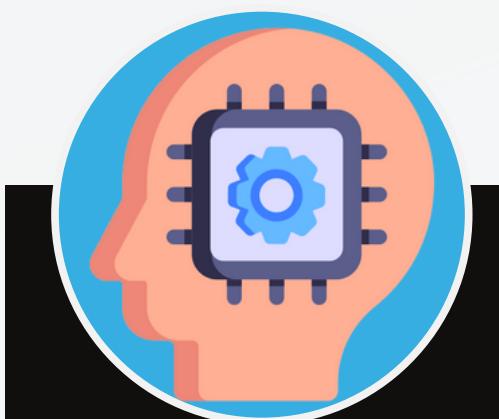


PRIMEIRO TRABALHO

SCC0230 - Inteligência Artificial

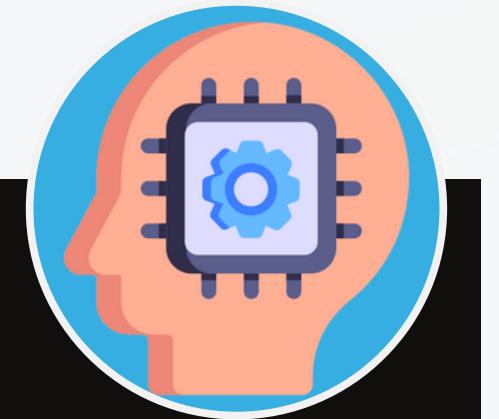
Solange Oliveira Rezende



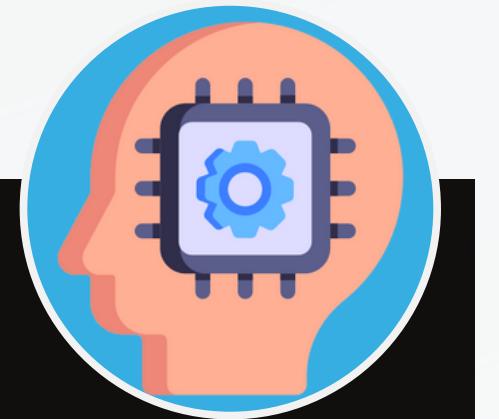
Bernardo
Maia Coelho
Vice-Líder
N^a USP 12542481



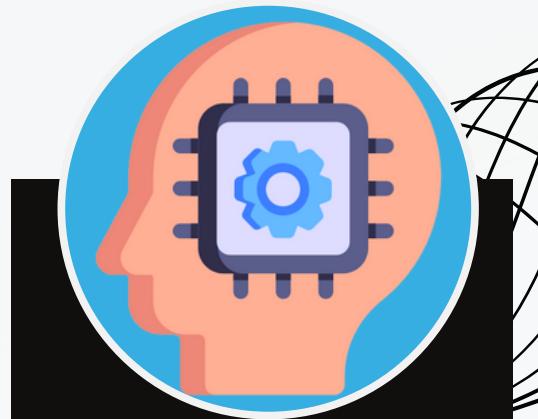
João Gabriel
Sasseron
Membro
N^a USP 12542564



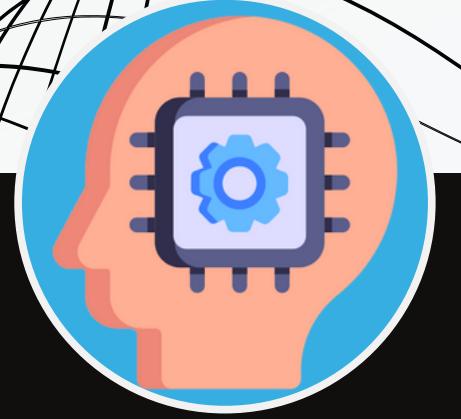
João Pedro
Buzzo Silva
Membro
N^a USP 10425191



Marcos Patrício
Nogueira Filho
Membro
N^a USP 11819063



Pedro Guilherme
dos Reis Teixeira
Membro
N^a USP 12542477



Rogério
Lopes Lübe
Líder
N^a USP 10770113

SUMÁRIO

01

TEMA E JUSTIFICATIVA

02

O QUE É O SOKOBAN?

