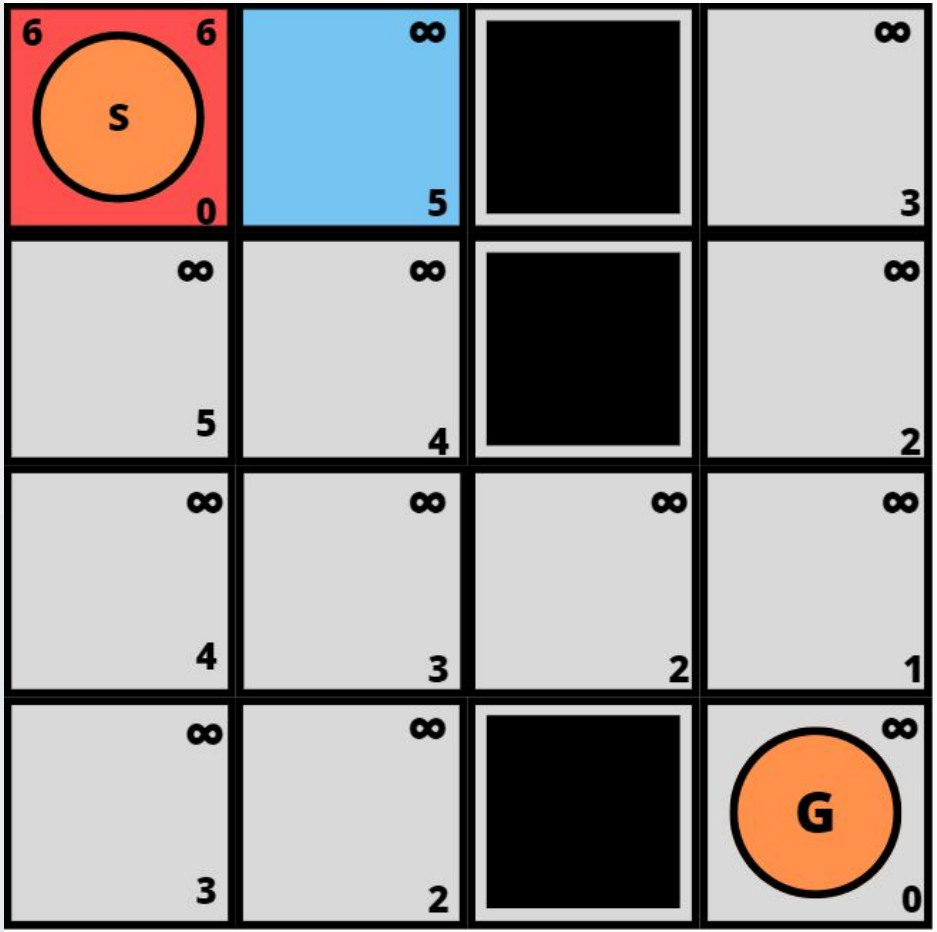
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



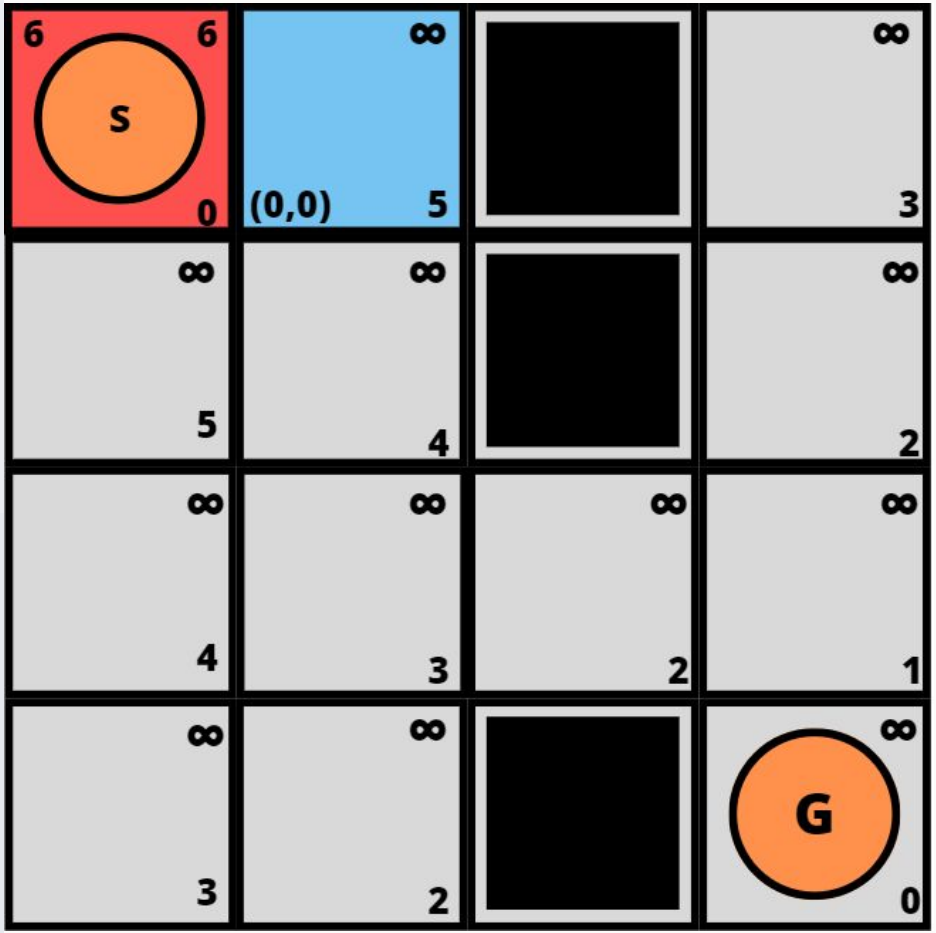
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
   1. $cost_v = g(u) + c(u, v)$
   2. **if** $cost_v \geq g(v)$ **return**
   3. $parent(v) \leftarrow u$
   4. $g(v) \leftarrow cost_v$
   5. $f(v) \leftarrow g(v) + h(v)$
   6. **if** $v \in Open$ **then** Reordenar $Open$
   7. **else** Insertar v en $Open$



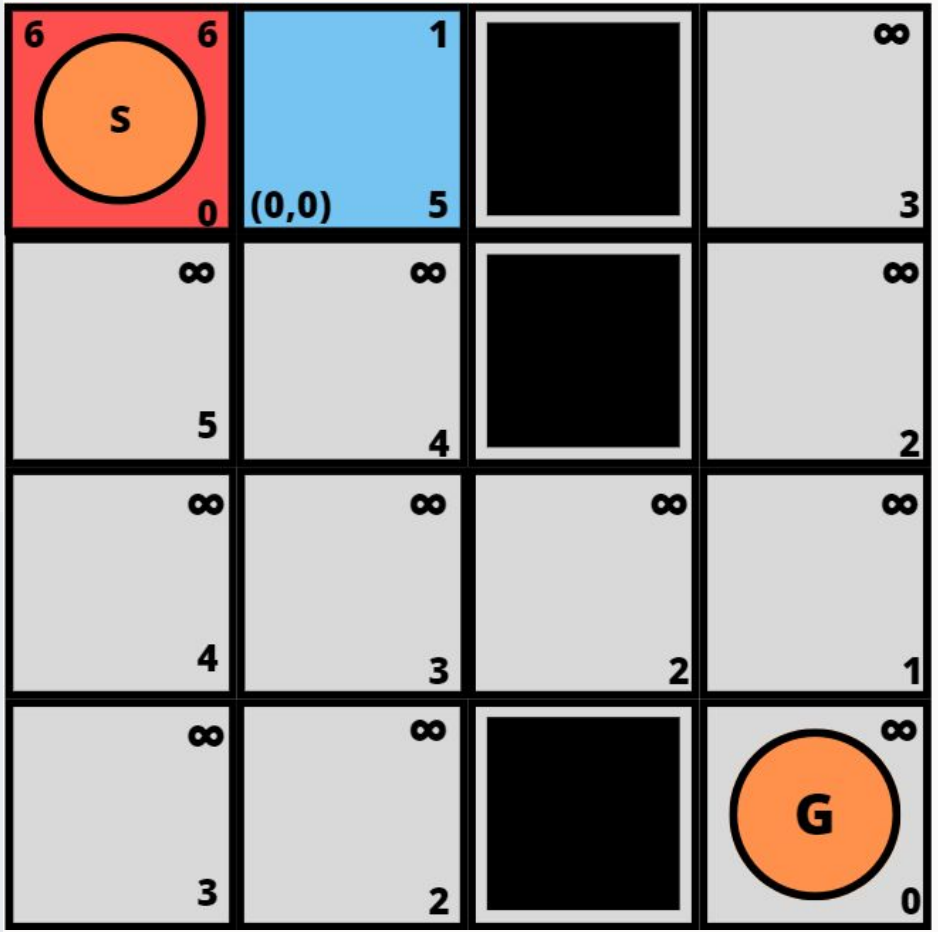
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
   1. $cost_v = g(u) + c(u, v)$
   2. **if** $cost_v \geq g(v)$ **return**
   3. $parent(v) \leftarrow u$
   4. $g(v) \leftarrow cost_v$
   5. $f(v) \leftarrow g(v) + h(v)$
   6. **if** $v \in Open$ **then** Reordenar $Open$
   7. **else** Insertar v en $Open$



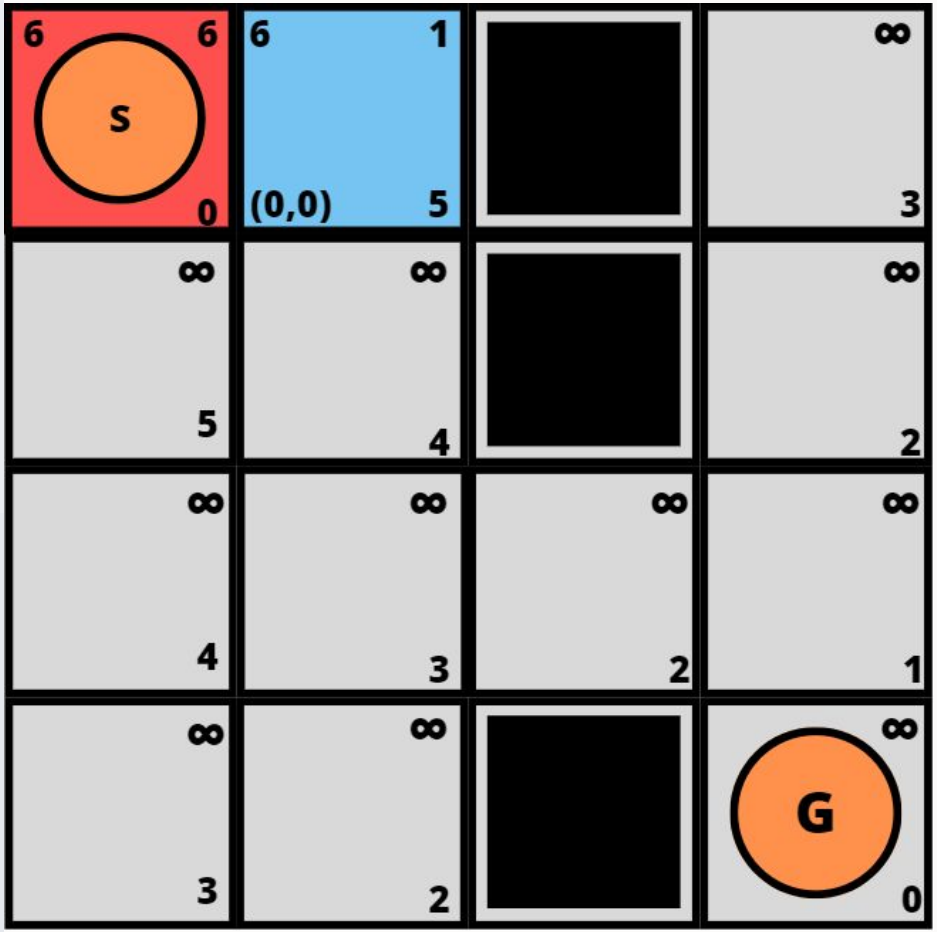
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



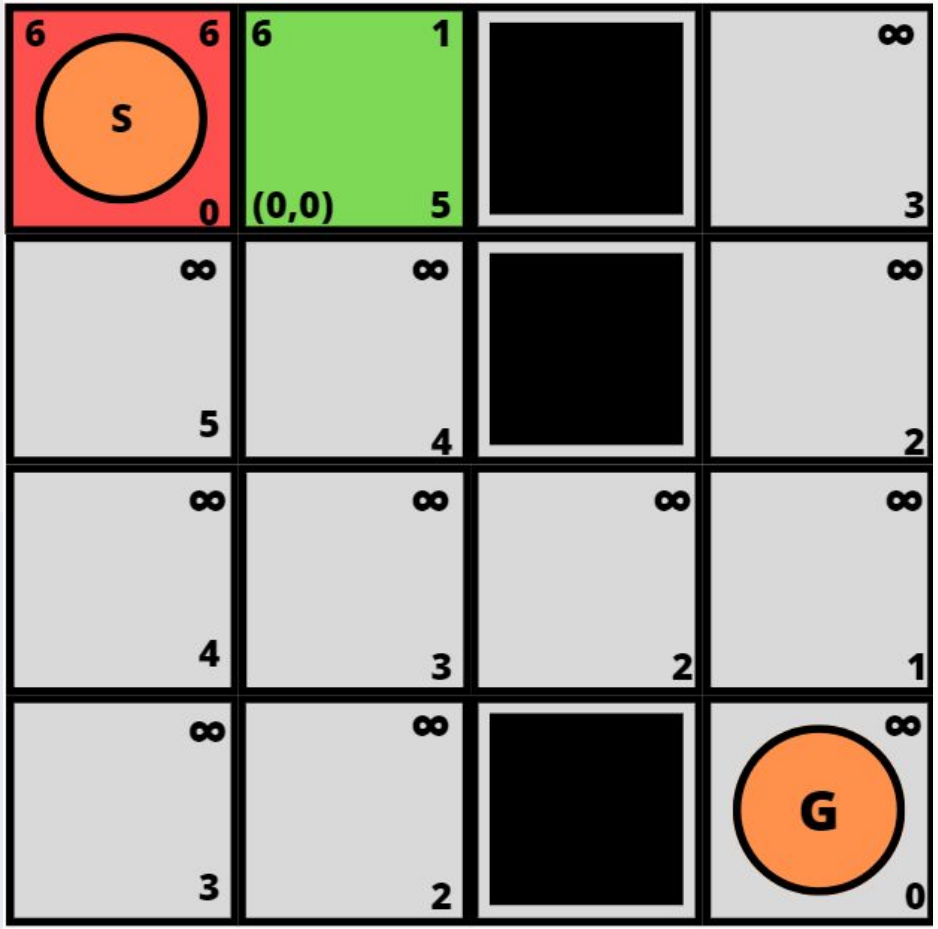
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



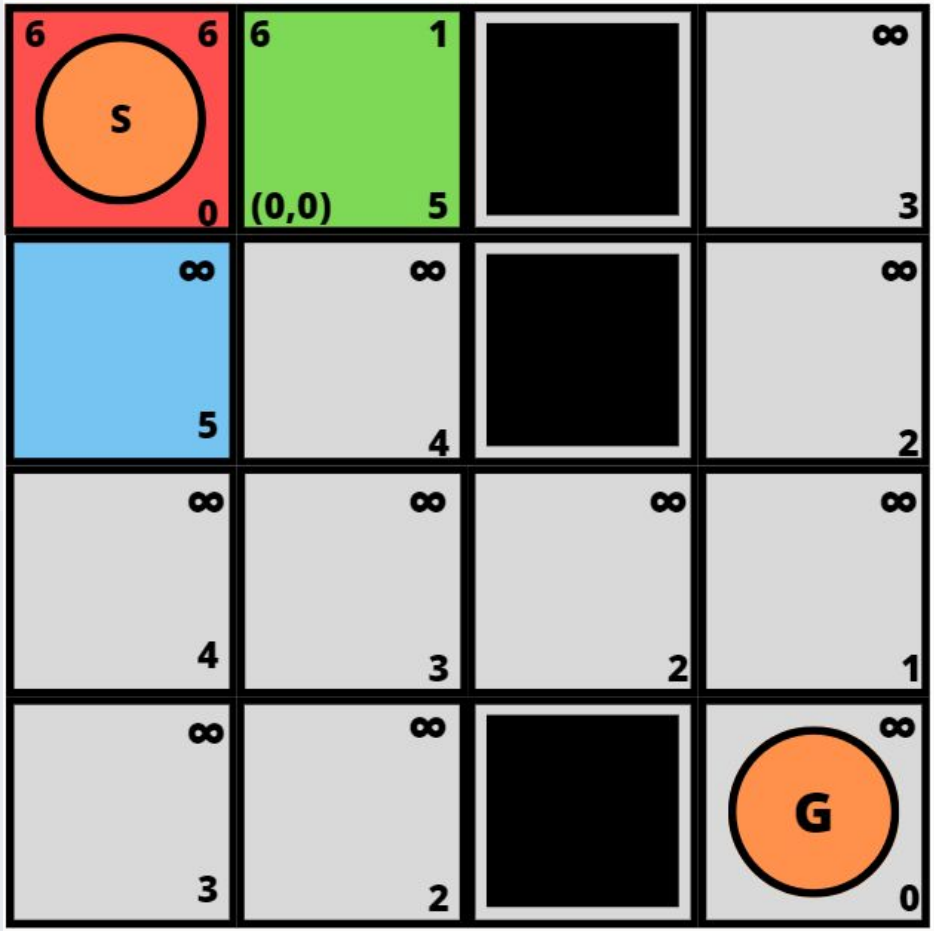
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



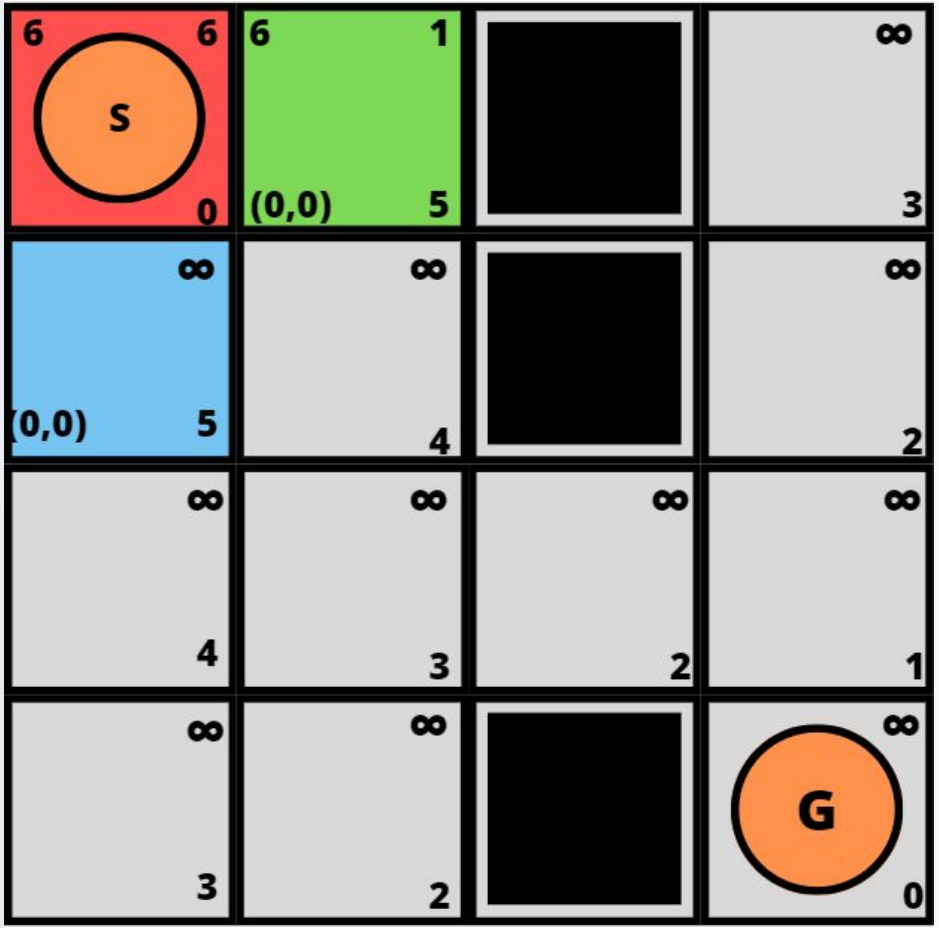
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



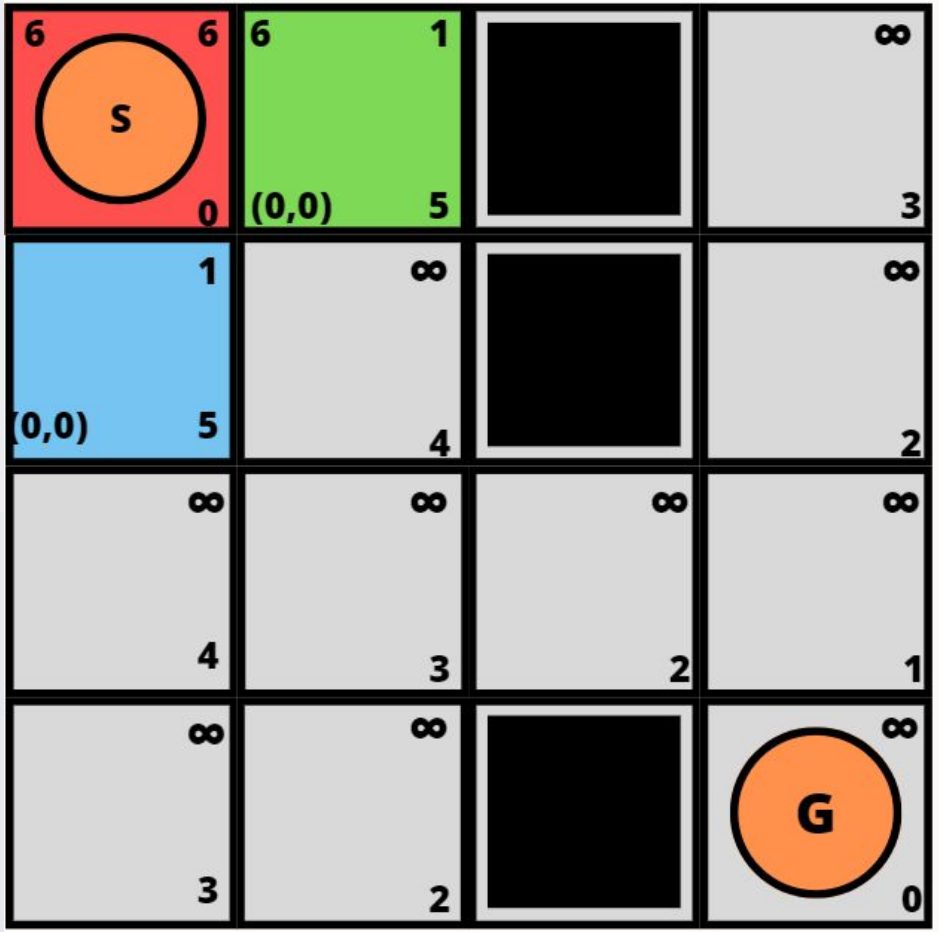
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