03

BUSCA NÃO INFORMADA

04

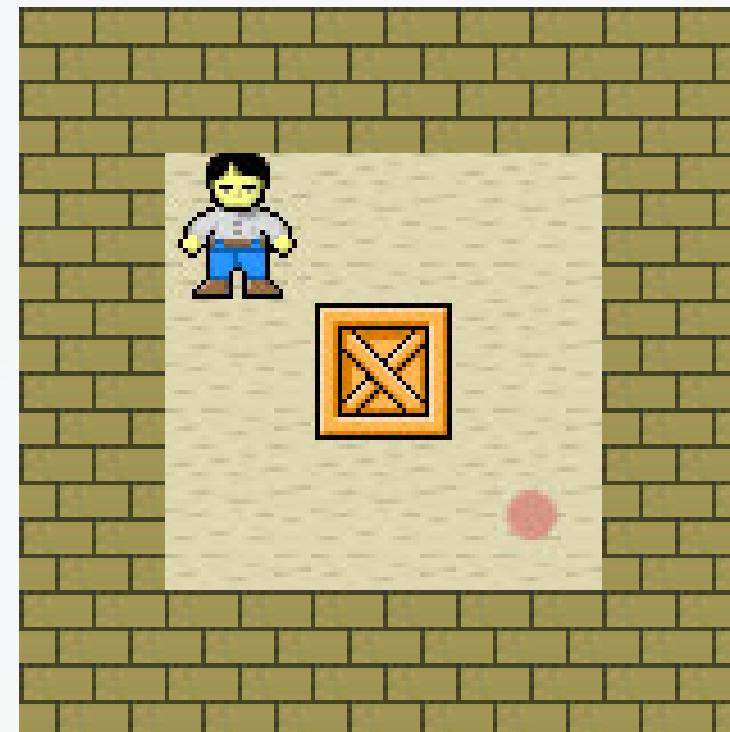
BUSCA INFORMADA



TEMA E JUSTIFICATIVA

TEMA

- O trabalho visa criar algoritmos de **busca**, tanto **informada** quanto **não informada**, para solucionar desafios do jogo **Sokoban**.

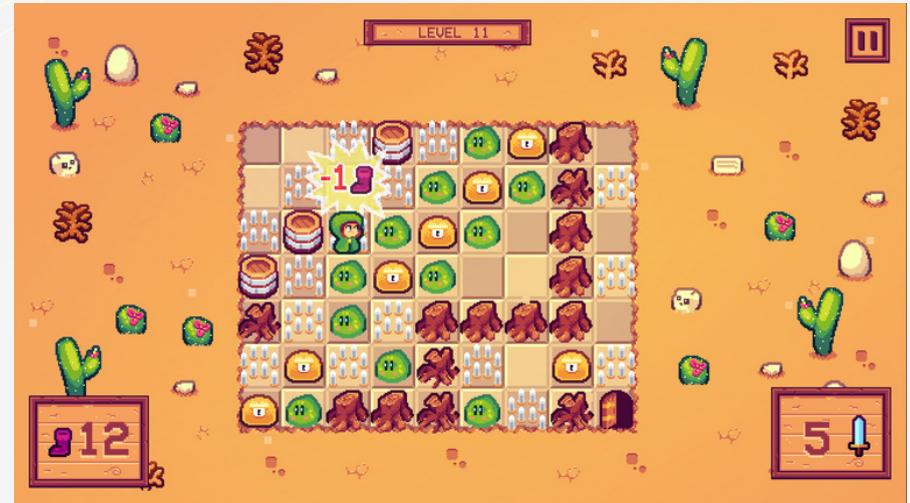


JUSTIFICATIVA

- A solução de um tabuleiro de **Sokoban** é um problema P-SPACE completo.
- É bem mais difícil do que parece!
- Desafios:
 - As possibilidades são muito grandes.
 - O jogador pode precisar passar por células já visitadas.
 - A árvore de possibilidades cresce rapidamente (alto branching factor).
- É um jogo divertido :)

JUSTIFICATIVA

- Vários jogos clássicos e modernos são inspirados em sokoban ou têm partes inspiradas em sokoban.



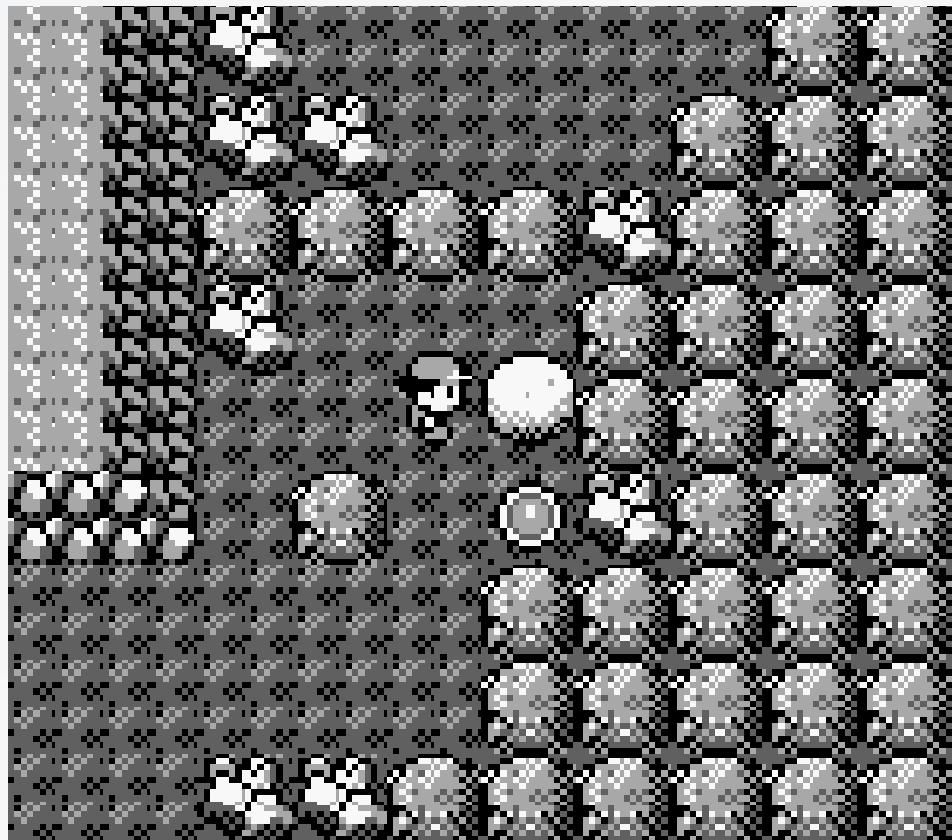
Slime Warrior Sokoban (2023)



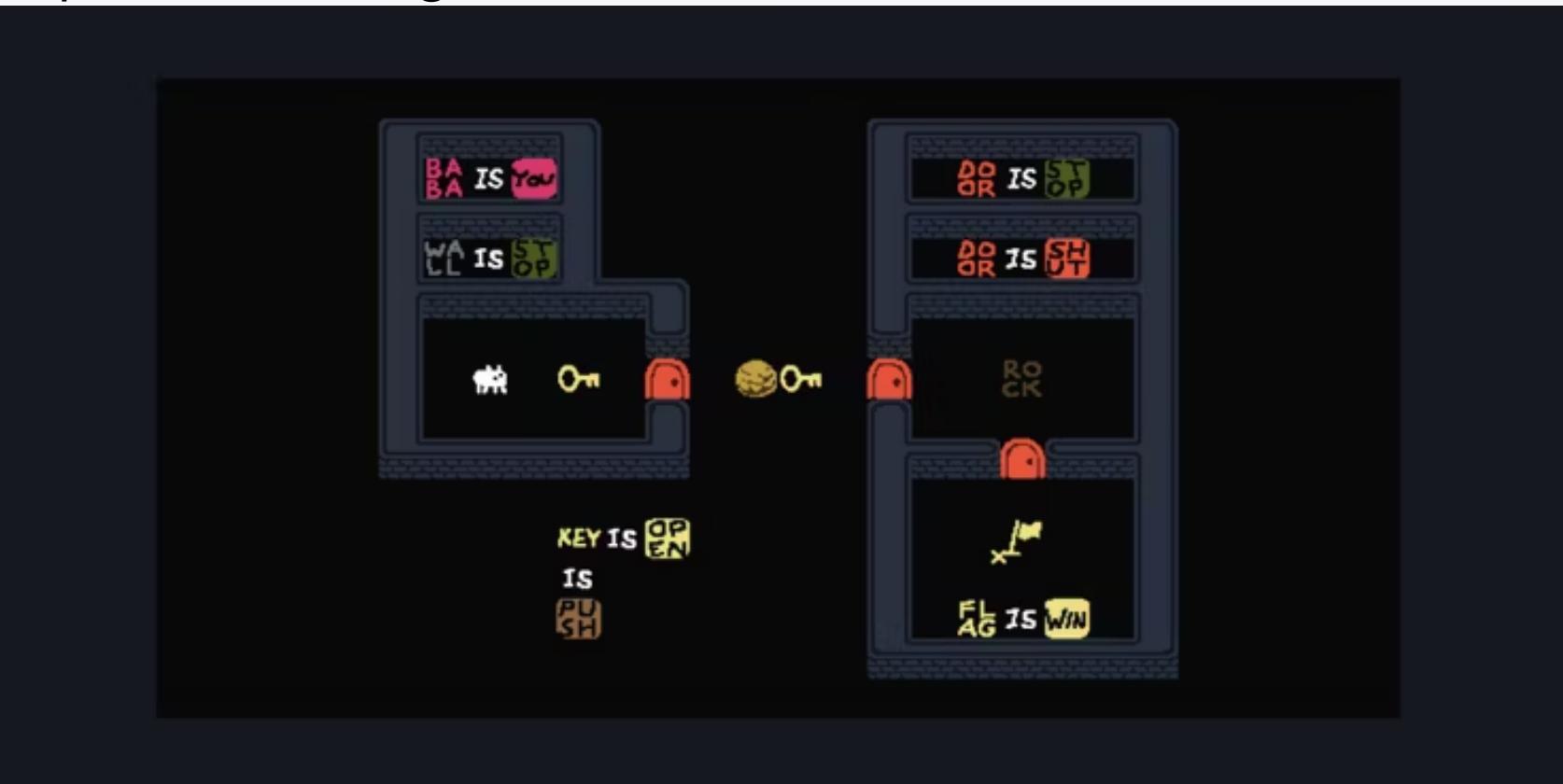
Stephen's Sausage Roll (2016)