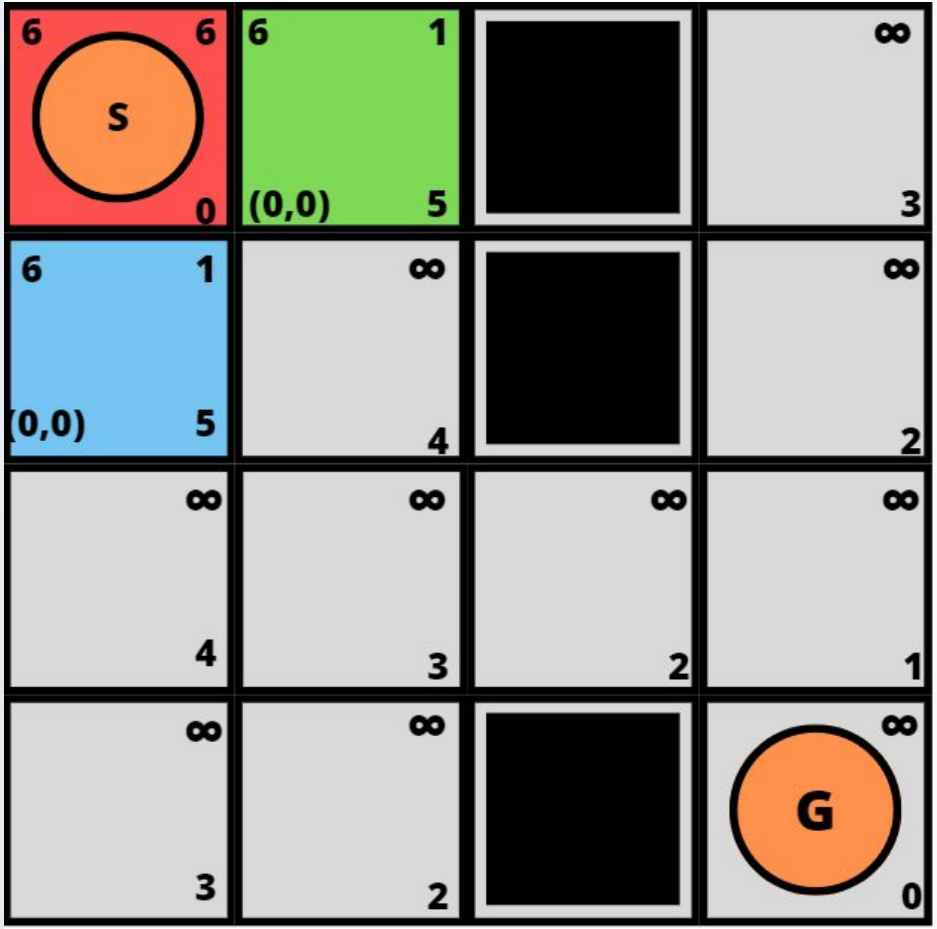
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



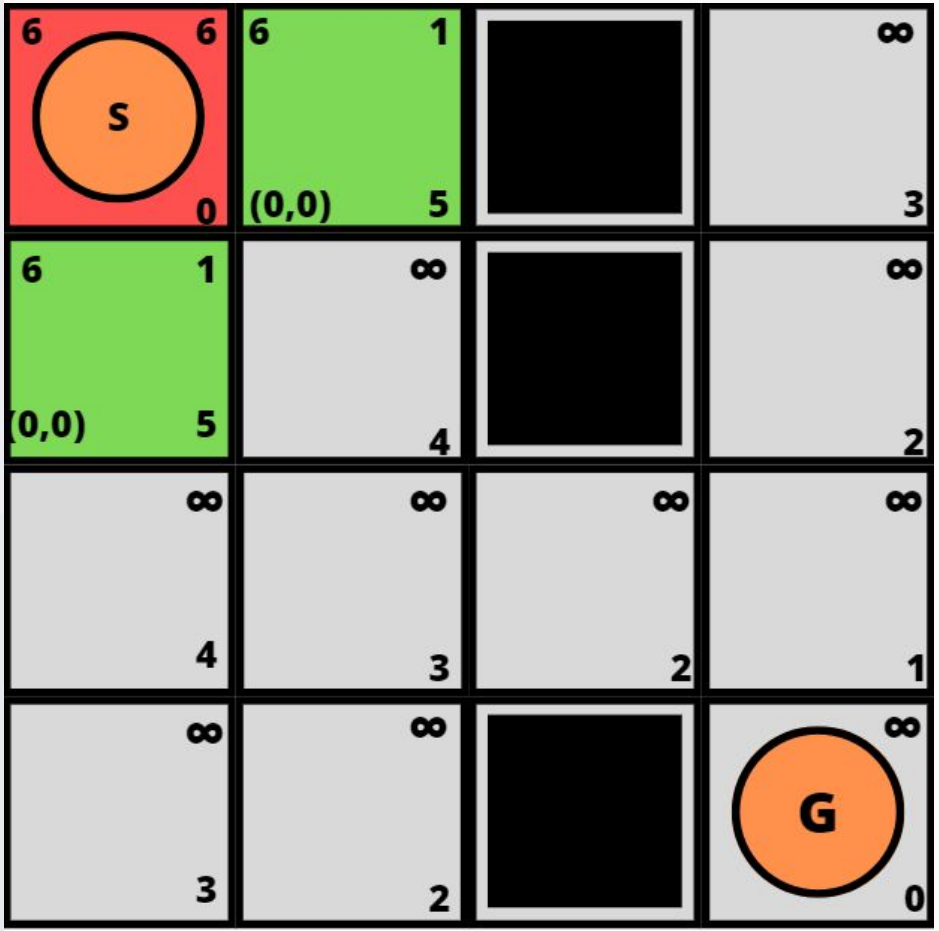
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



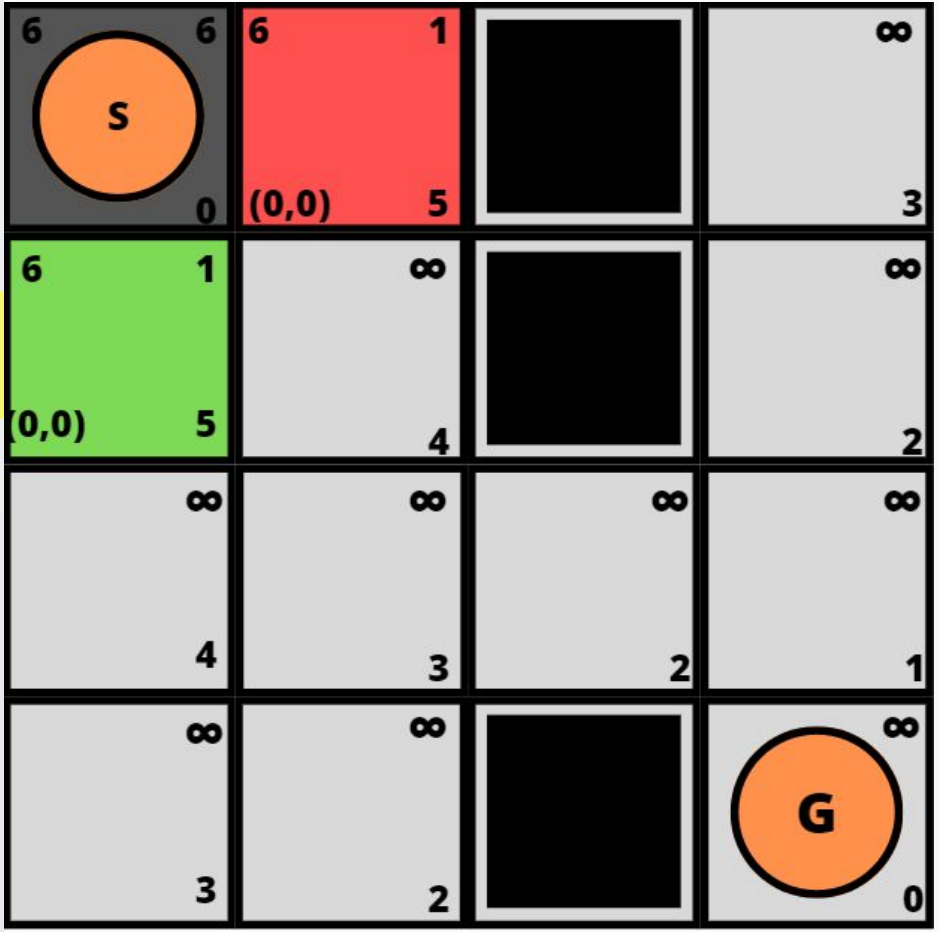
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



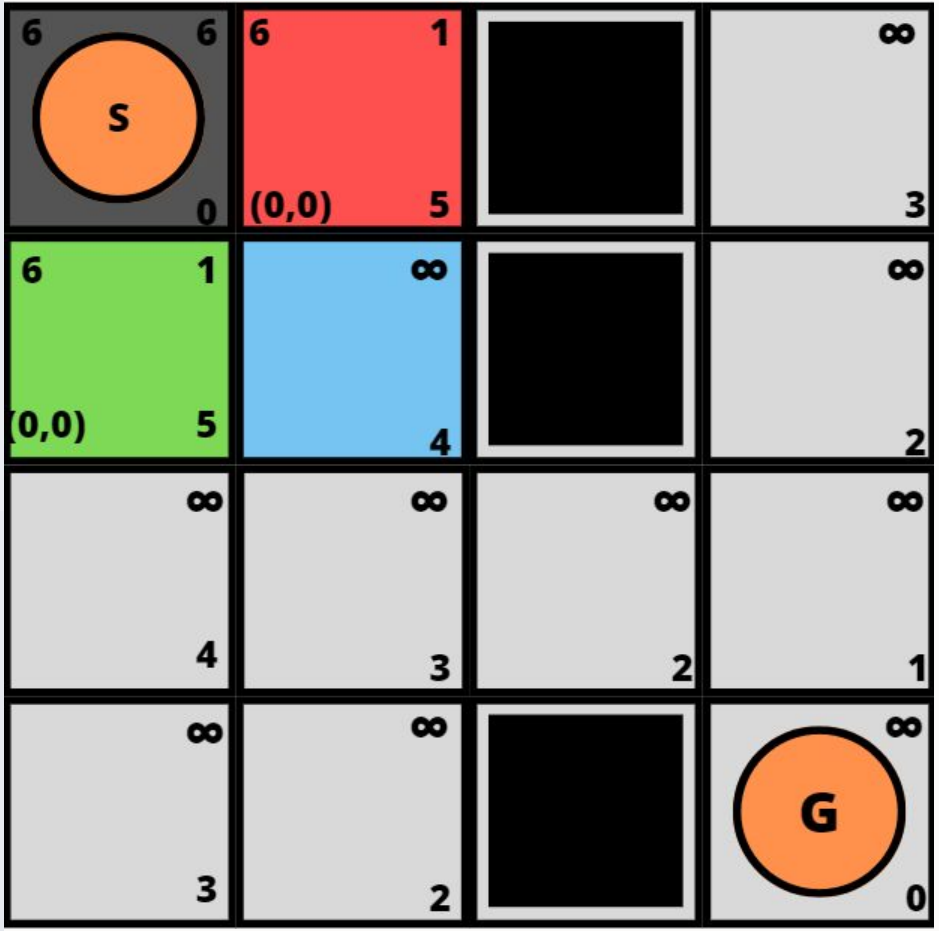
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



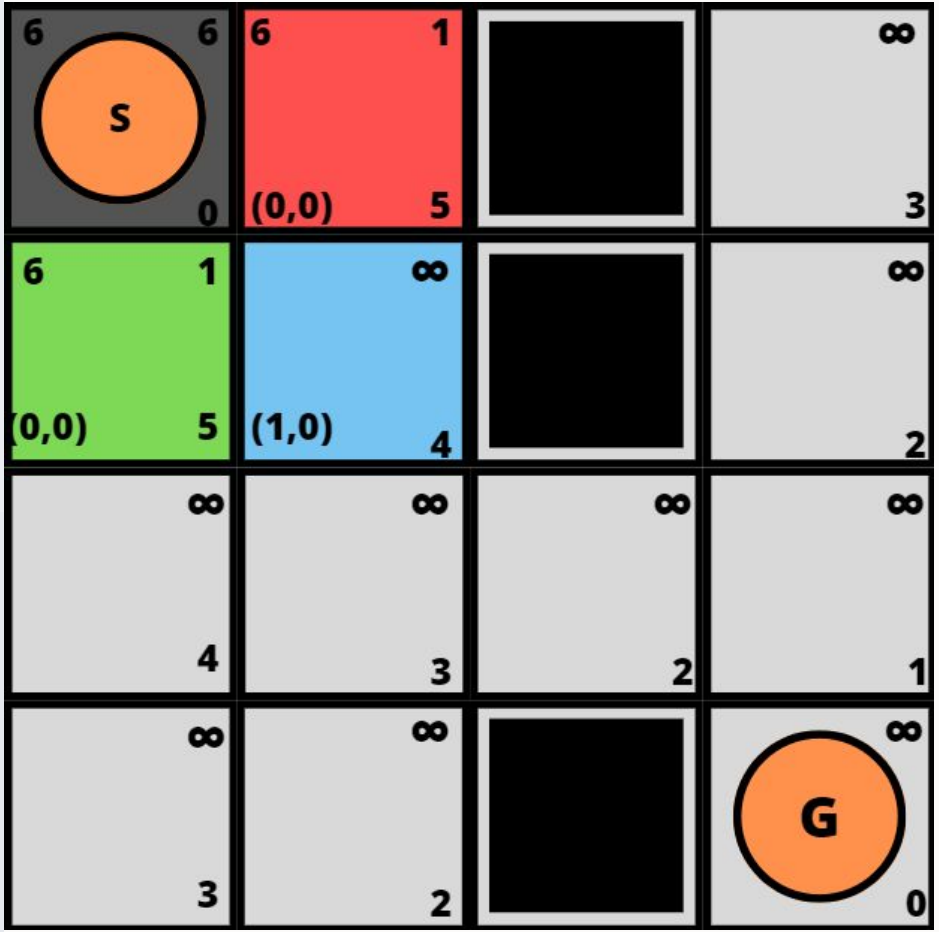
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



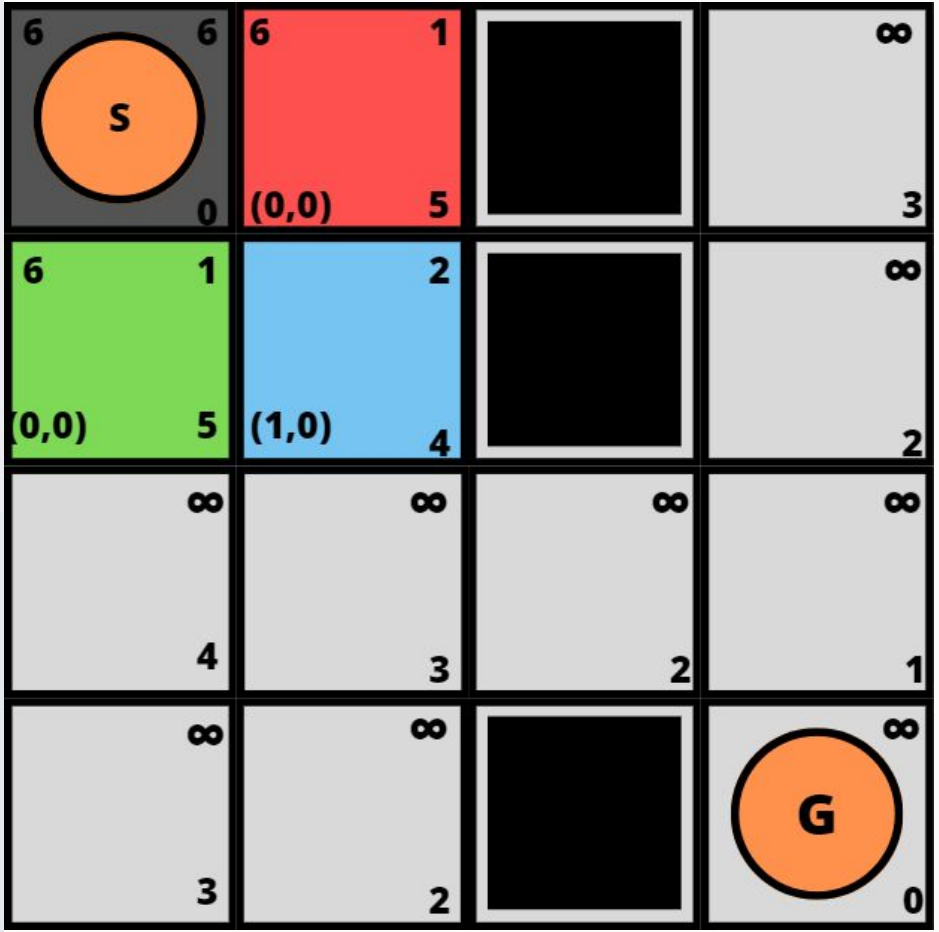
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
   1. $cost_v = g(u) + c(u, v)$
   2. **if** $cost_v \geq g(v)$ **return**
   3. $parent(v) \leftarrow u$
   4. $g(v) \leftarrow cost_v$
   5. $f(v) \leftarrow g(v) + h(v)$
   6. **if** $v \in Open$ **then** Reordenar $Open$
   7. **else** Insertar v en $Open$



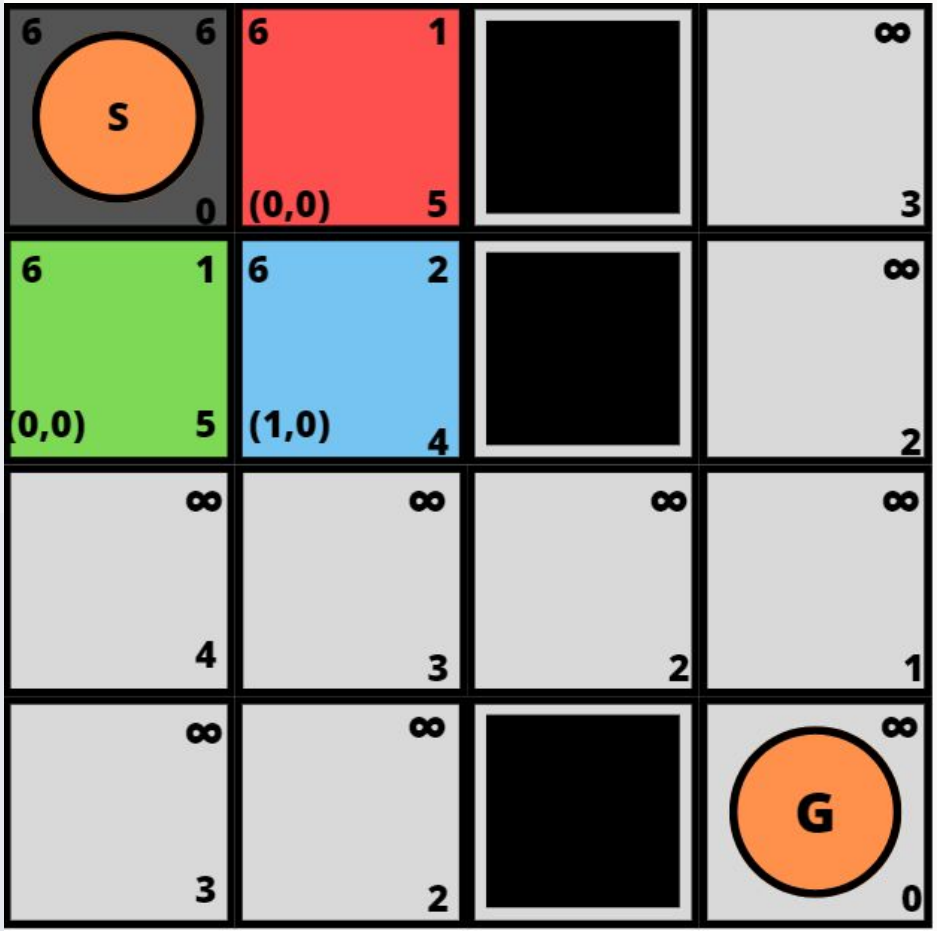
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



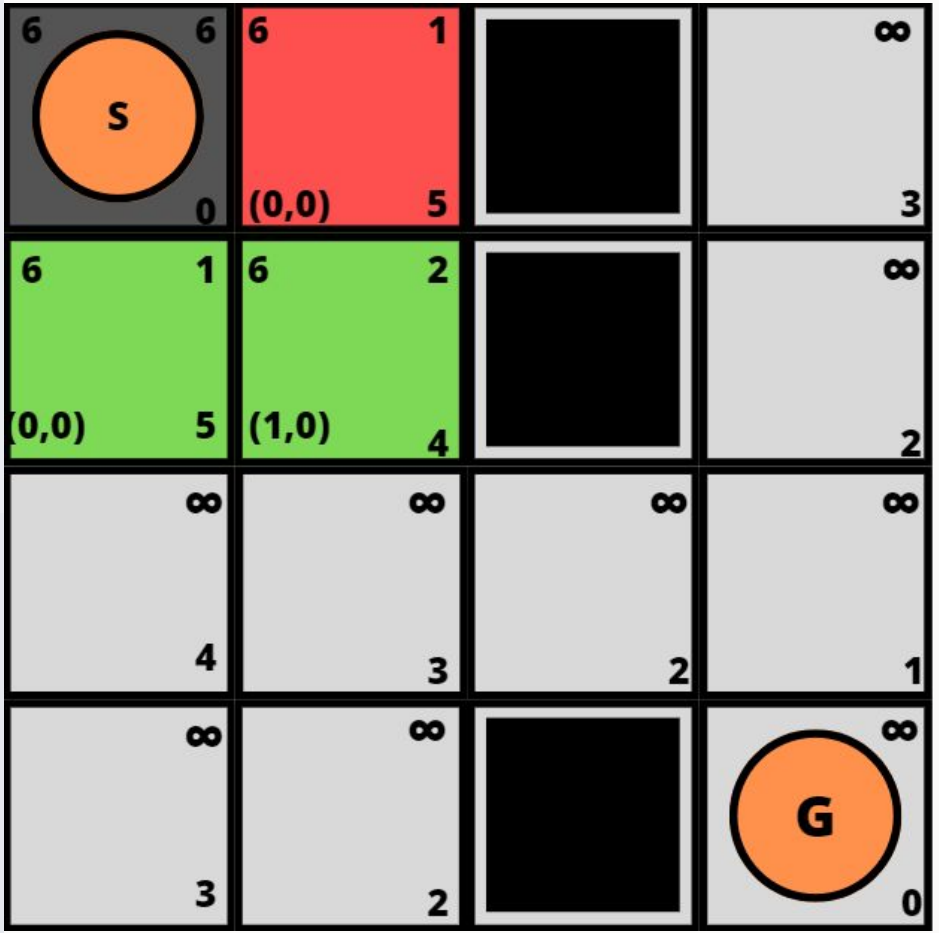
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