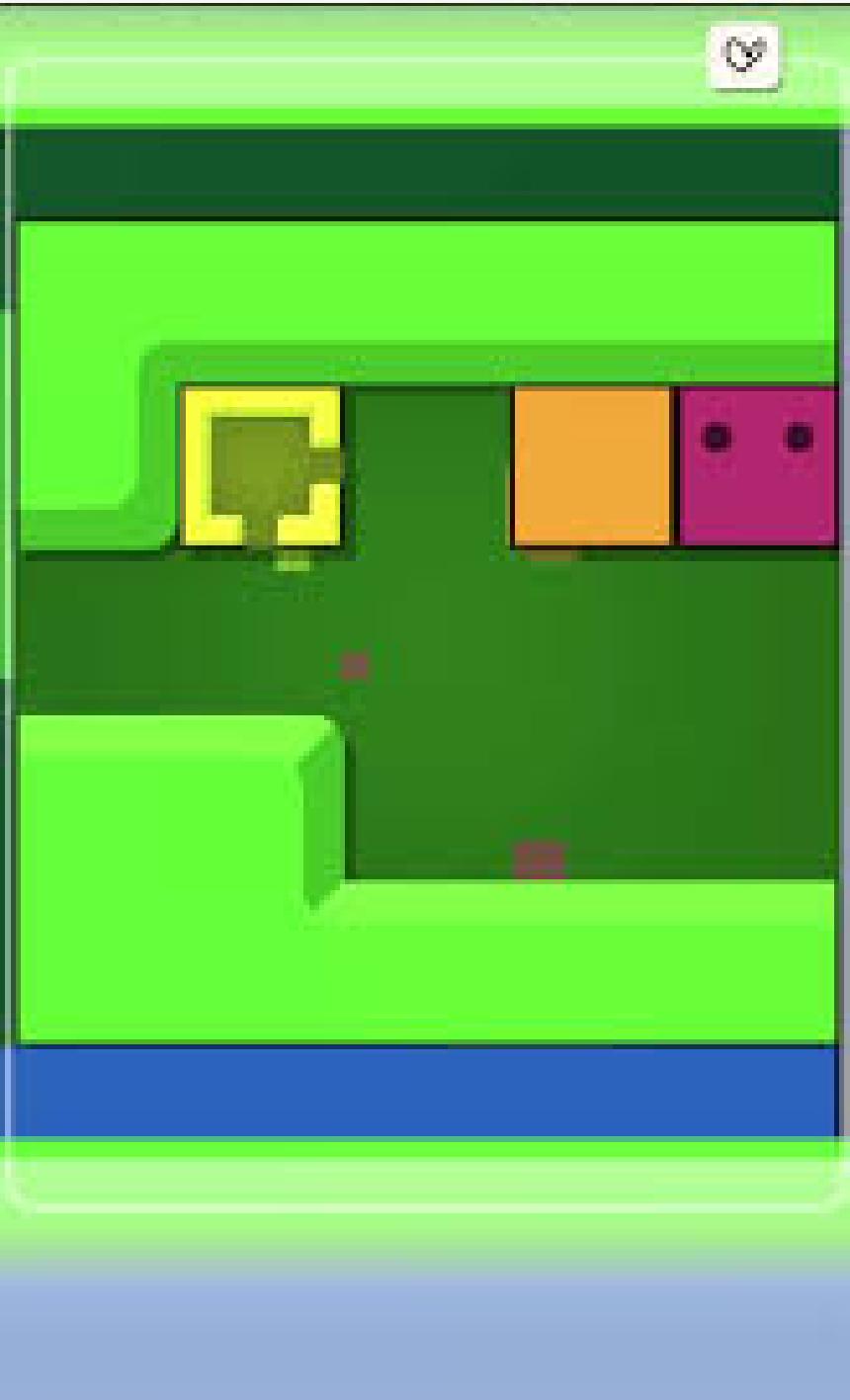
Pokmon Platinum (2008)