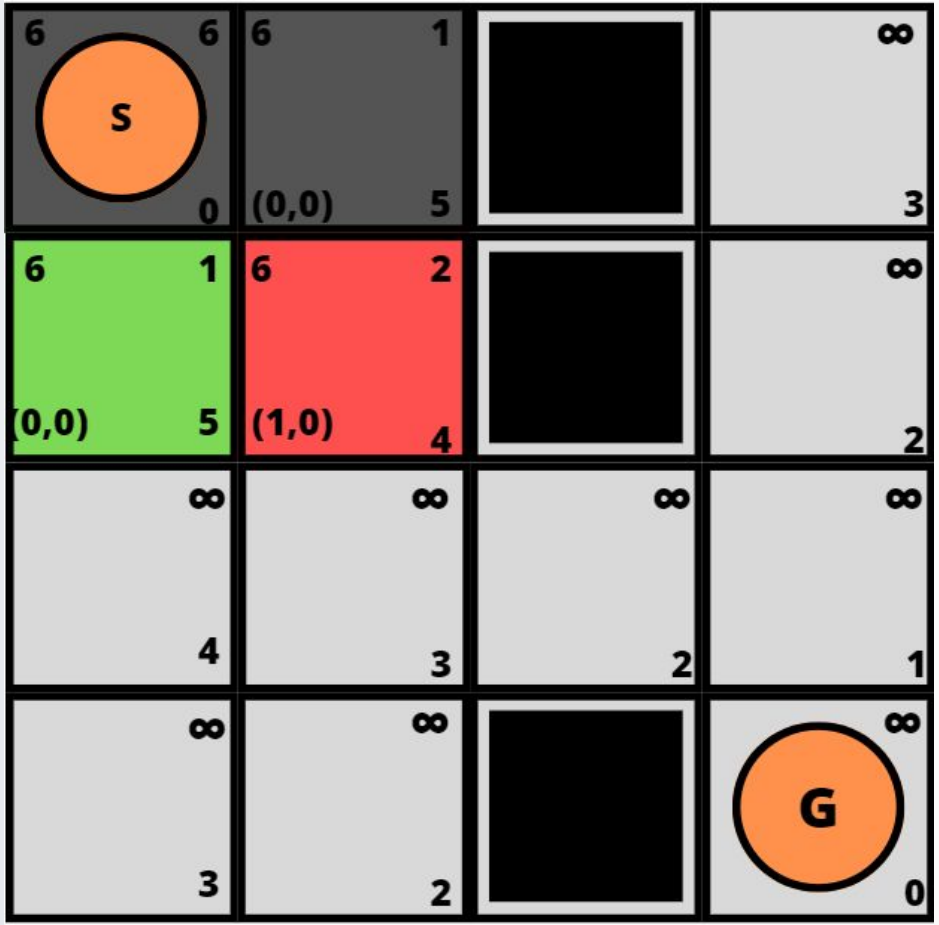
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



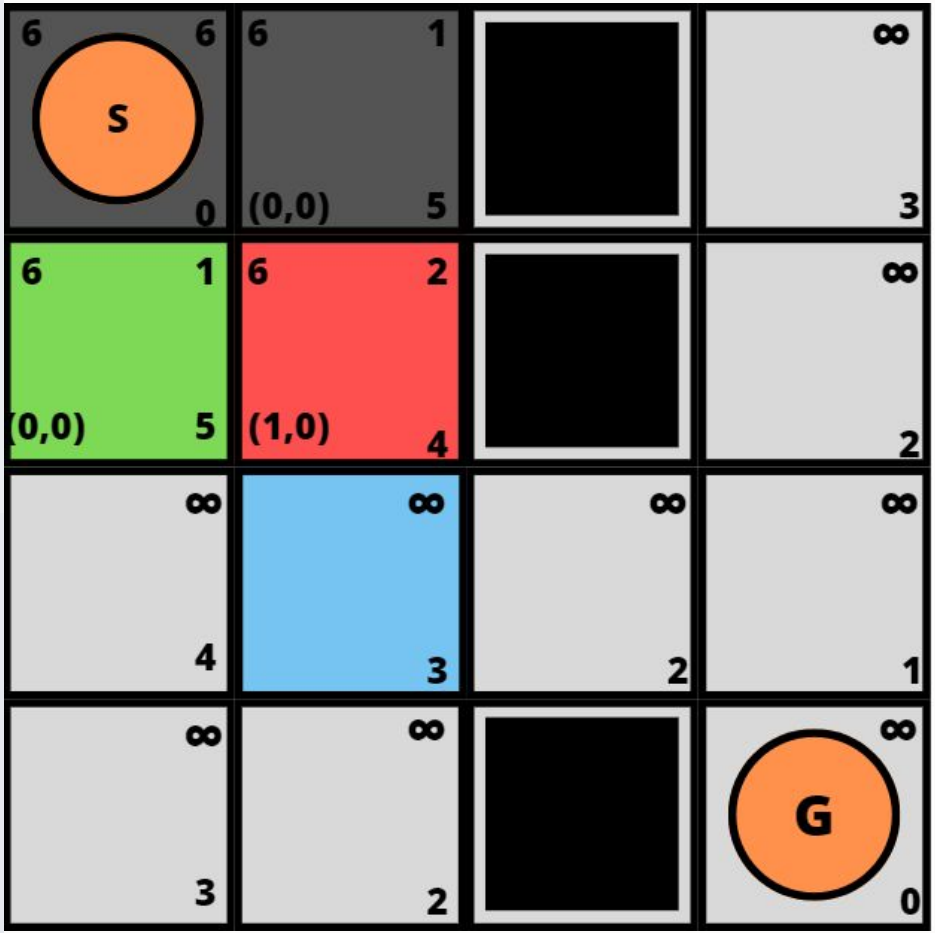
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



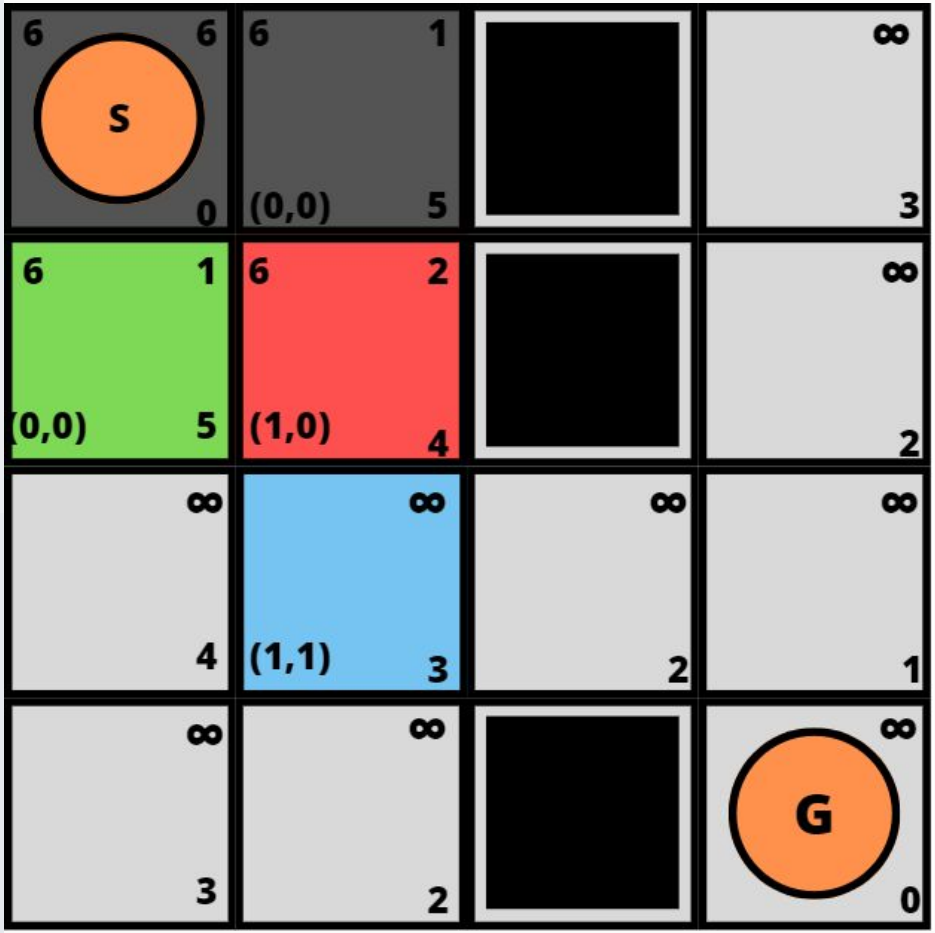
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



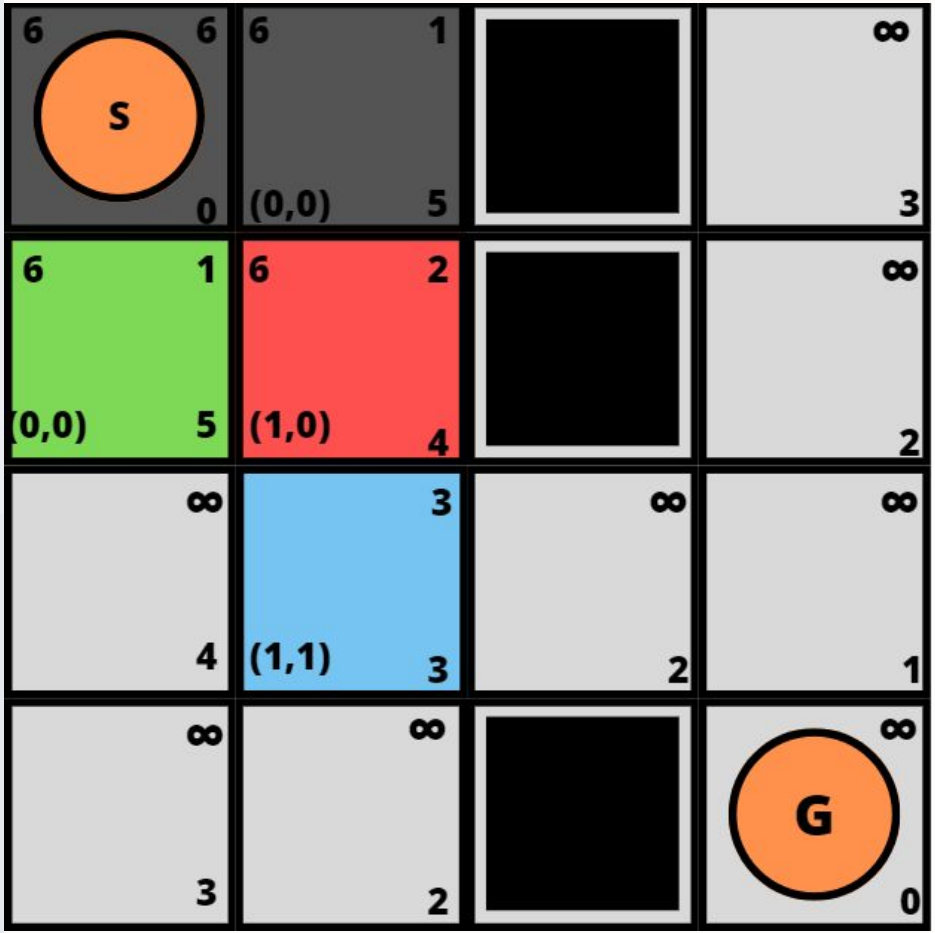
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
   1. $cost_v = g(u) + c(u, v)$
   2. **if** $cost_v \geq g(v)$ **return**
   3. $parent(v) \leftarrow u$
   4. $g(v) \leftarrow cost_v$
   5. $f(v) \leftarrow g(v) + h(v)$
   6. **if** $v \in Open$ **then** Reordenar $Open$
   7. **else** Insertar v en $Open$



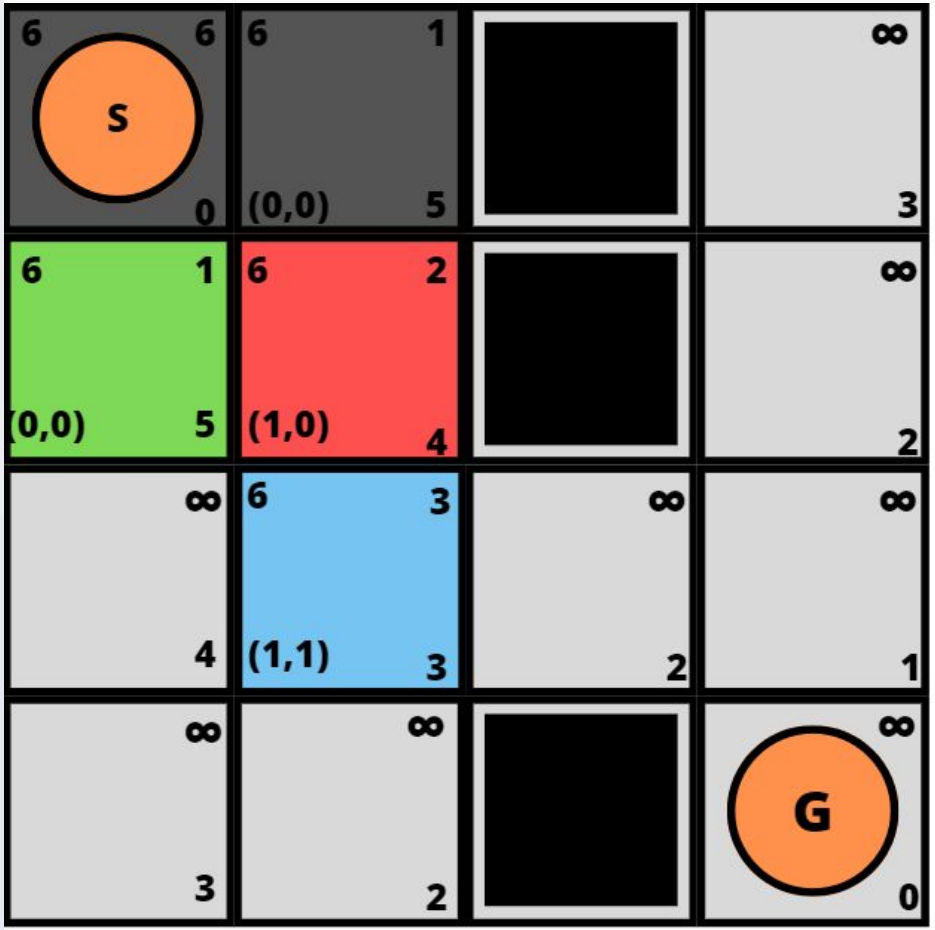
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



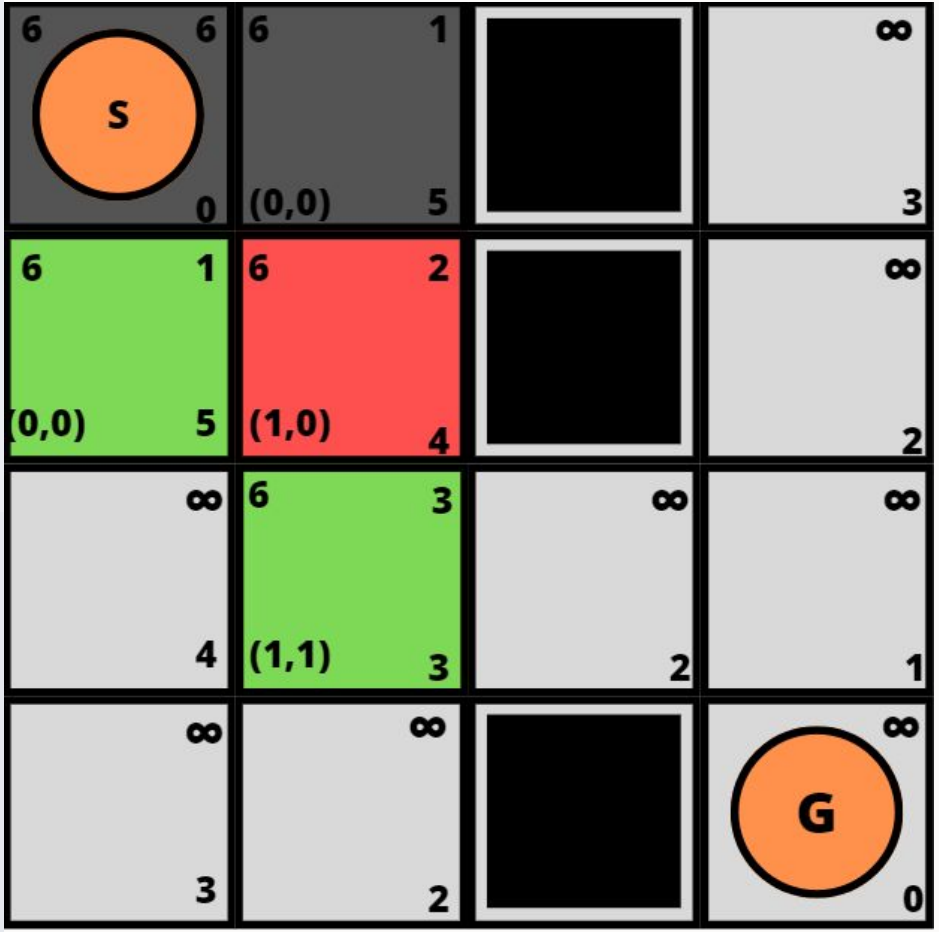
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



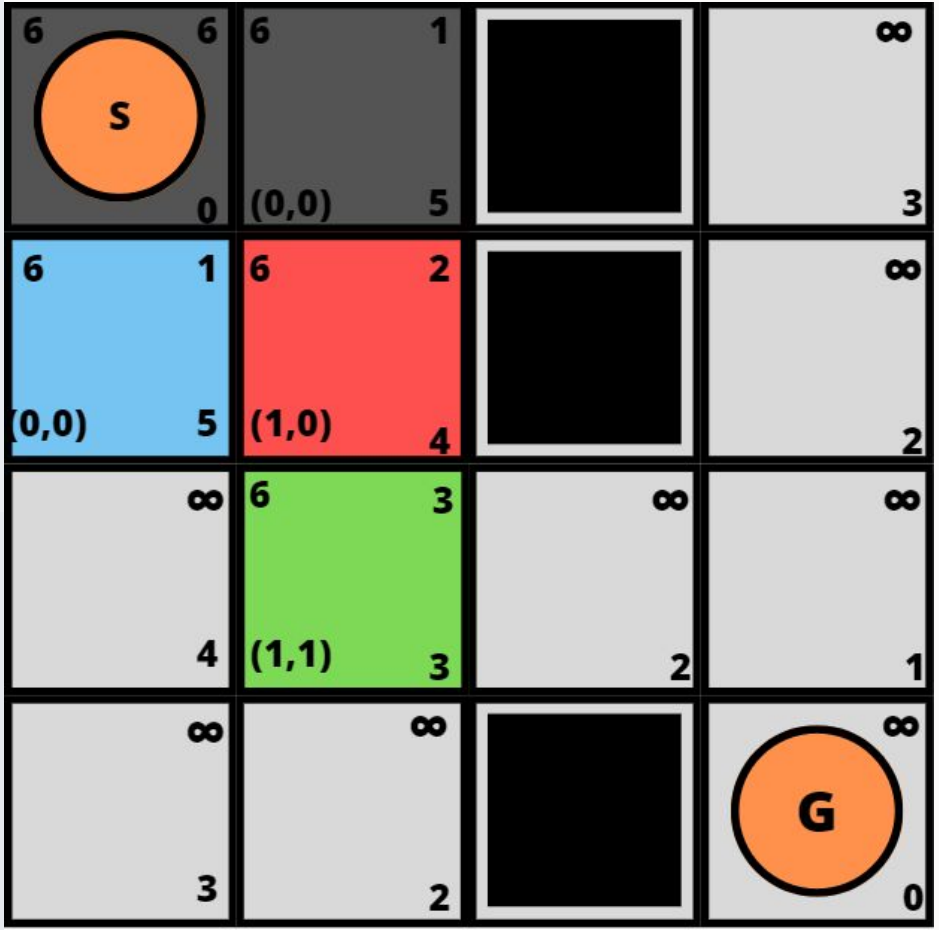
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



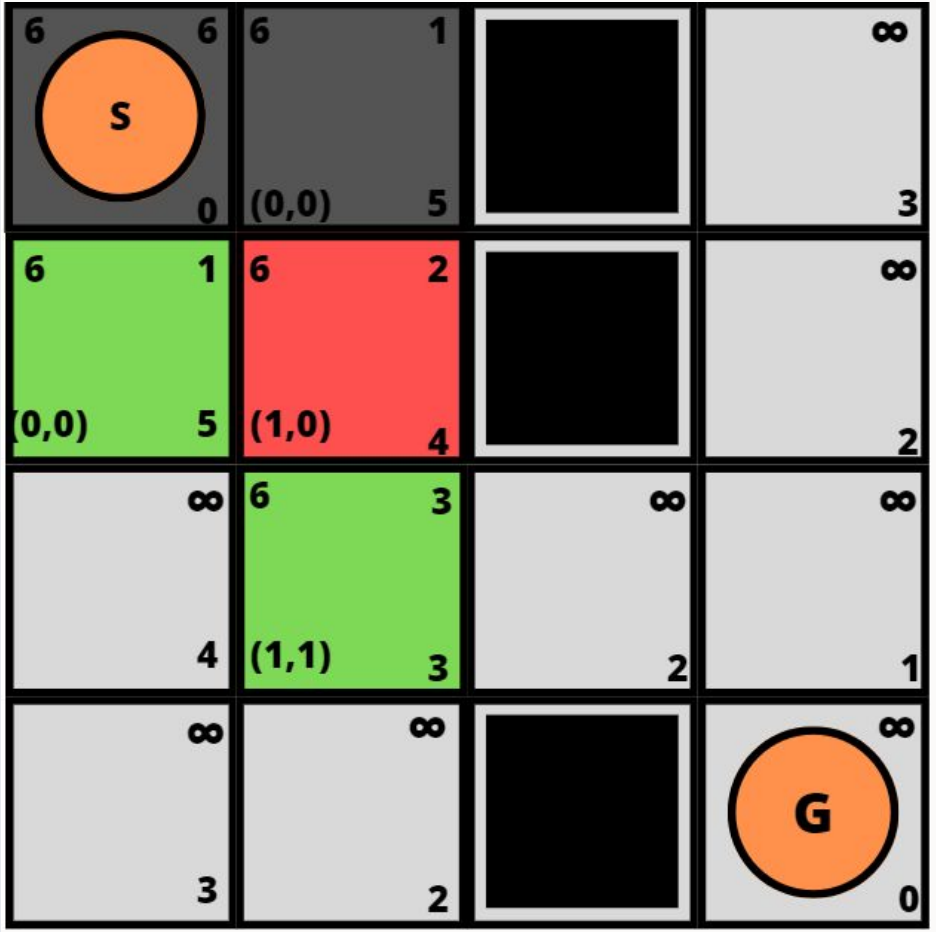
1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2 $Open \leftarrow \{s_0\}$
3 $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4 **while** $Open \neq \emptyset$
5 Extrae un u desde $Open$ con menor valor-f
6 **if** u es objetivo **return** u
7 **for each** $v \in Succ(u)$ **do**
1 $cost_v = g(u) + c(u, v)$
2 **if** $cost_v \geq g(v)$ **return**
3 $parent(v) \leftarrow u$
4 $g(v) \leftarrow cost_v$
5 $f(v) \leftarrow g(v) + h(v)$
6 **if** $v \in Open$ **then** Reordenar $Open$
7 **else** Insertar v en $Open$