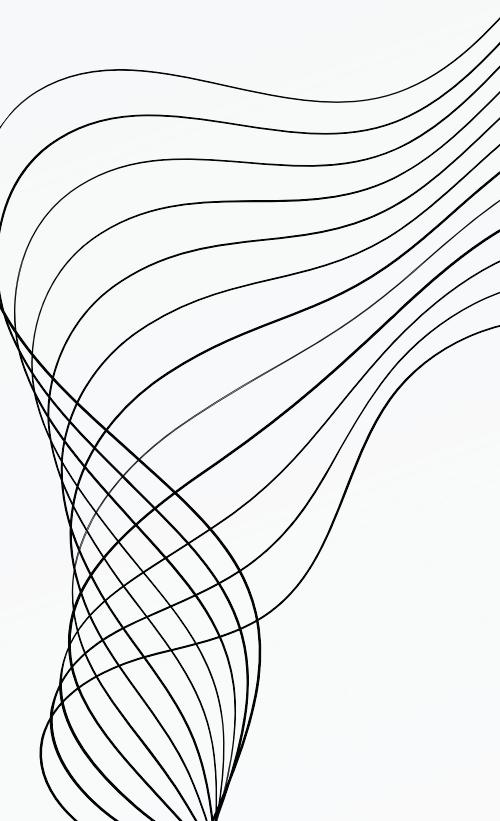
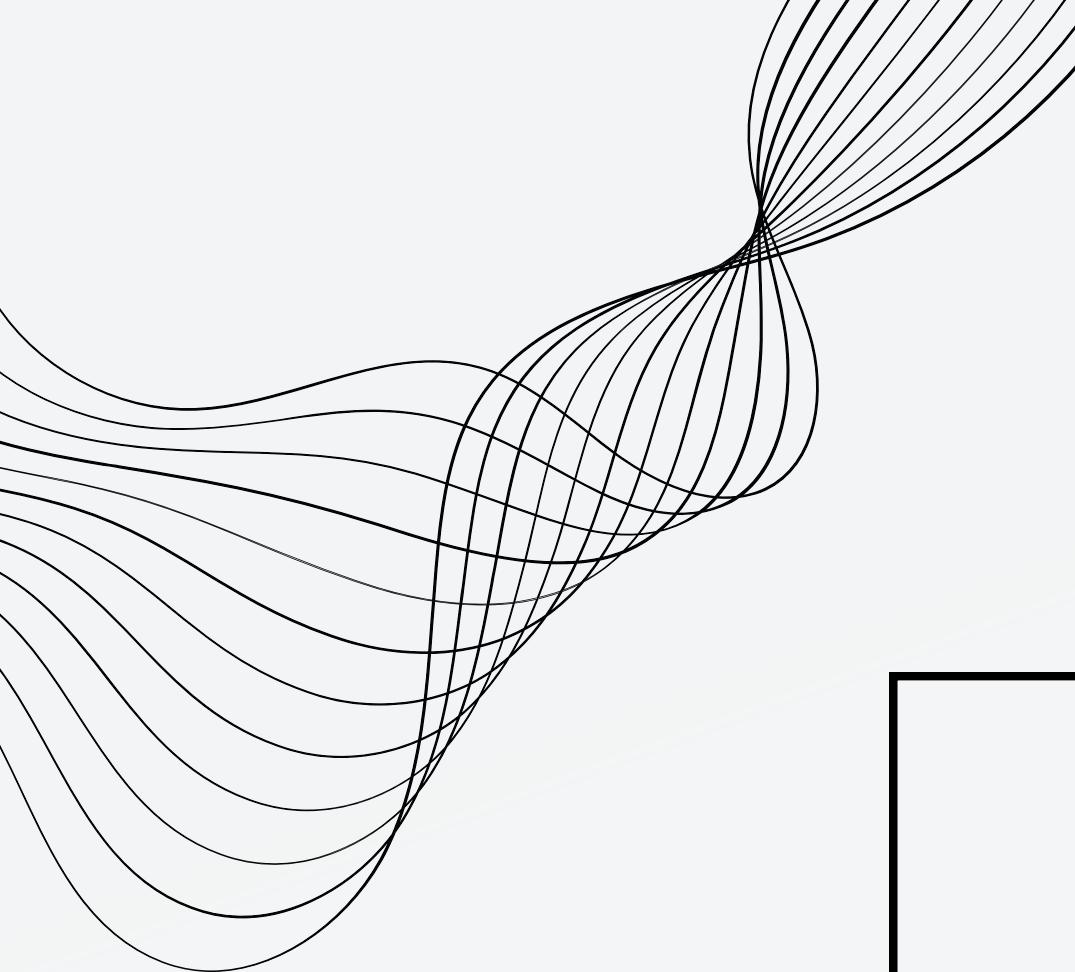
Pokmon Blue (1996)



Baba is You (2019)