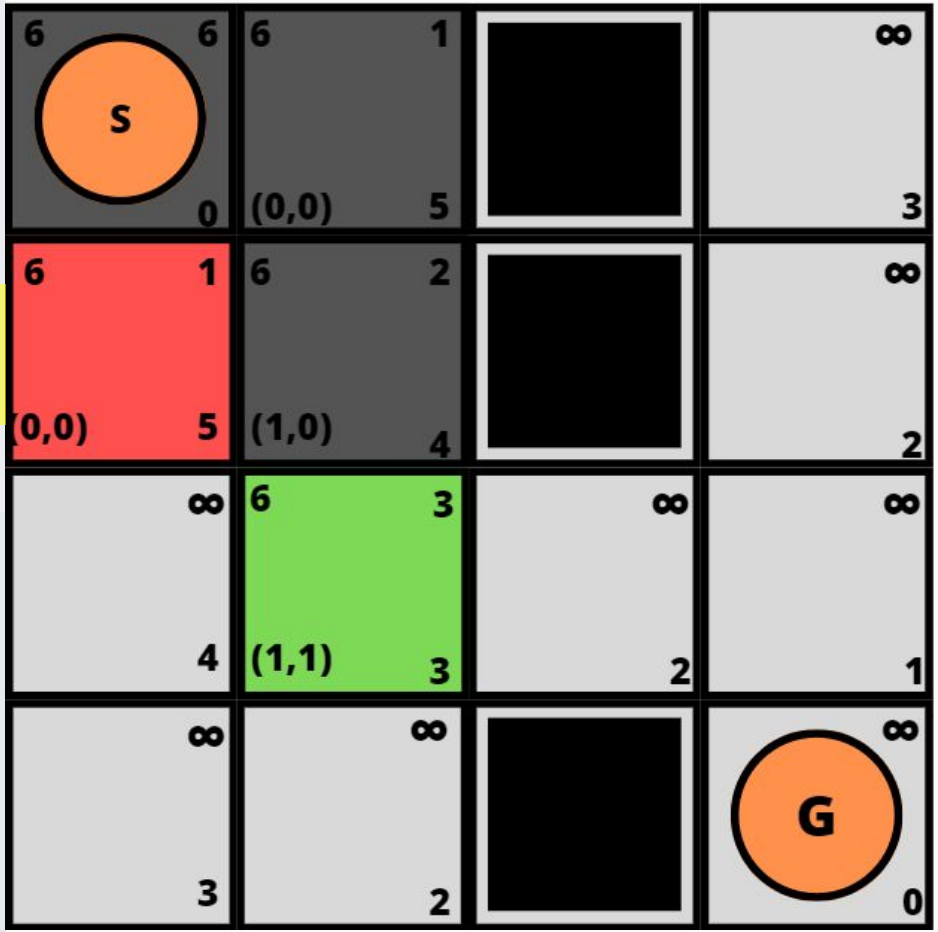
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



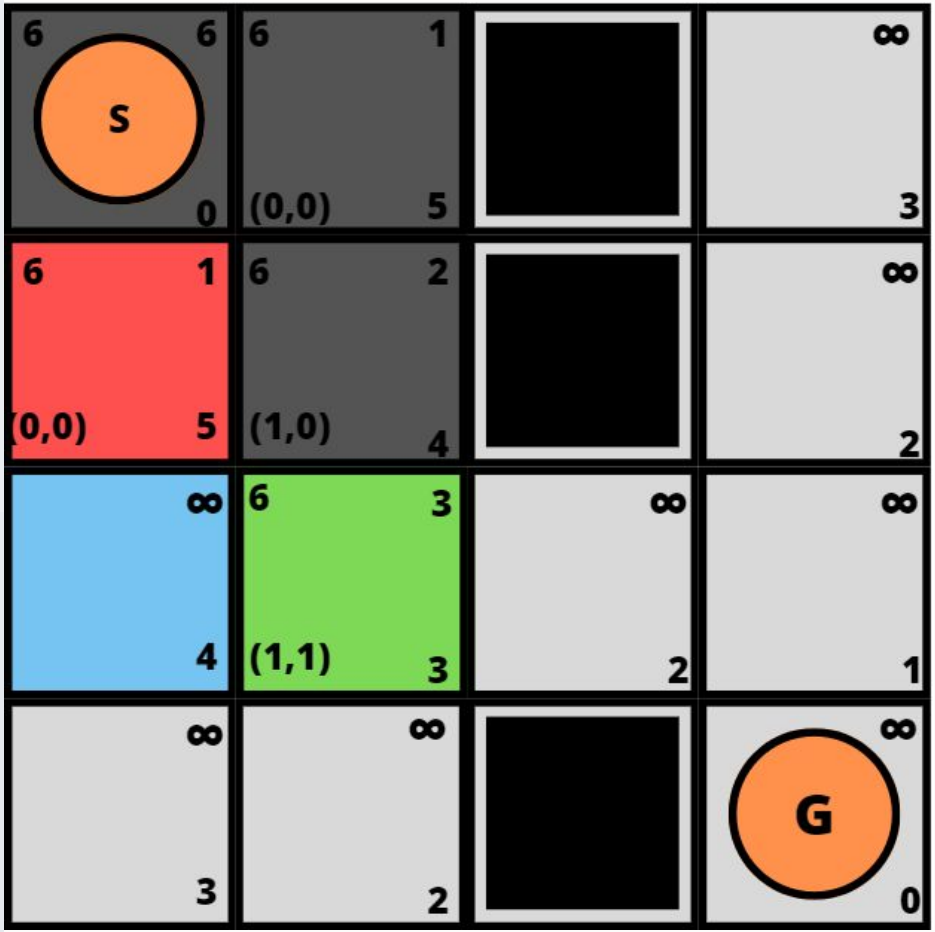
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



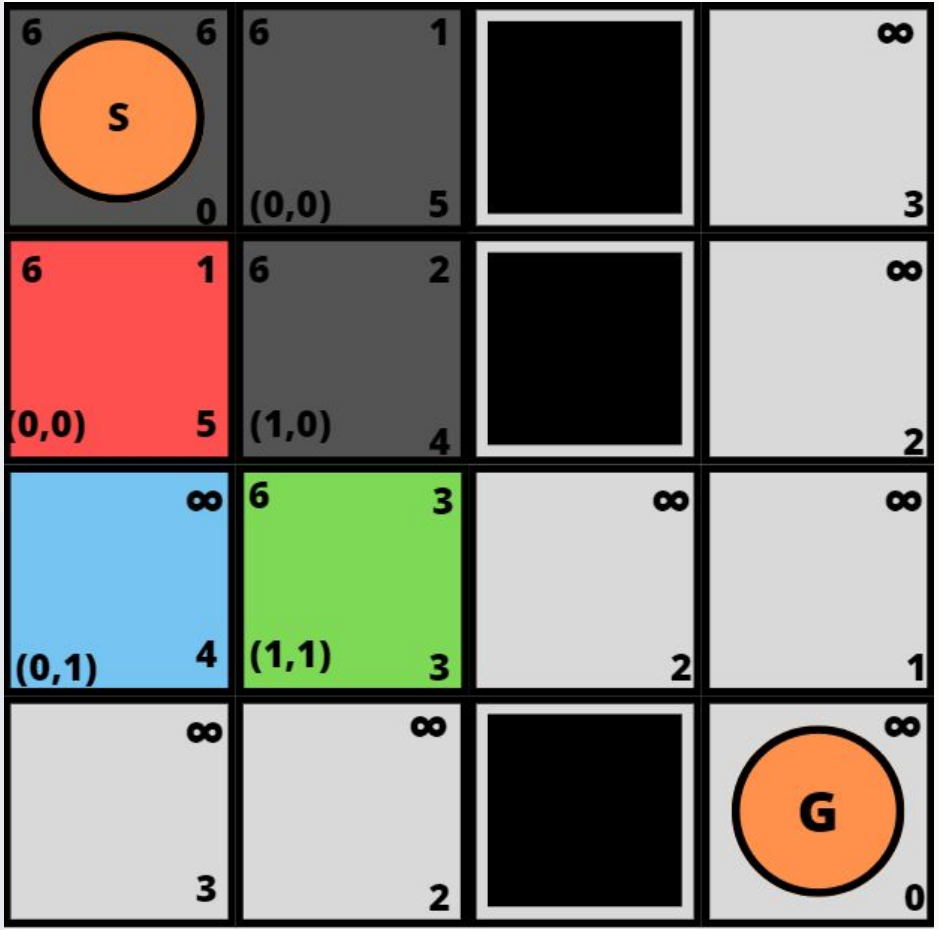
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



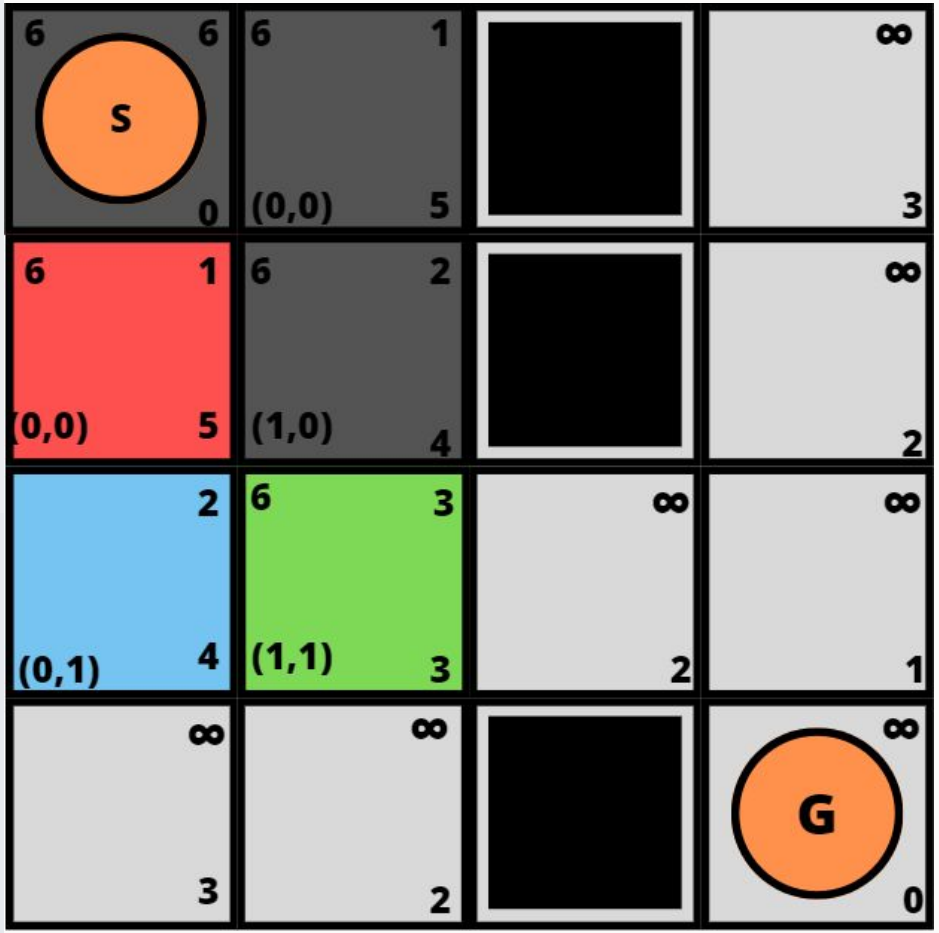
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



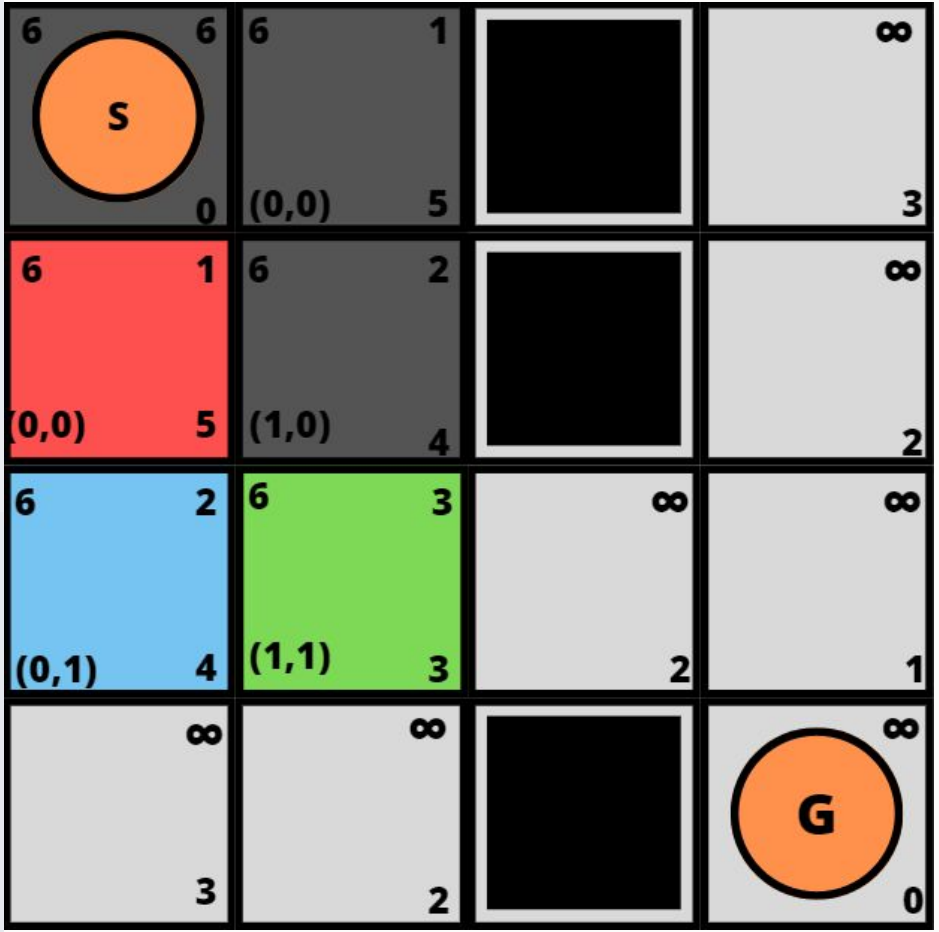
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



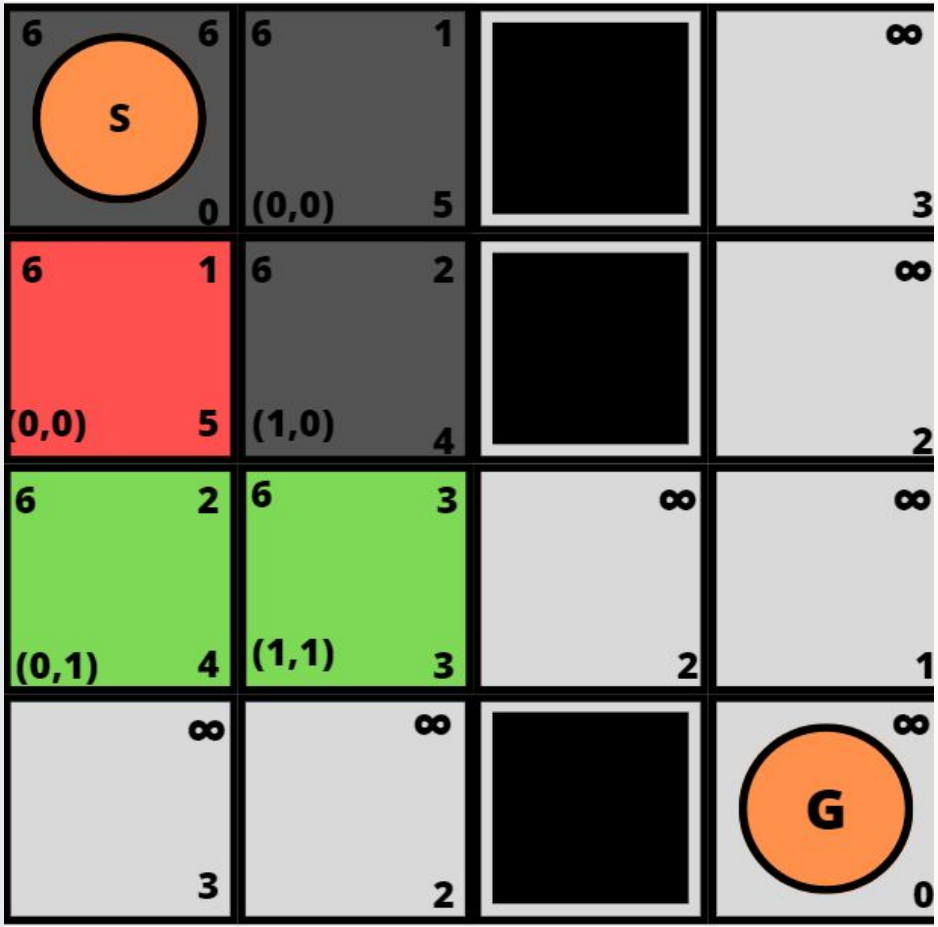
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



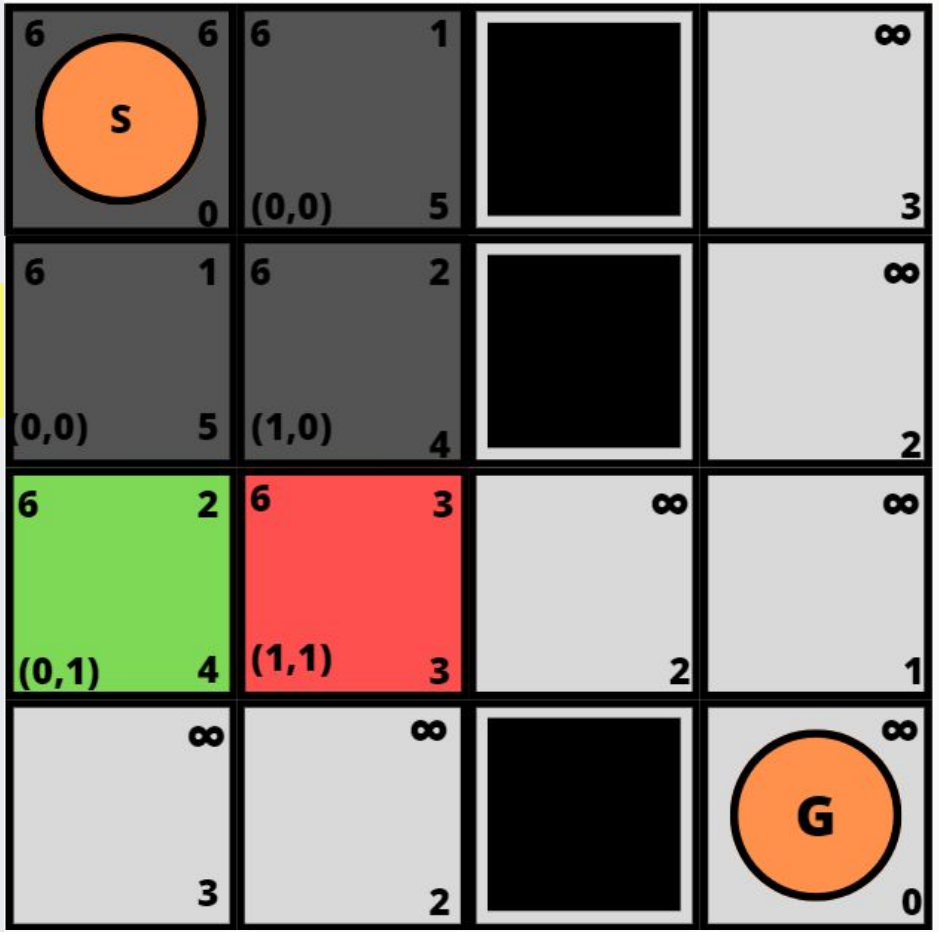
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



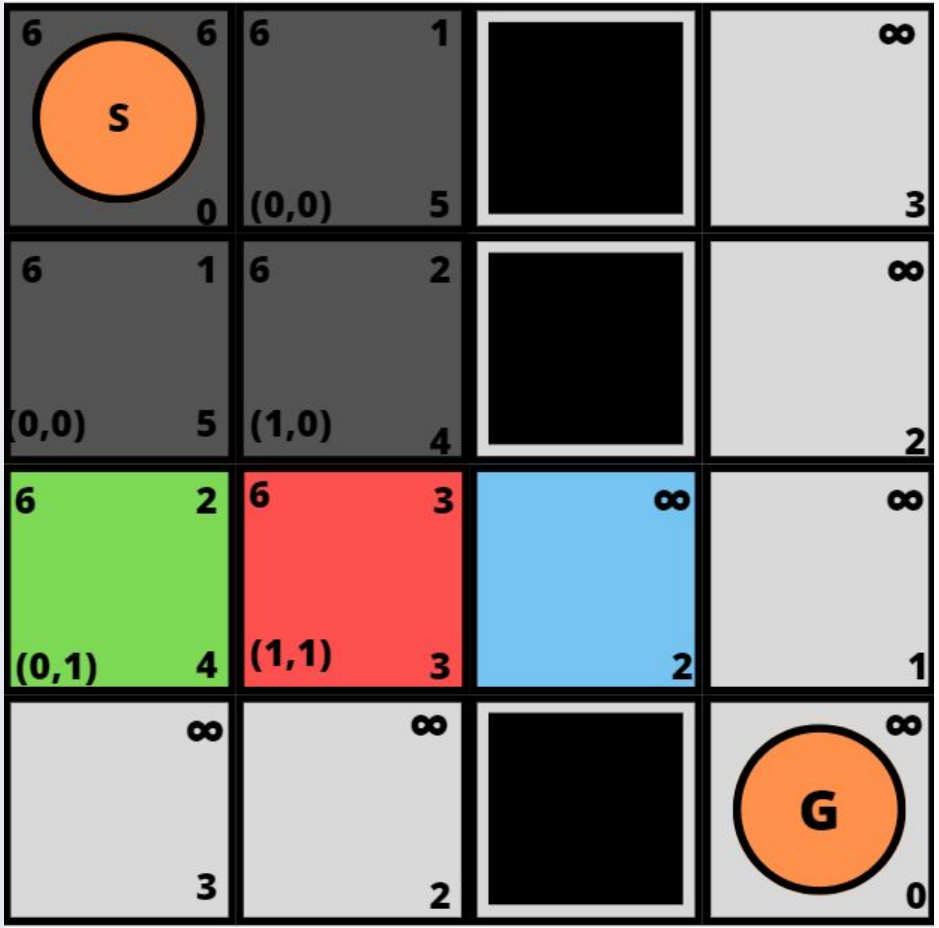
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



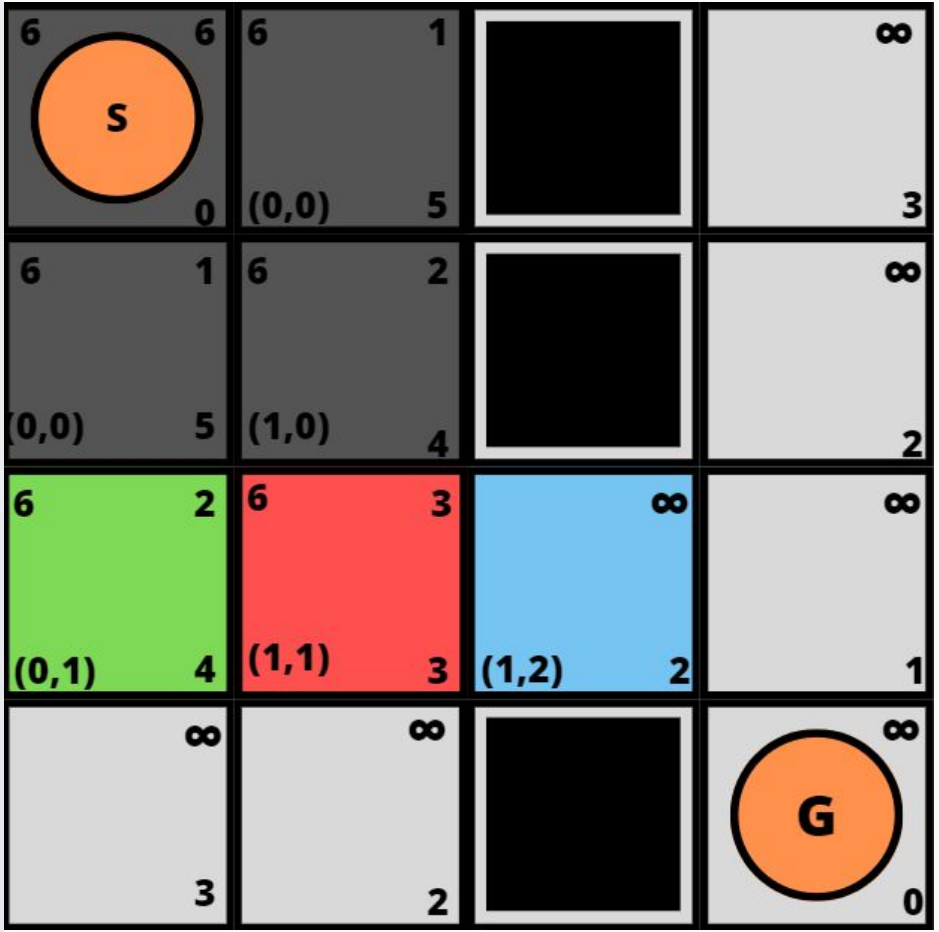
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



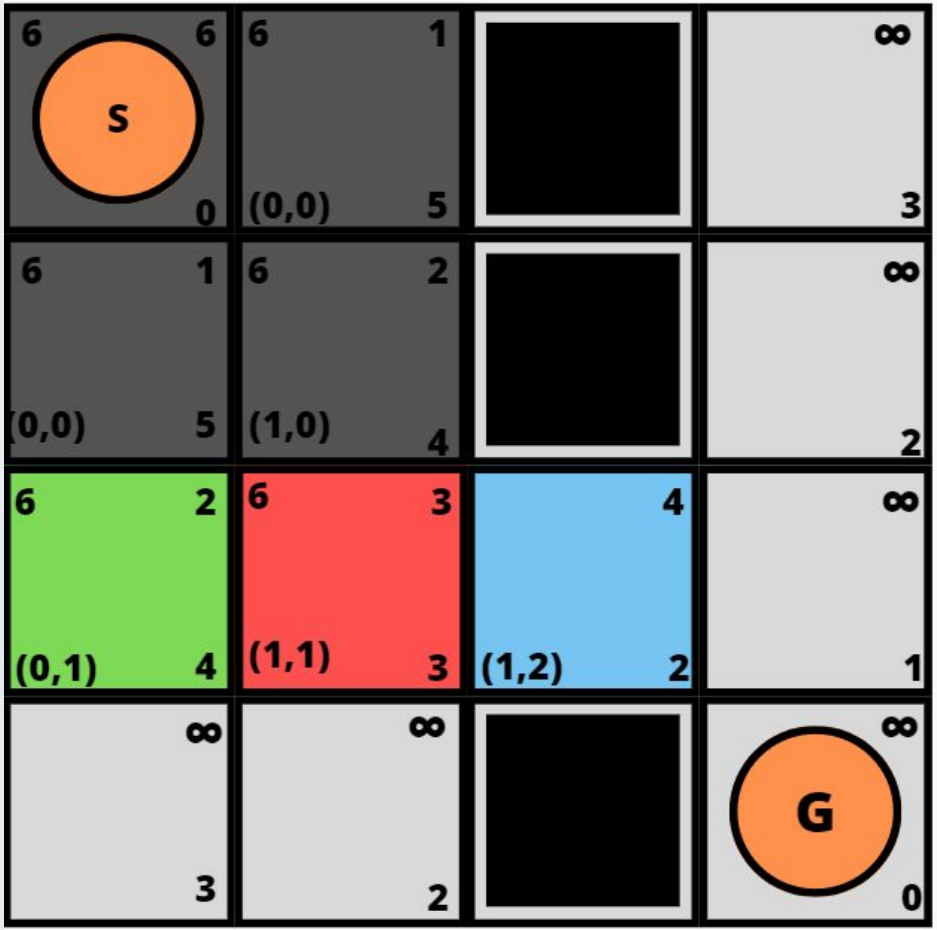
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
   1. $cost_v = g(u) + c(u, v)$
   2. **if** $cost_v \geq g(v)$ **return**
   3. $parent(v) \leftarrow u$
   4. $g(v) \leftarrow cost_v$
   5. $f(v) \leftarrow g(v) + h(v)$
   6. **if** $v \in Open$ **then** Reordenar $Open$
   7. **else** Insertar v en $Open$



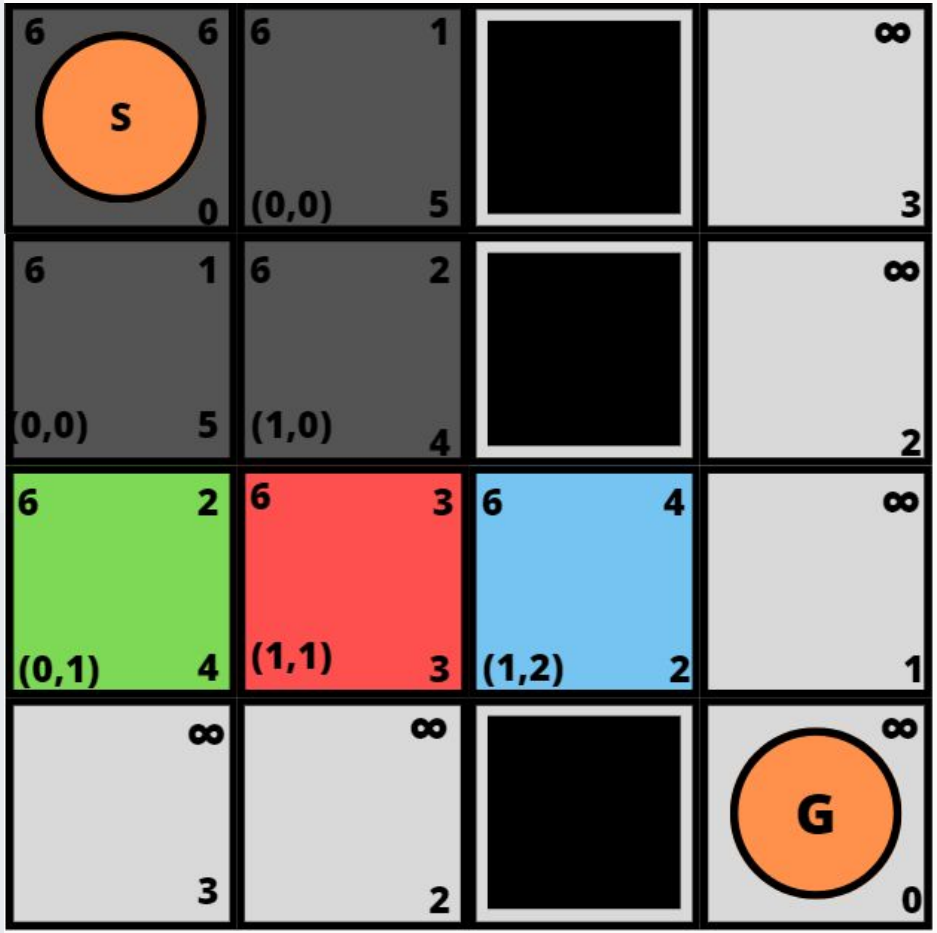
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



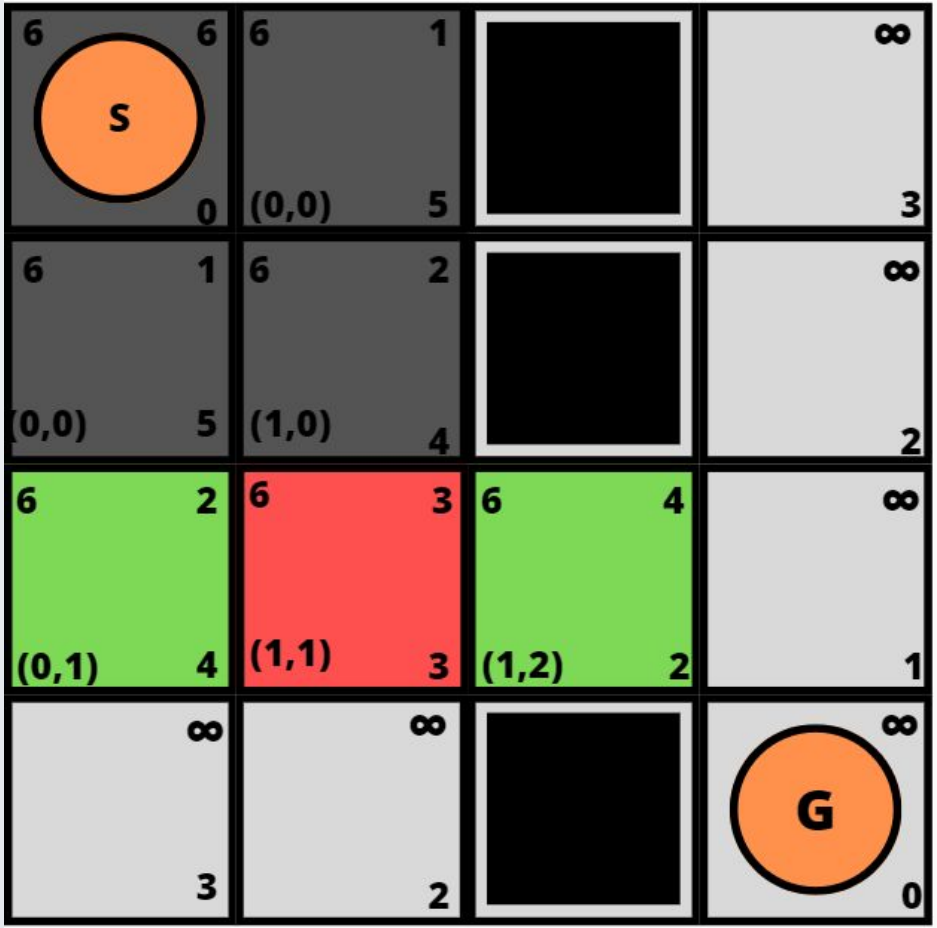
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



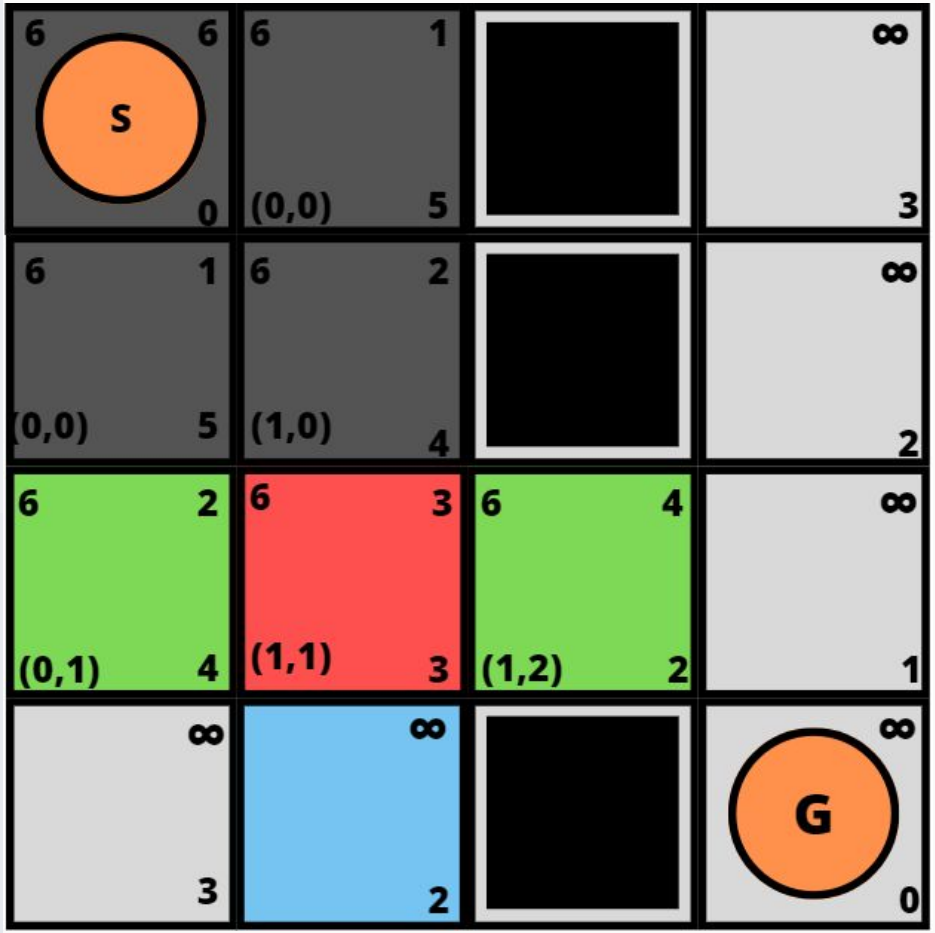
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



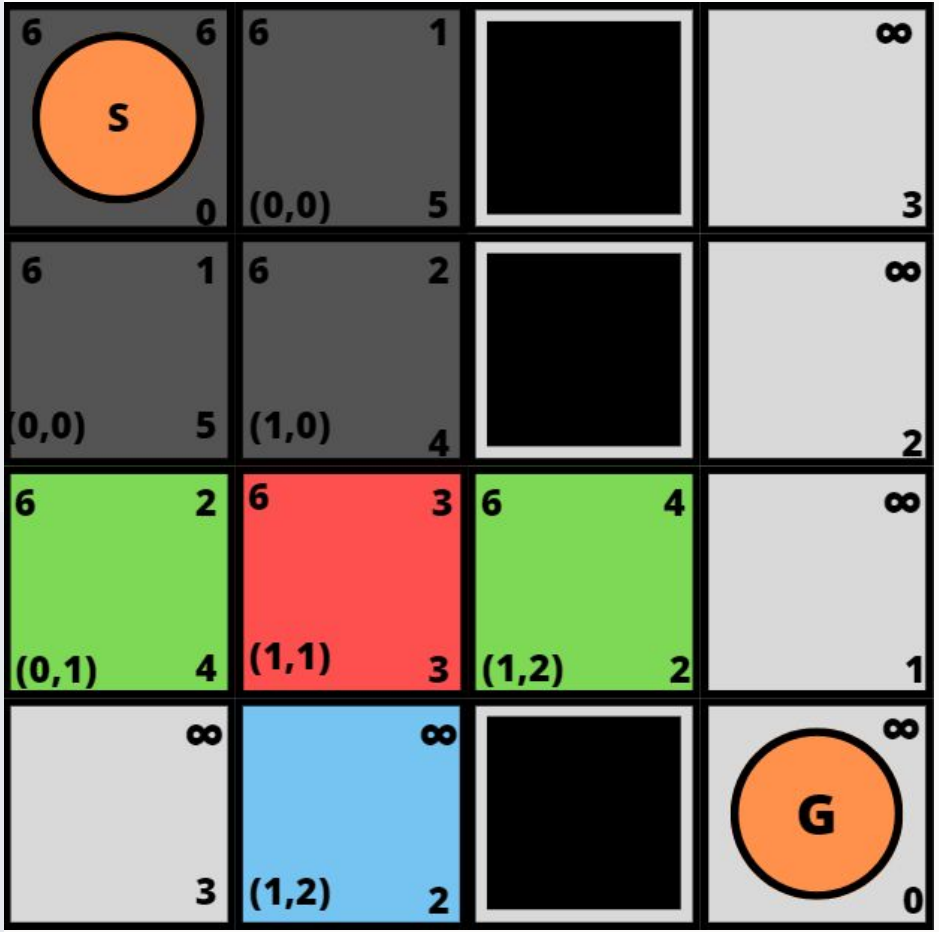
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



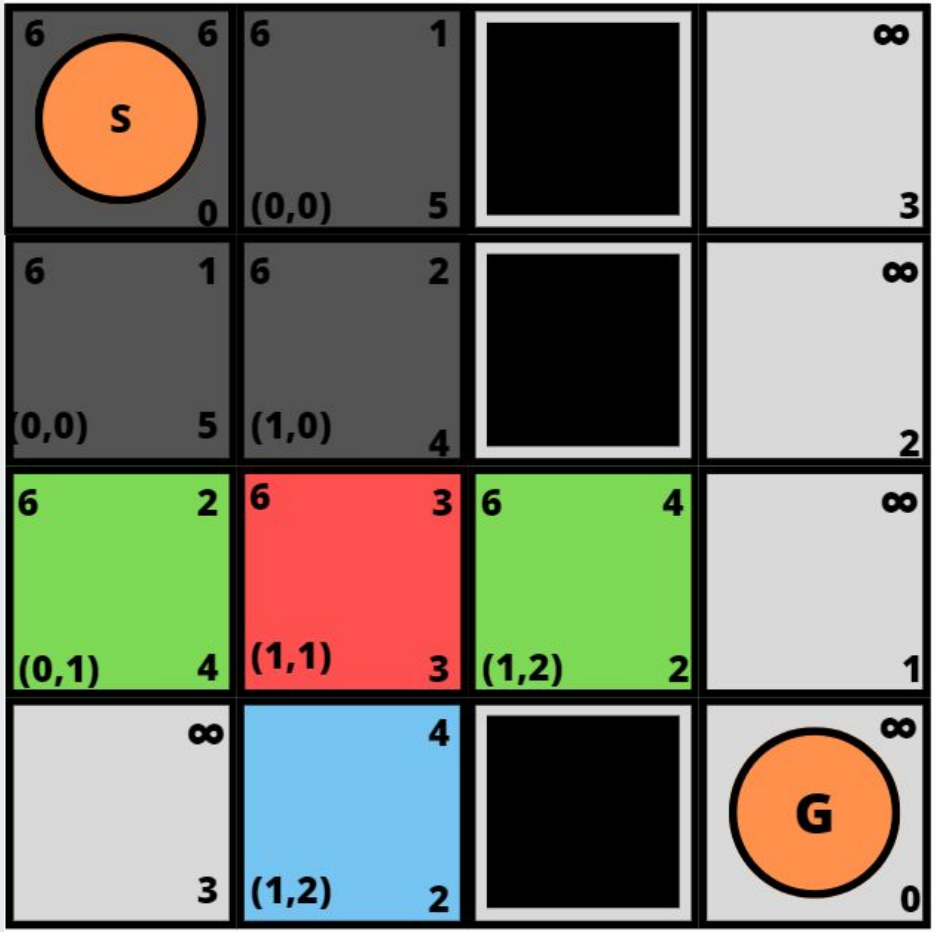
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



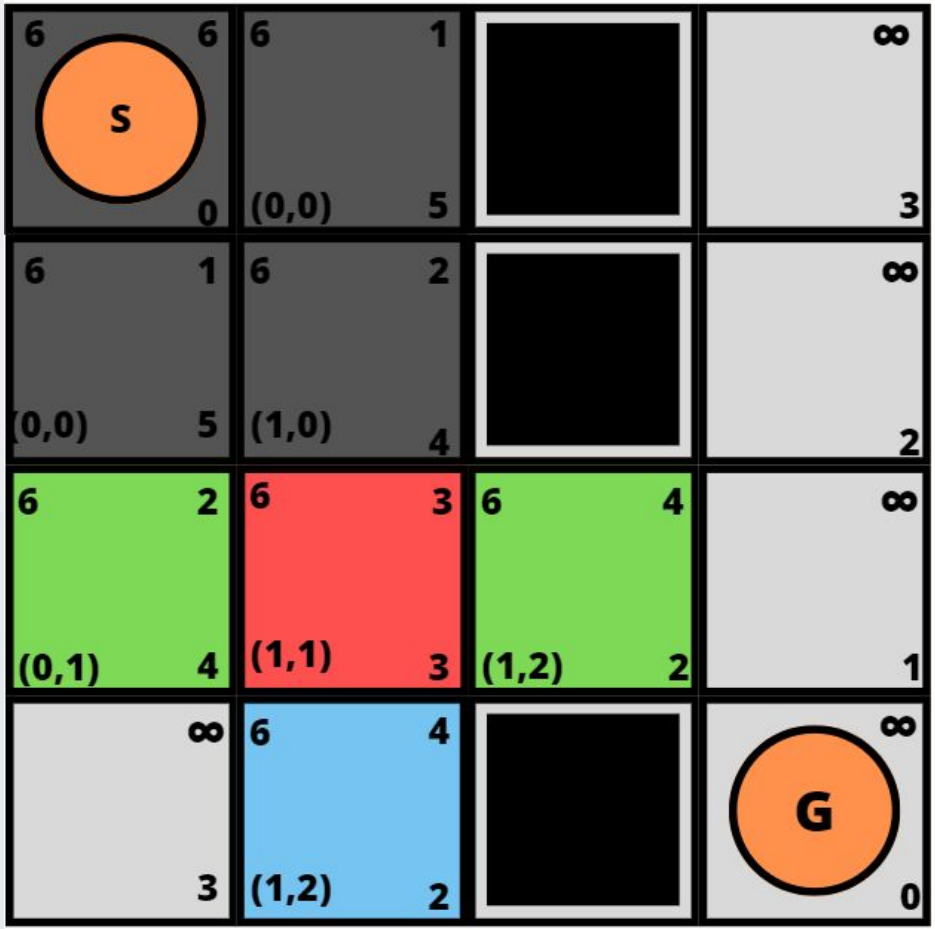
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