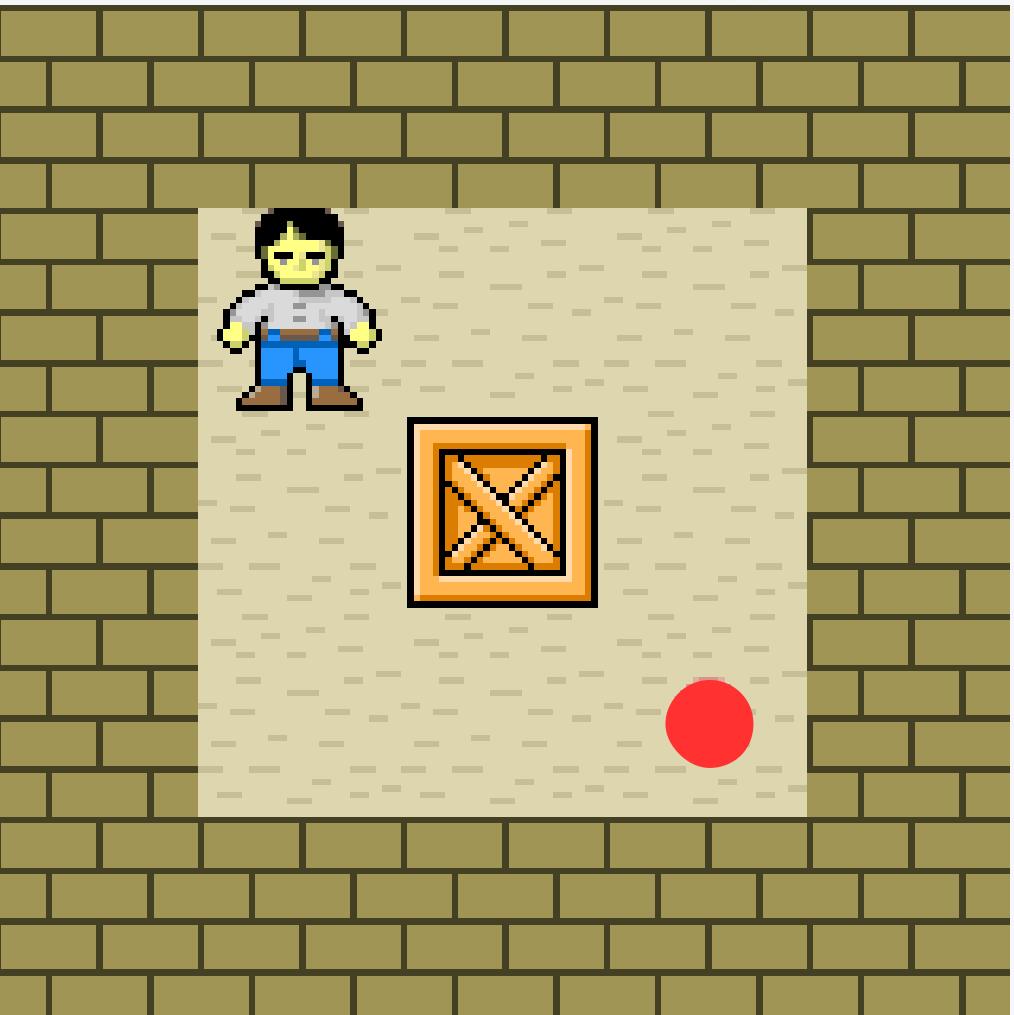


Patrick's Parabox (2022)

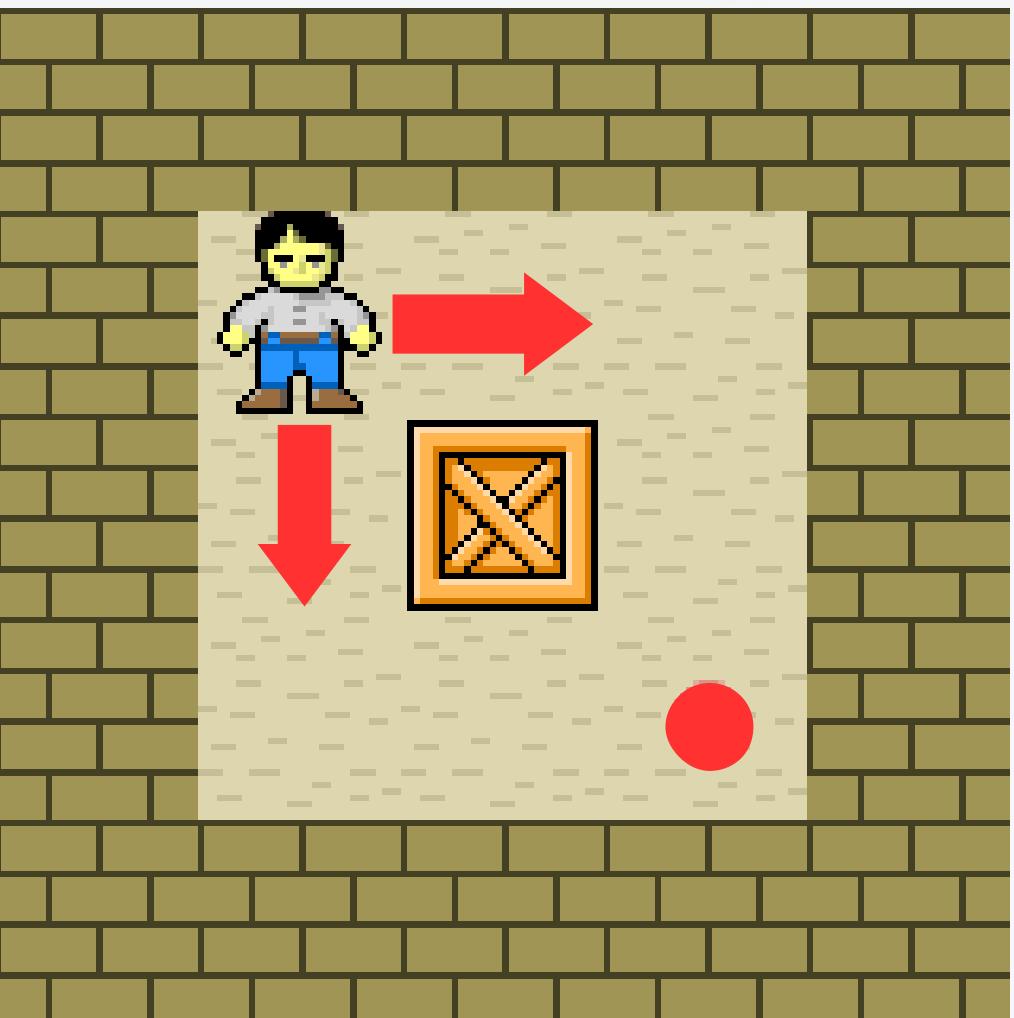


**O QUE É O
SOKOBAN?**

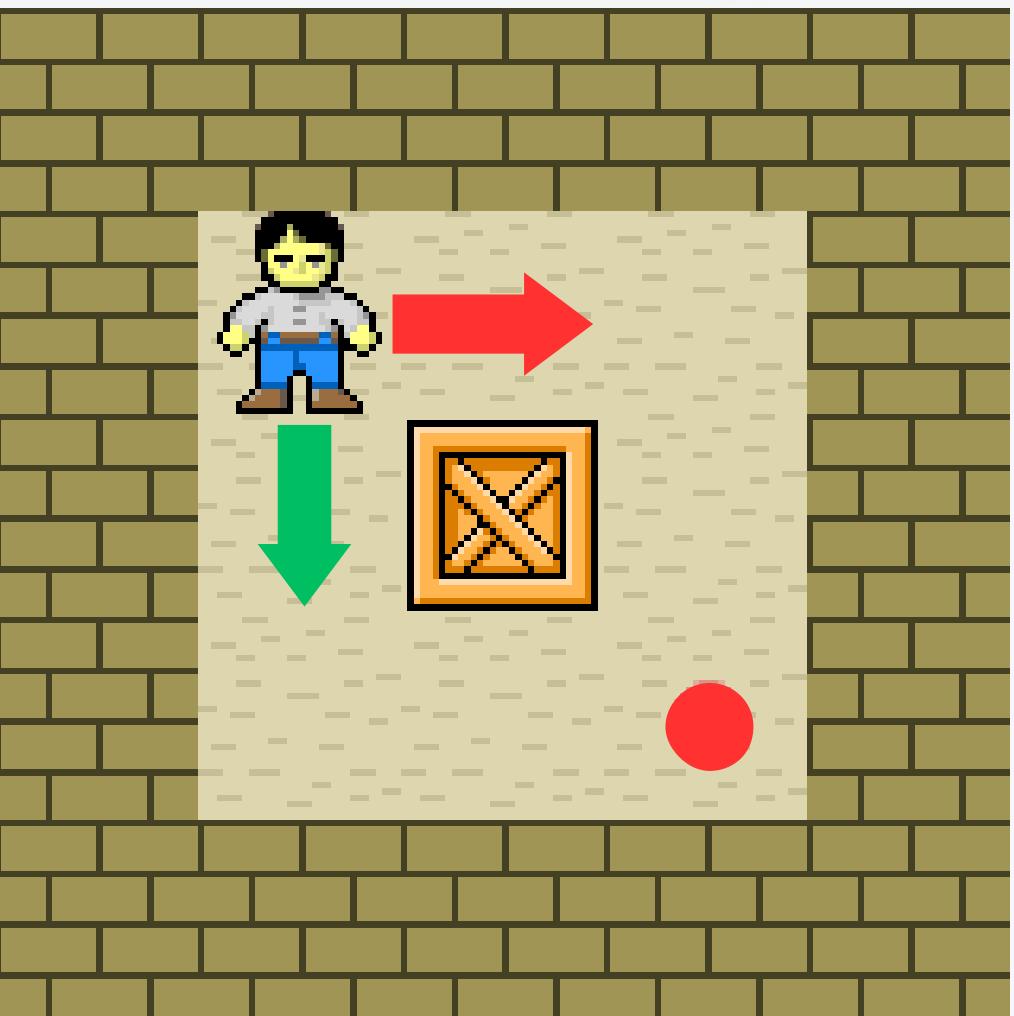