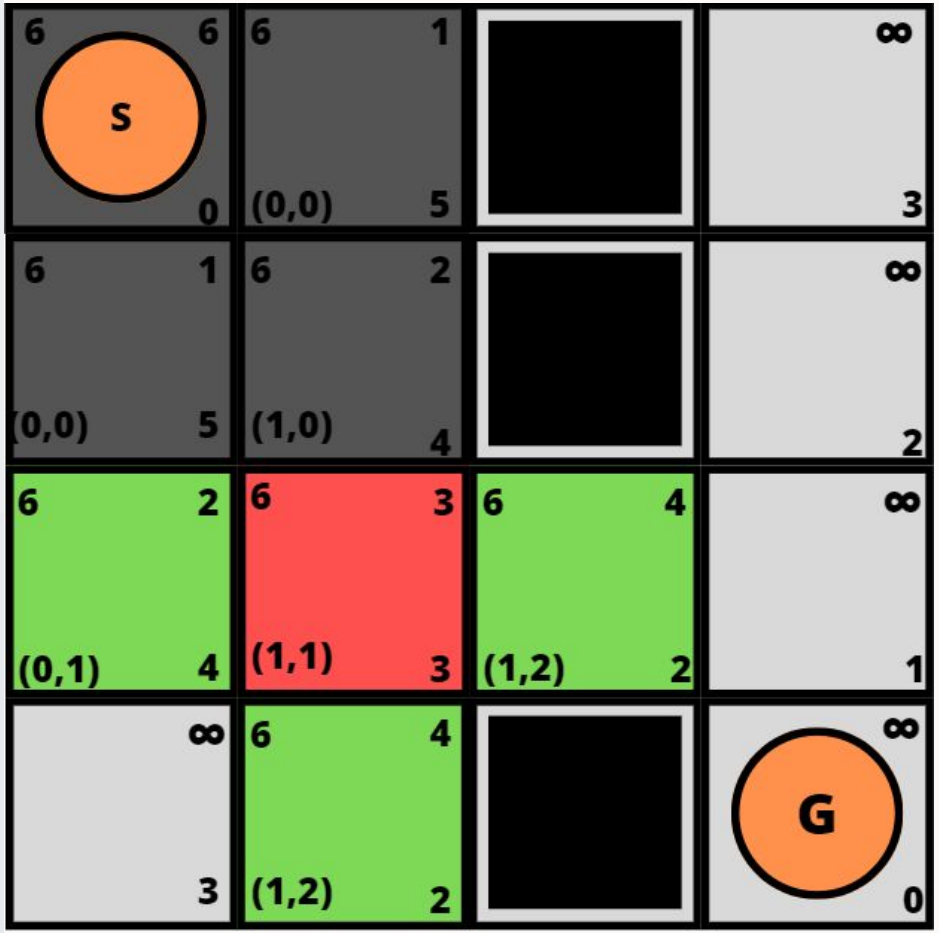
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



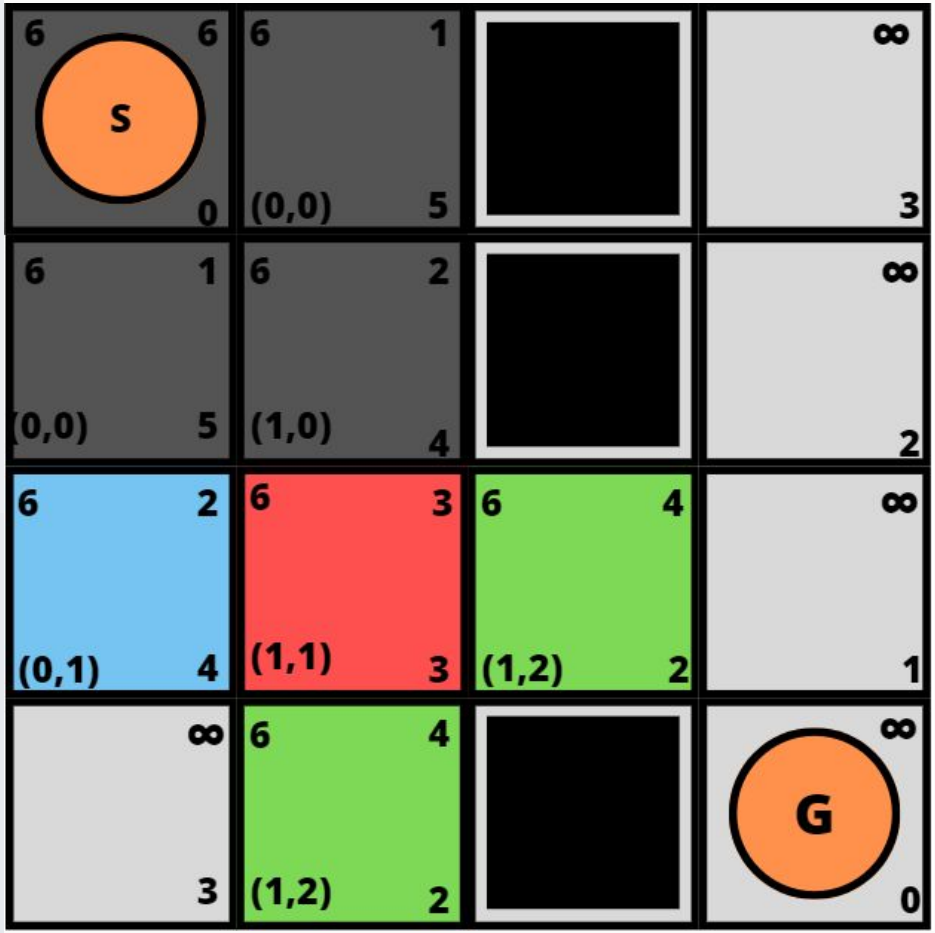
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



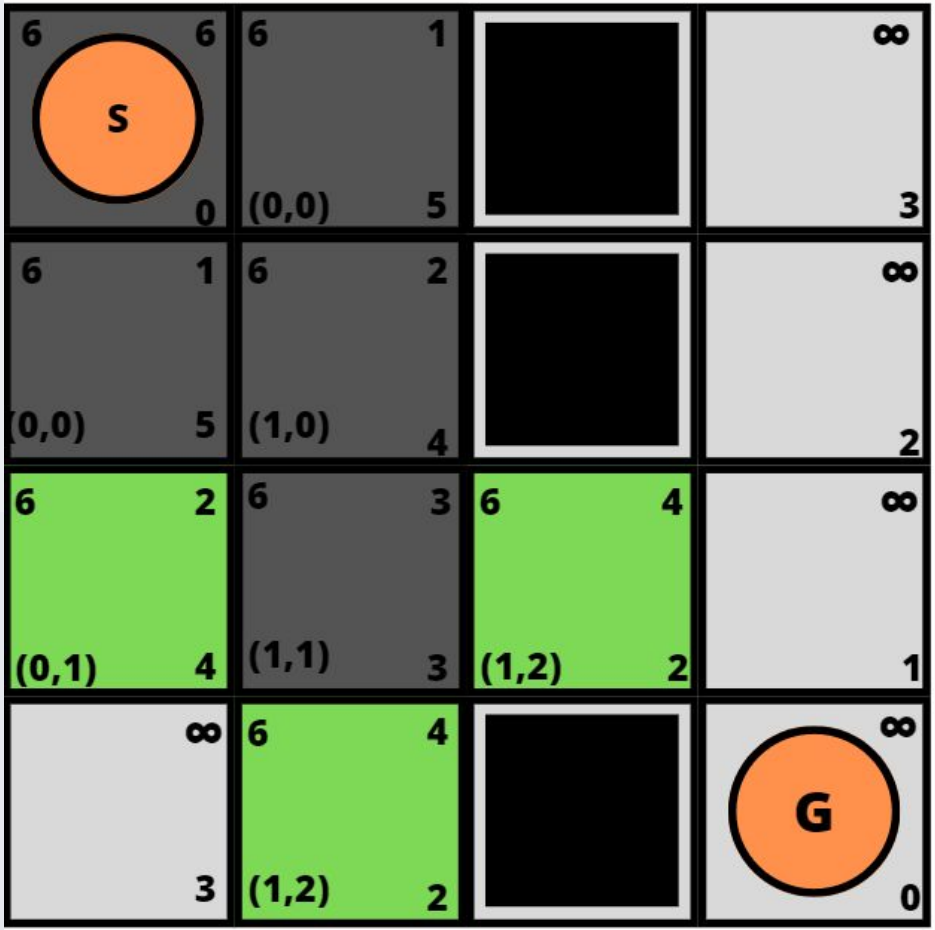
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



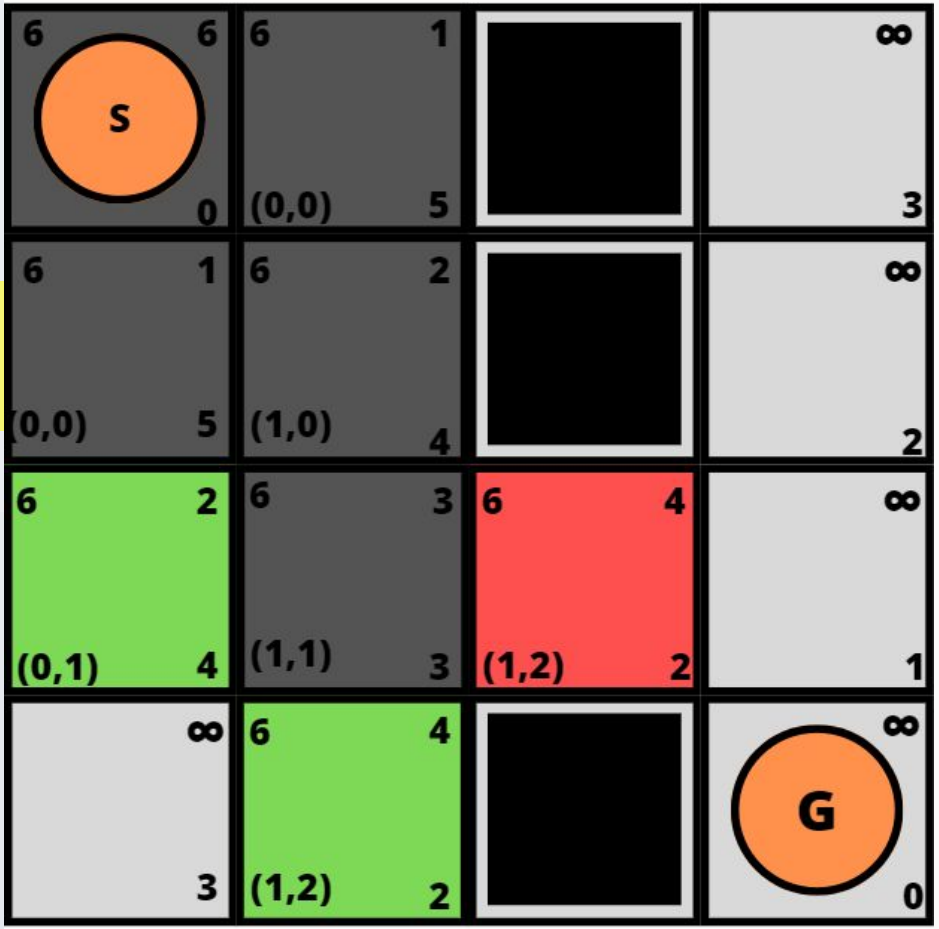
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
   1. $cost_v = g(u) + c(u, v)$
   2. **if** $cost_v \geq g(v)$ **return**
   3. $parent(v) \leftarrow u$
   4. $g(v) \leftarrow cost_v$
   5. $f(v) \leftarrow g(v) + h(v)$
   6. **if** $v \in Open$ **then** Reordenar $Open$
   7. **else** Insertar v en $Open$



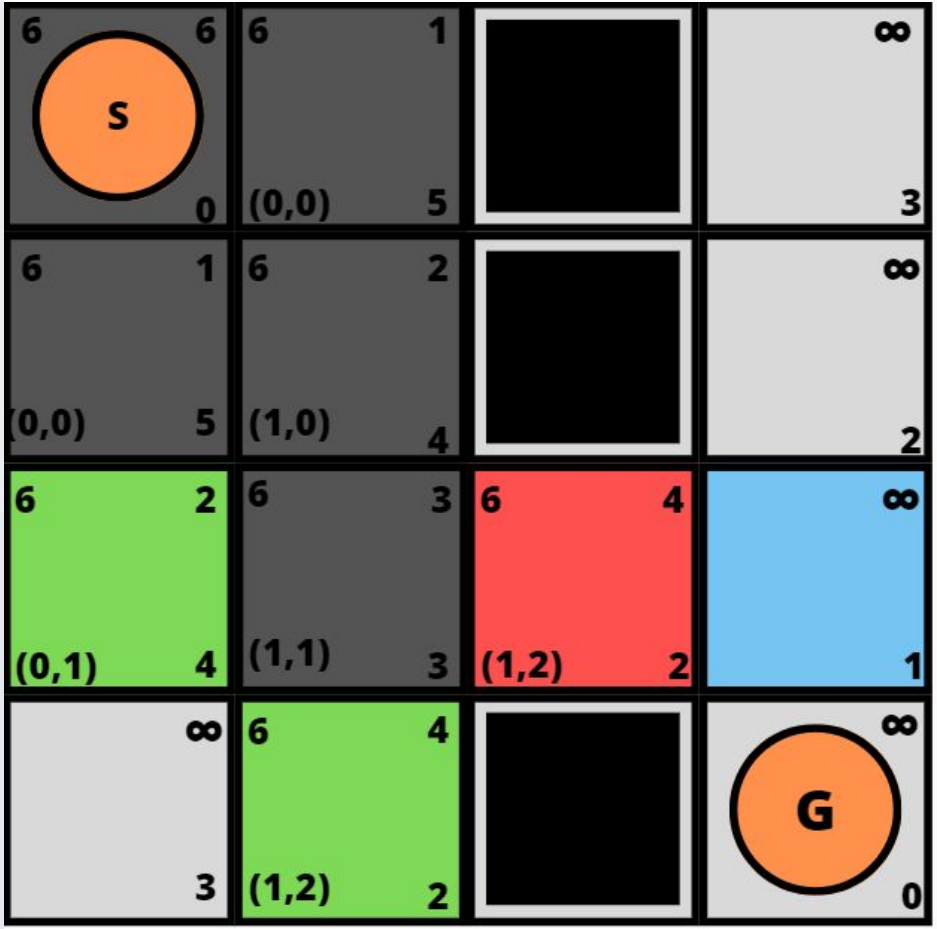
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



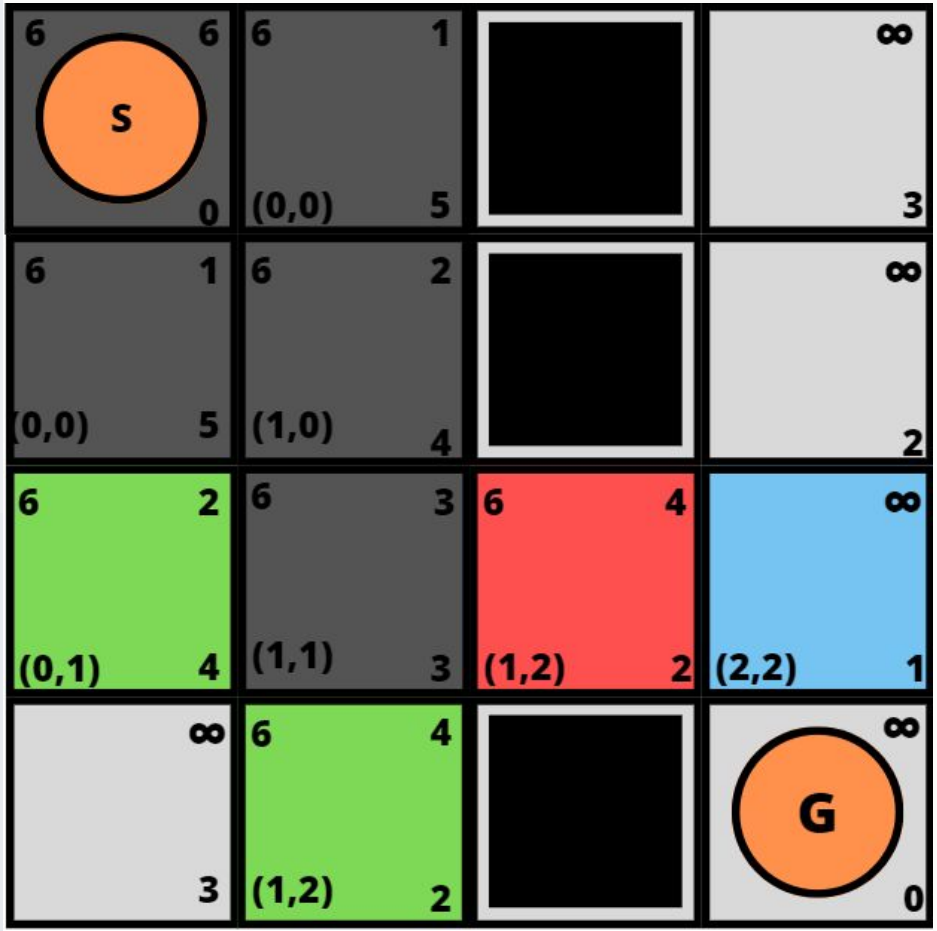
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



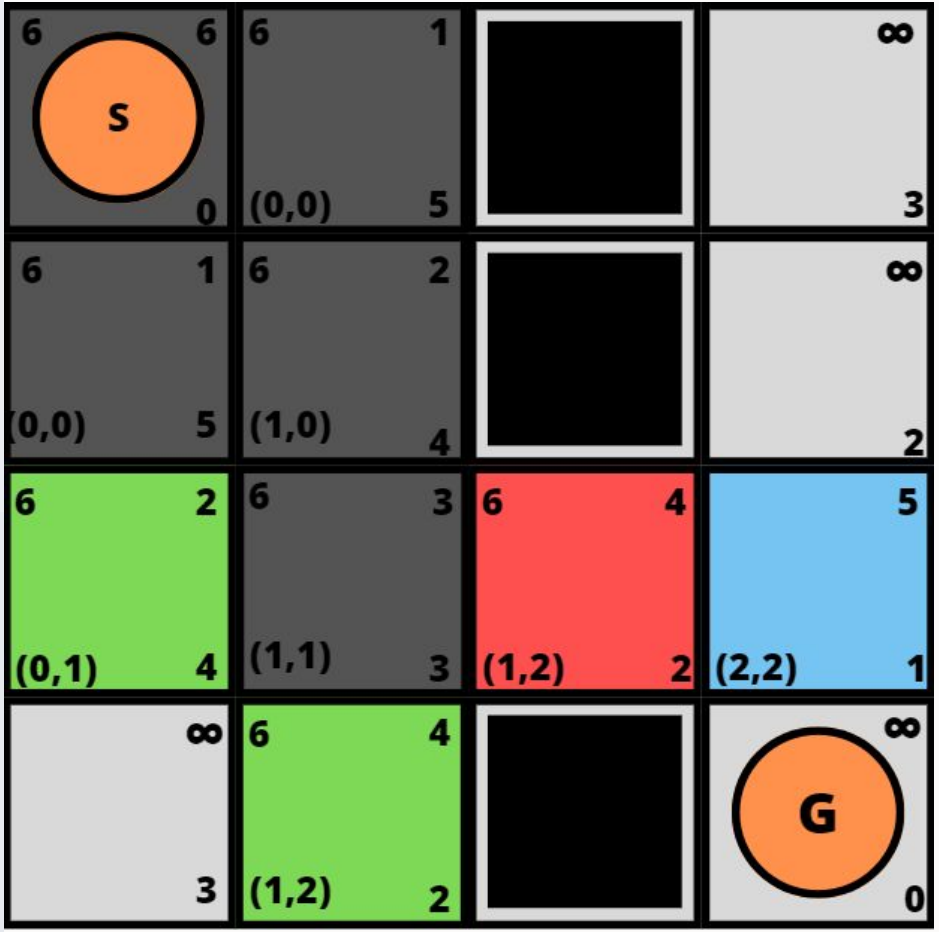
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
   1. $cost_v = g(u) + c(u, v)$
   2. **if** $cost_v \geq g(v)$ **return**
   3. $parent(v) \leftarrow u$
   4. $g(v) \leftarrow cost_v$
   5. $f(v) \leftarrow g(v) + h(v)$
   6. **if** $v \in Open$ **then** Reordenar $Open$
   7. **else** Insertar v en $Open$



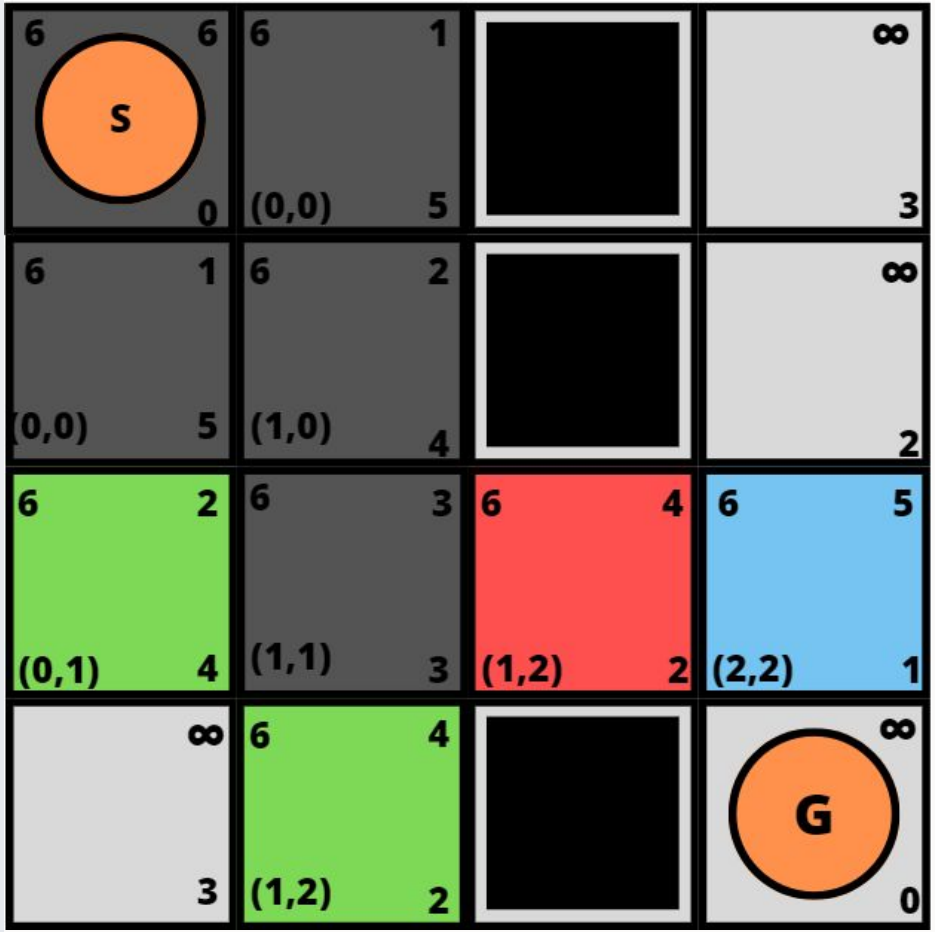
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



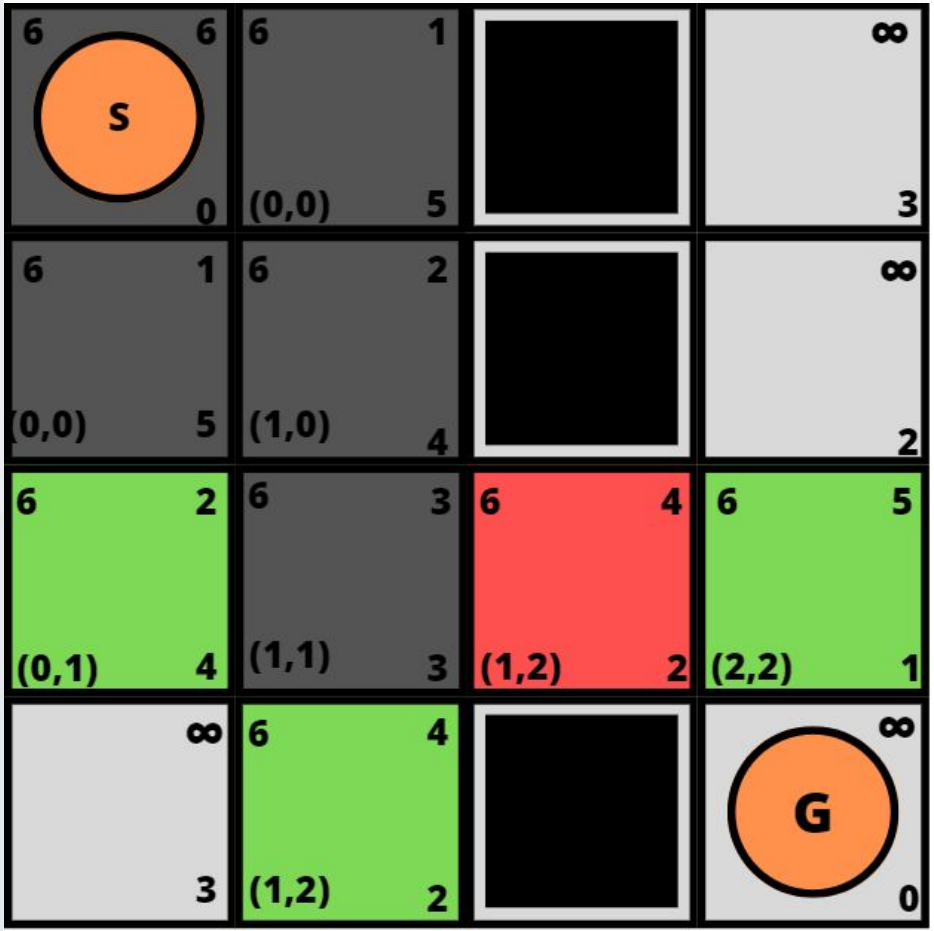
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$

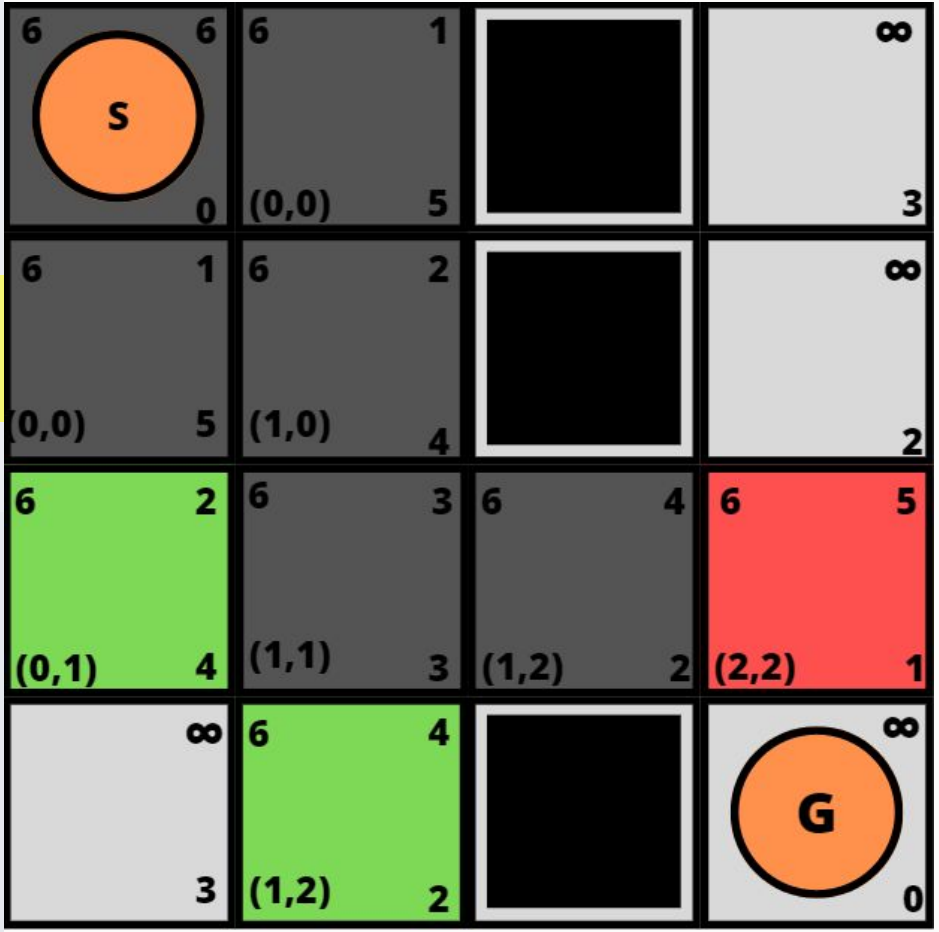
4 **while** $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor-f

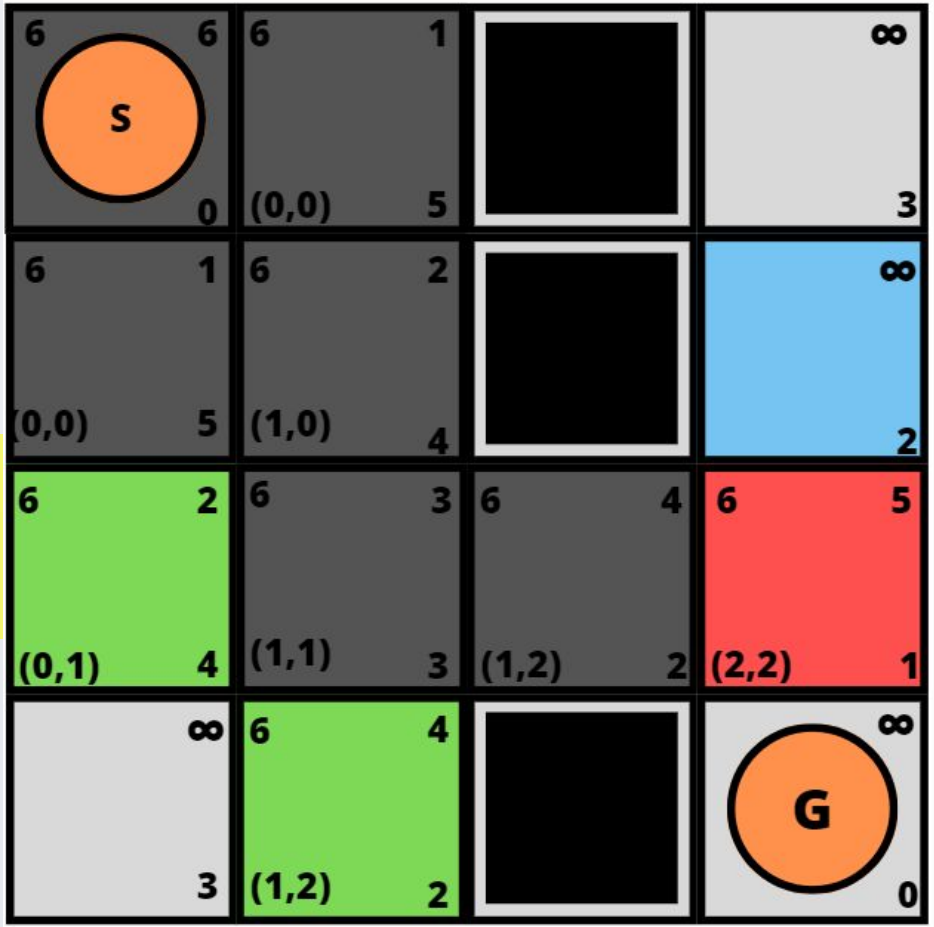
6 **if** u es objetivo **return** u

7 **for each** $v \in Succ(u)$ **do**

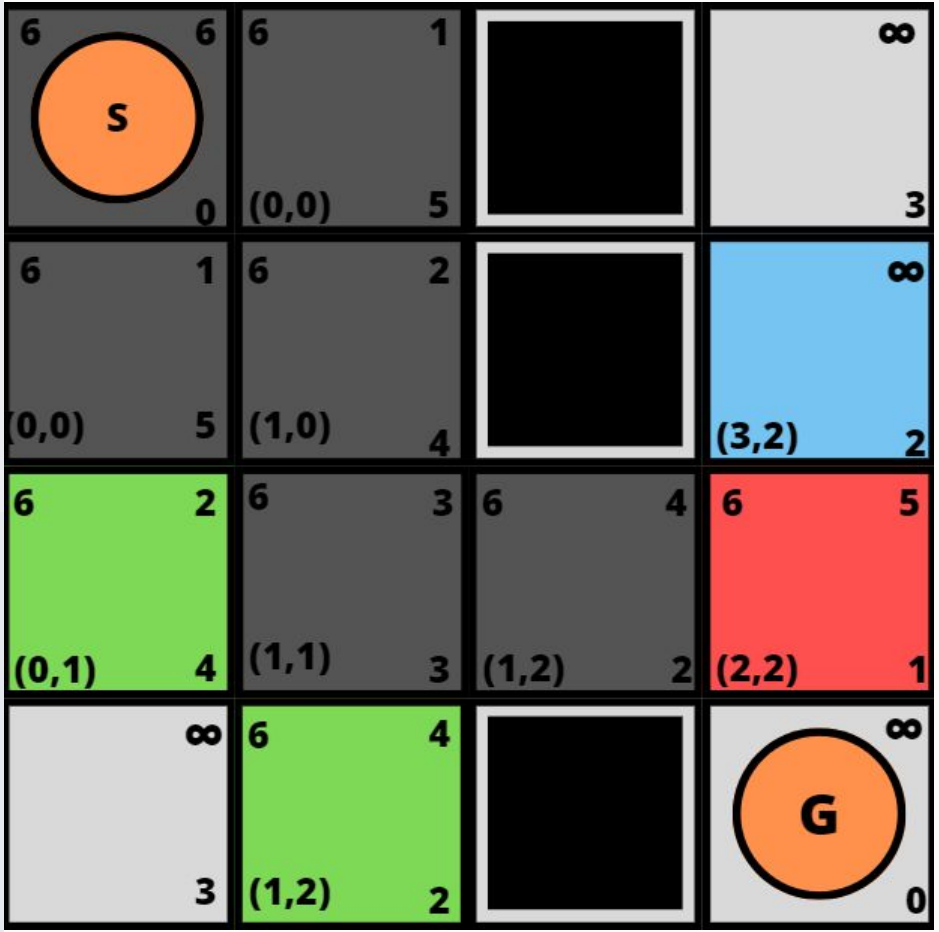
1. $cost_v = g(u) + c(u, v)$
2. **if** $cost_v \geq g(v)$ **return**
3. $parent(v) \leftarrow u$
4. $g(v) \leftarrow cost_v$
5. $f(v) \leftarrow g(v) + h(v)$
6. **if** $v \in Open$ **then** Reordenar $Open$
7. **else** Insertar v en $Open$



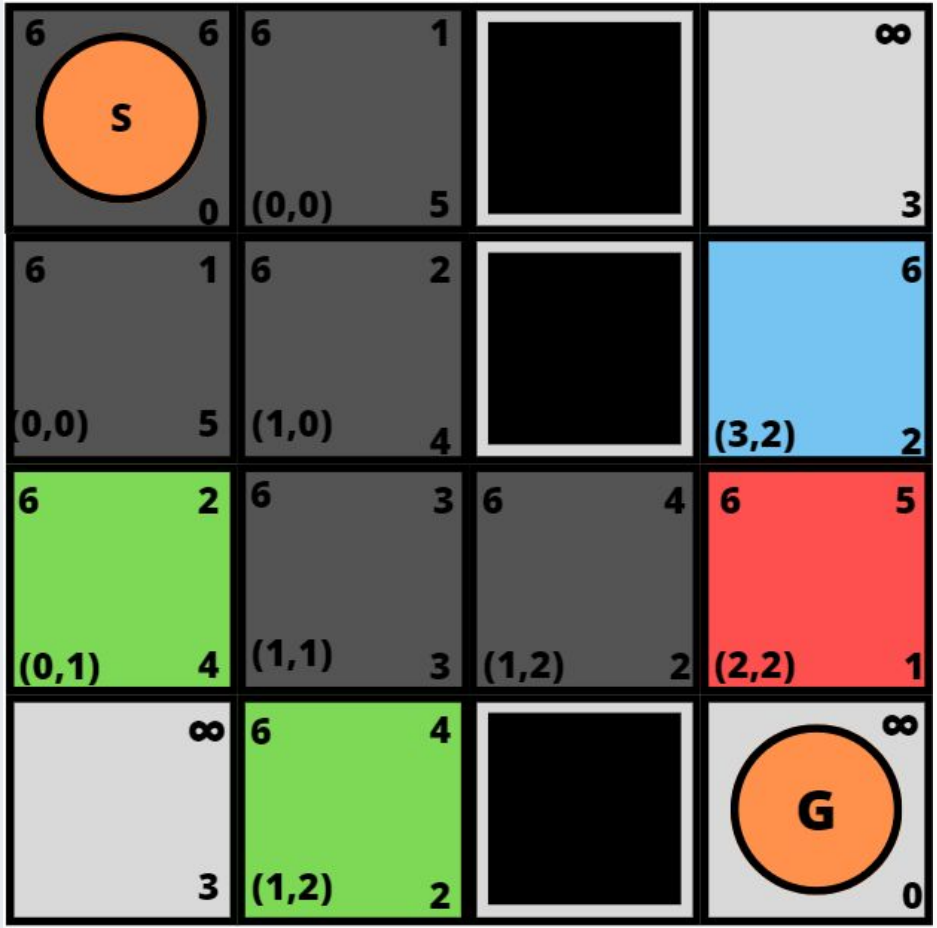
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
   1. $cost_v = g(u) + c(u, v)$
   2. **if** $cost_v \geq g(v)$ **return**
   3. $parent(v) \leftarrow u$
   4. $g(v) \leftarrow cost_v$
   5. $f(v) \leftarrow g(v) + h(v)$
   6. **if** $v \in Open$ **then** Reordenar $Open$
   7. **else** Insertar v en $Open$



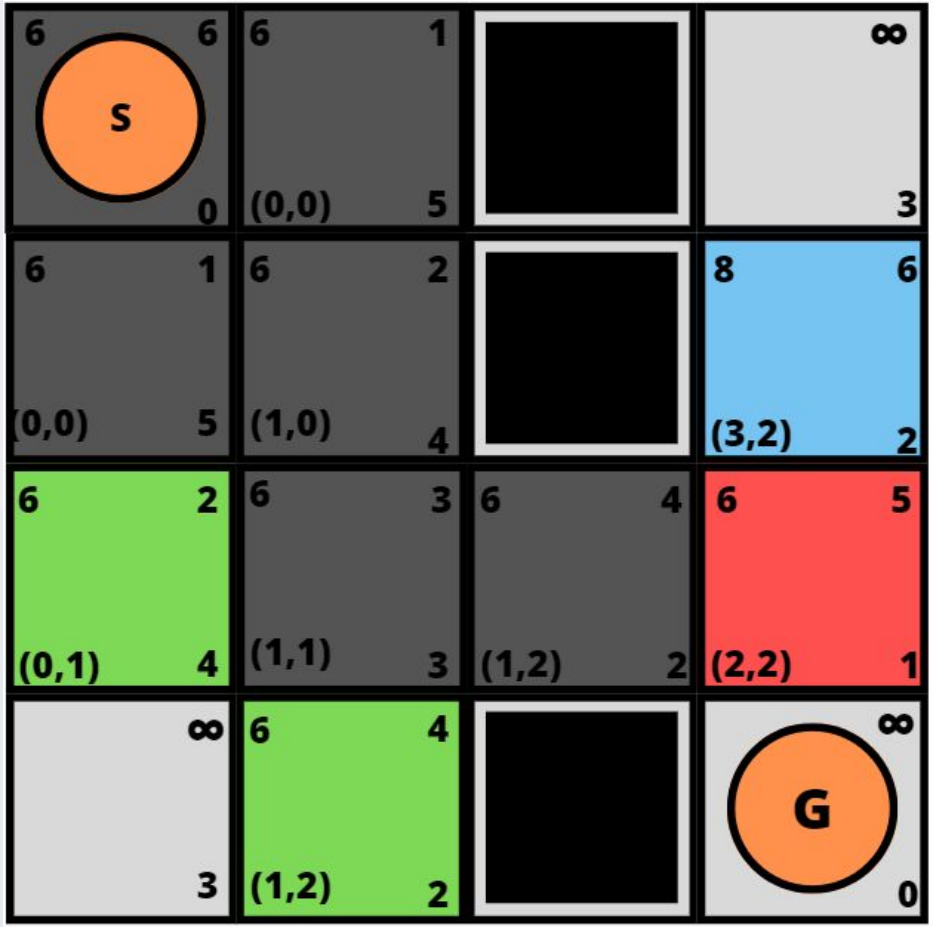
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



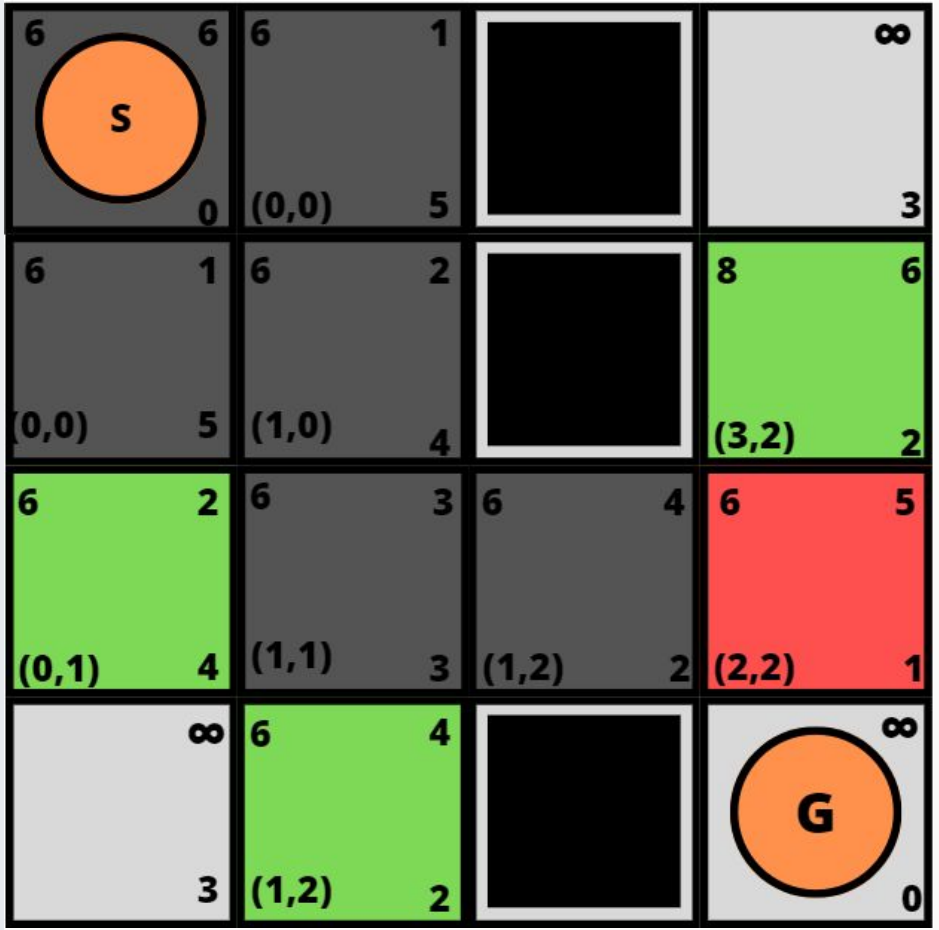
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



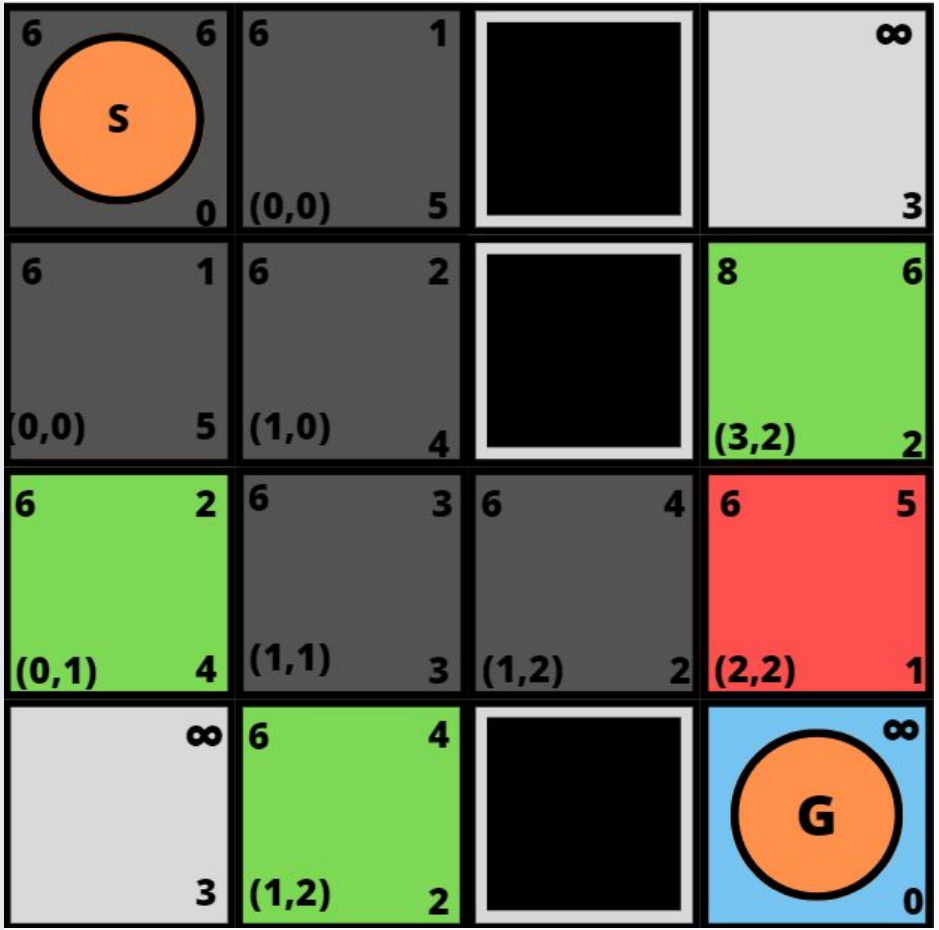
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



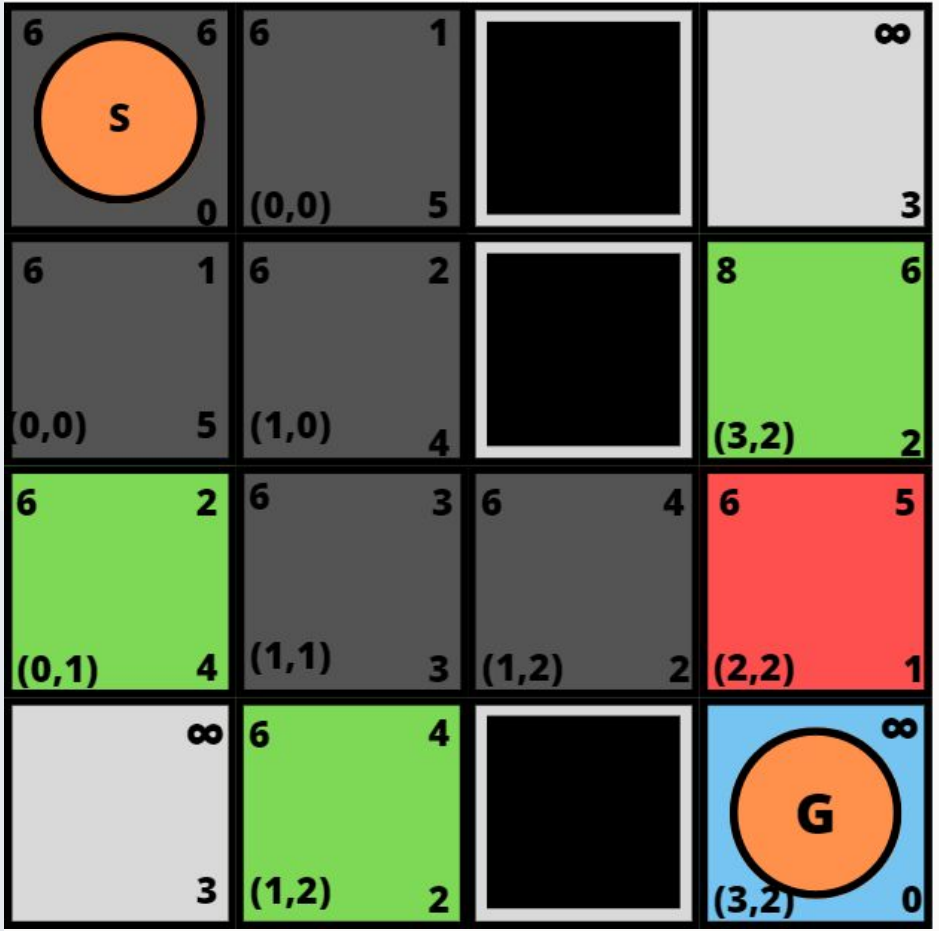
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$

4 **while** $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor-f

6 **if** u es objetivo **return** u

7 **for each** $v \in Succ(u)$ **do**

1 $cost_v = g(u) + c(u, v)$

2 **if** $cost_v \geq g(v)$ **return**

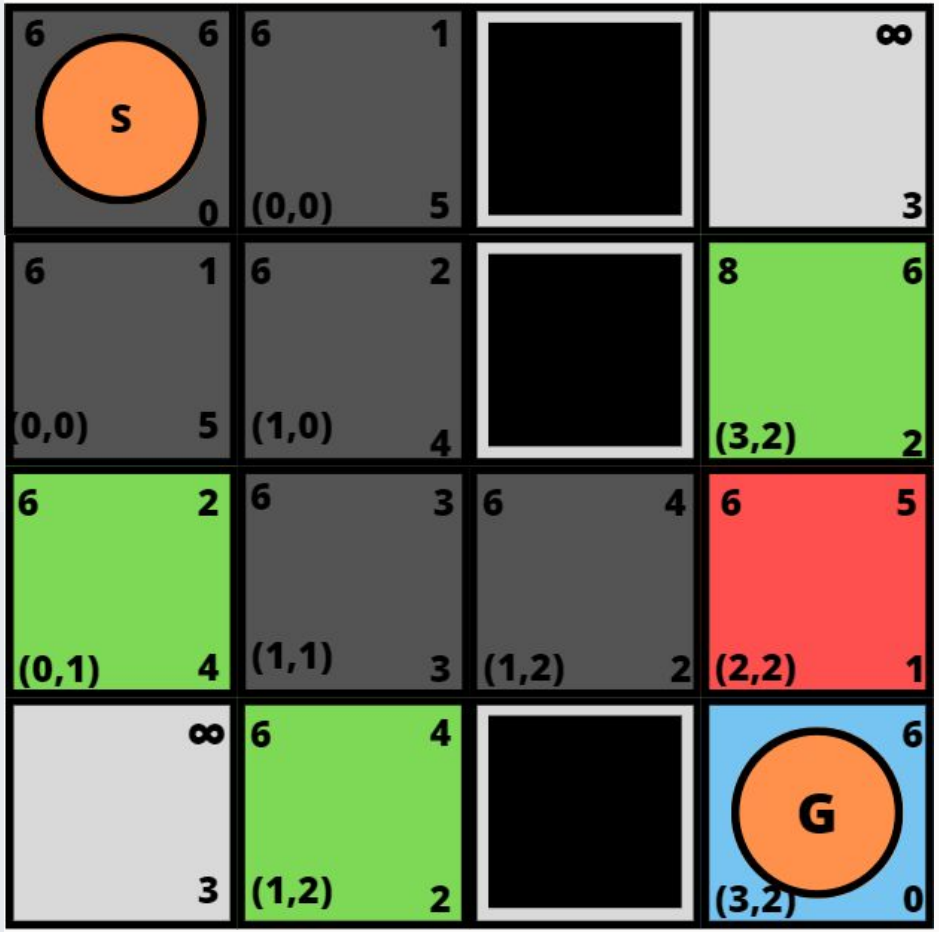
3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

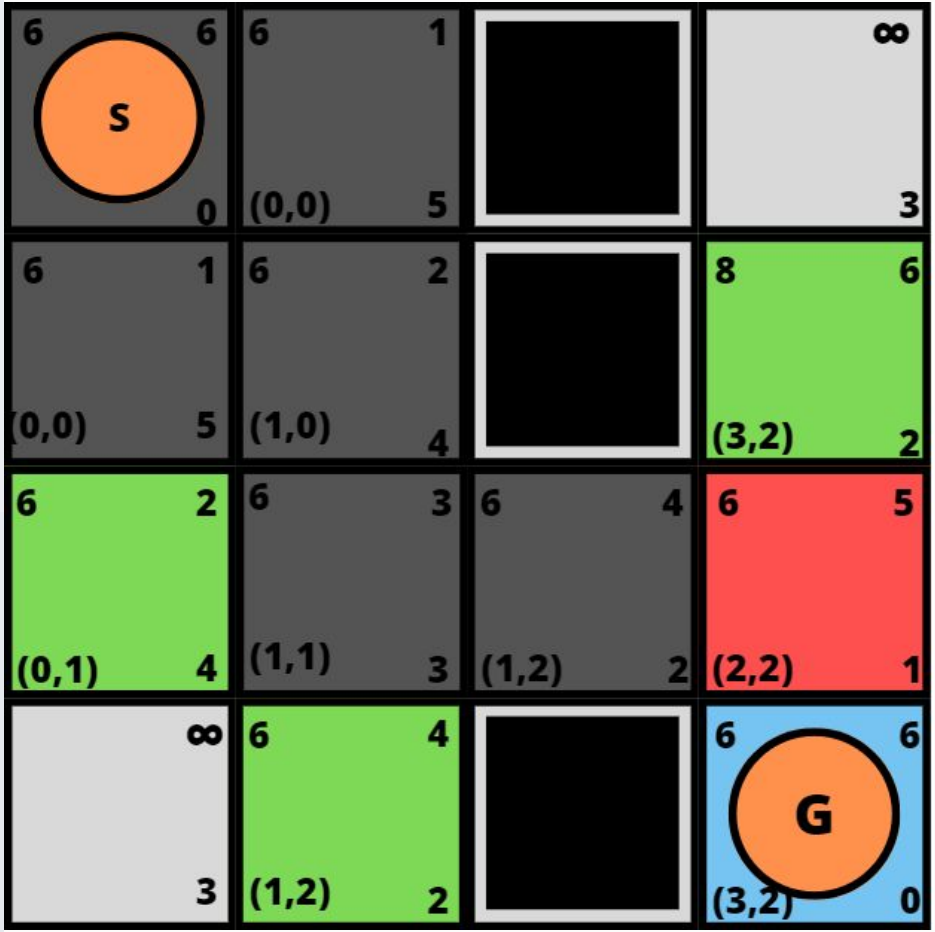
5 $f(v) \leftarrow g(v) + h(v)$

6 **if** $v \in Open$ **then** Reordenar $Open$

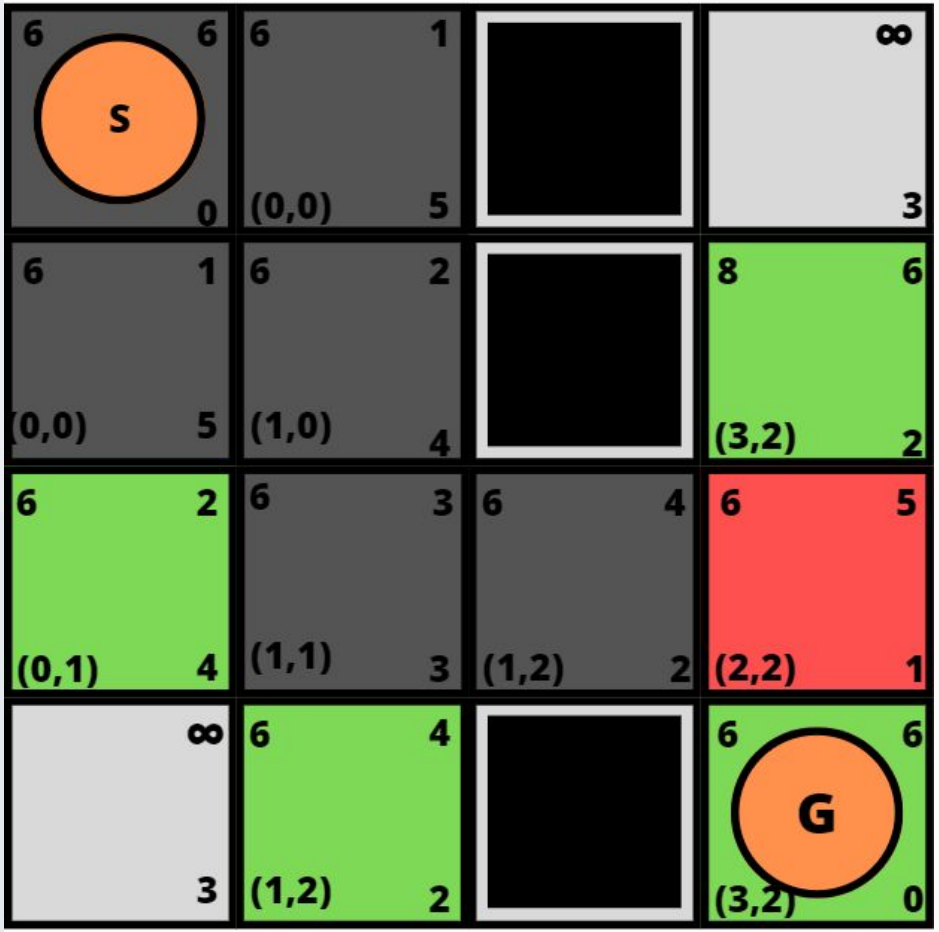
7 **else** Insertar v en $Open$



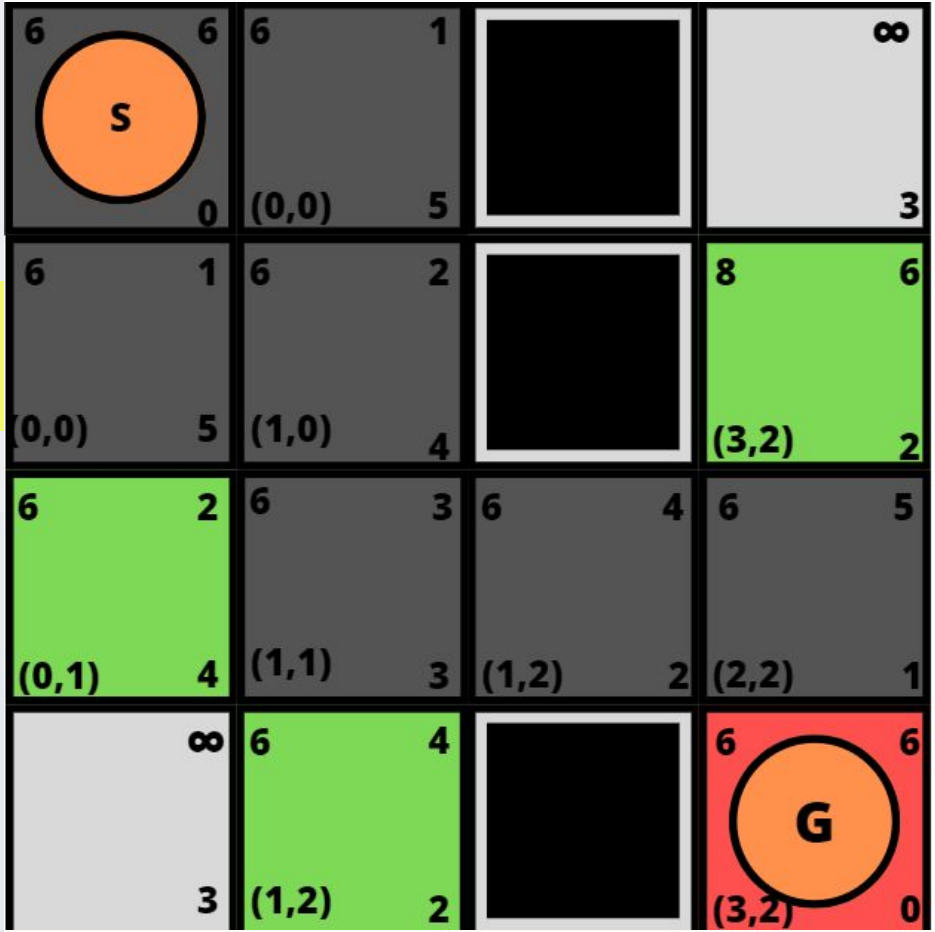
1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$

4 **while** $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor-f

6 **if** u es objetivo **return** u

7 **for each** $v \in Succ(u)$ **do**

1 $cost_v = g(u) + c(u, v)$

2 **if** $cost_v \geq g(v)$ **return**

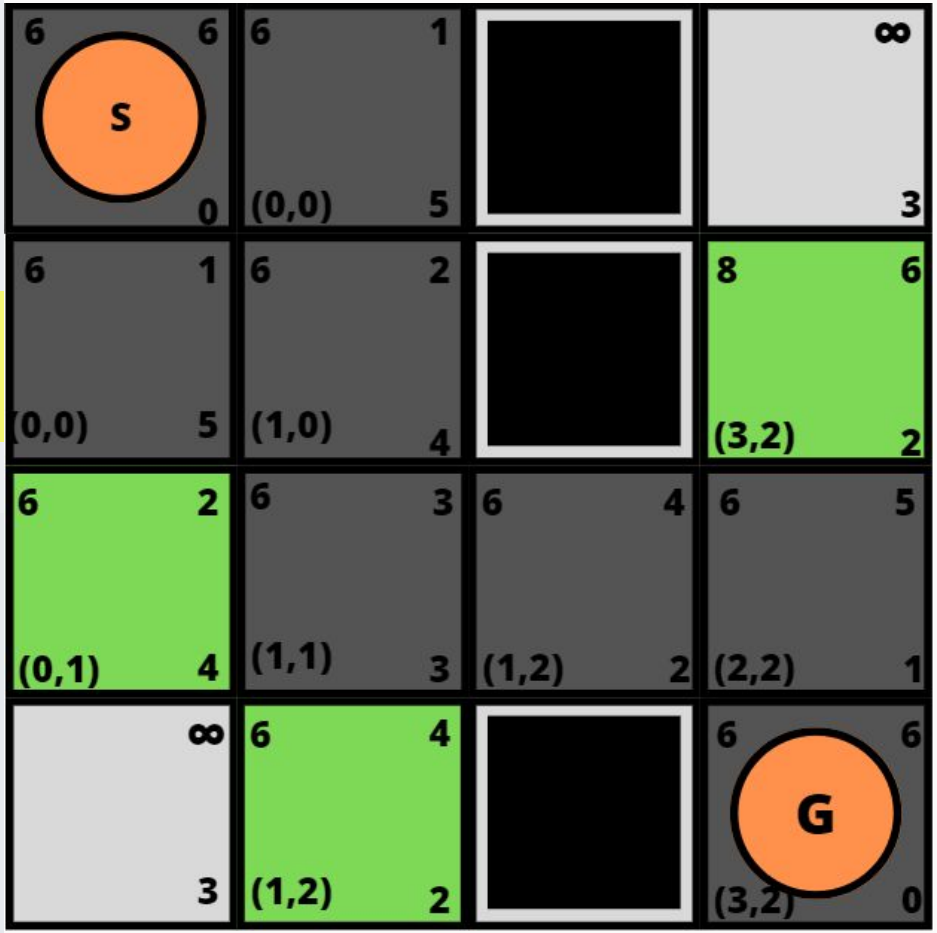
3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

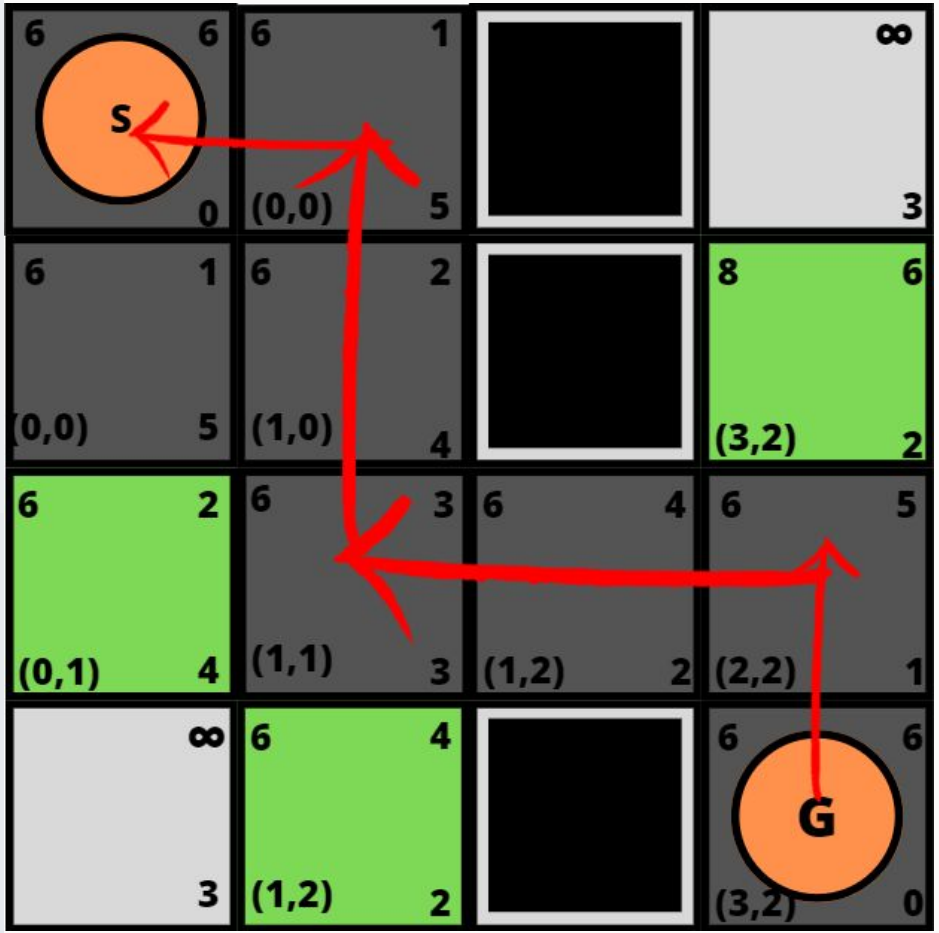
5 $f(v) \leftarrow g(v) + h(v)$

6 **if** $v \in Open$ **then** Reordenar $Open$

7 **else** Insertar v en $Open$



1. **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
2. $Open \leftarrow \{s_0\}$
3. $g(s_0) \leftarrow 0;\ f(s_0) \leftarrow h(s_0)$
4. **while** $Open \neq \emptyset$
5. Extrae un u desde $Open$ con menor valor-f
6. **if** u es objetivo **return** u
7. **for each** $v \in Succ(u)$ **do**
    1. $cost_v = g(u) + c(u, v)$
    2. **if** $cost_v \geq g(v)$ **return**
    3. $parent(v) \leftarrow u$
    4. $g(v) \leftarrow cost_v$
    5. $f(v) \leftarrow g(v) + h(v)$
    6. **if** $v \in Open$ **then** Reordenar $Open$
    7. **else** Insertar v en $Open$



Otro ejemplo de A* - Computerphile - Youtube

A* (A Star) Search Algorithm

¿Por qué A* es óptimo? - Youtube

¿Por qué A* es óptimo? - YouTube