SOKOBAN



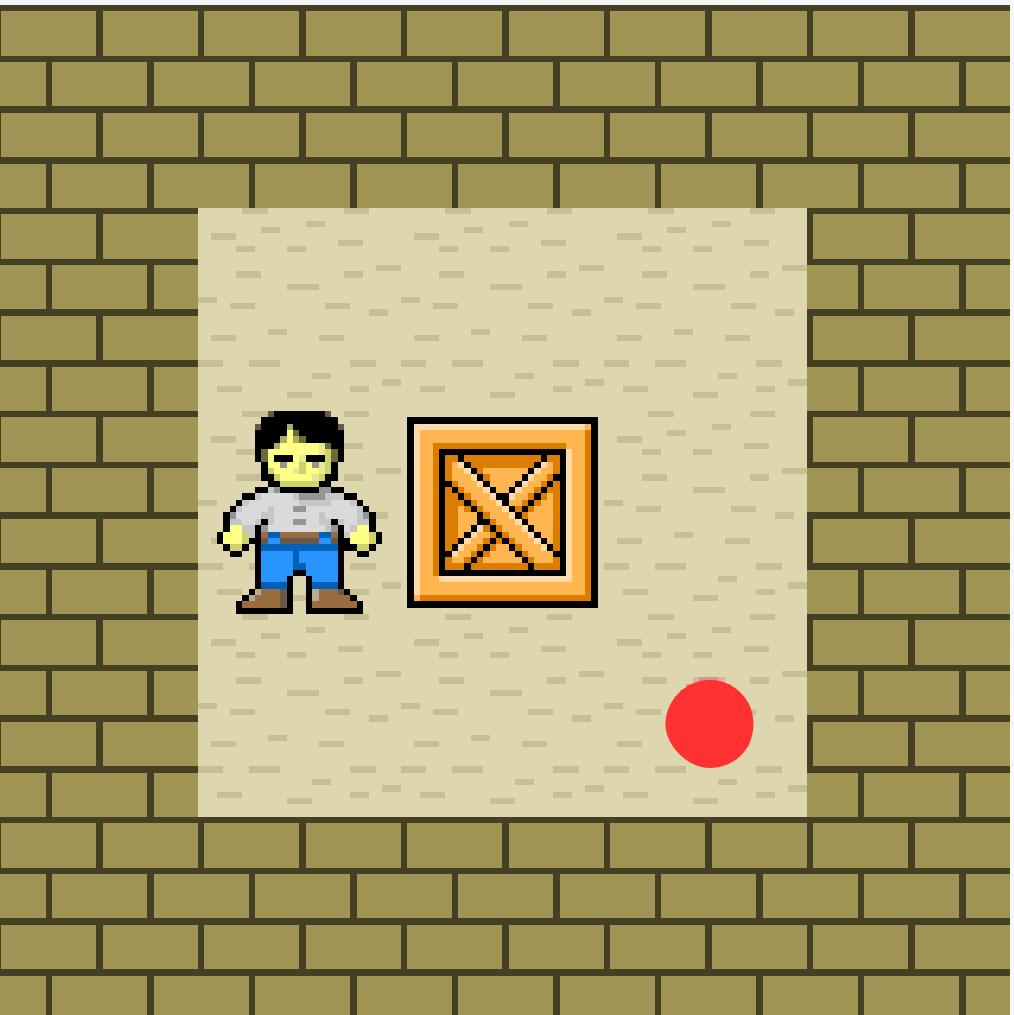
SOKOBAN



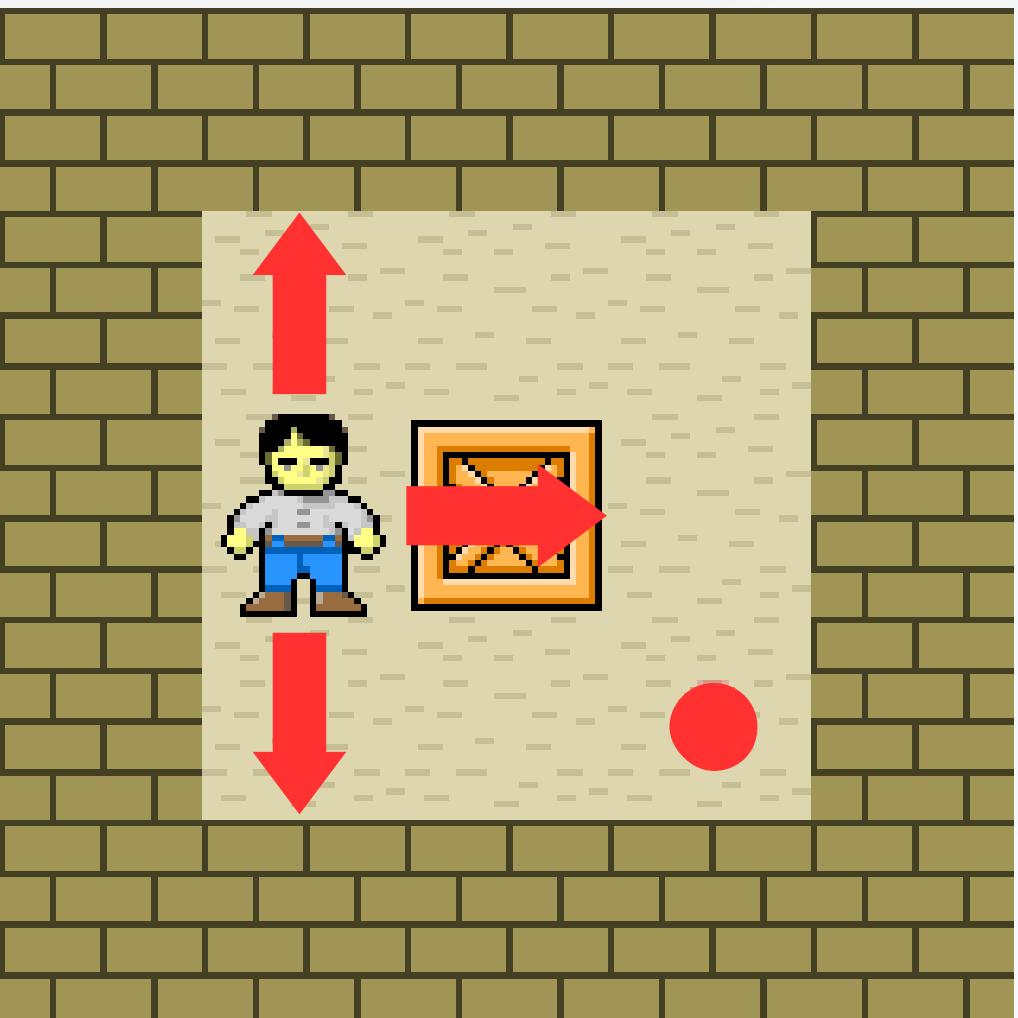
SOKOBAN



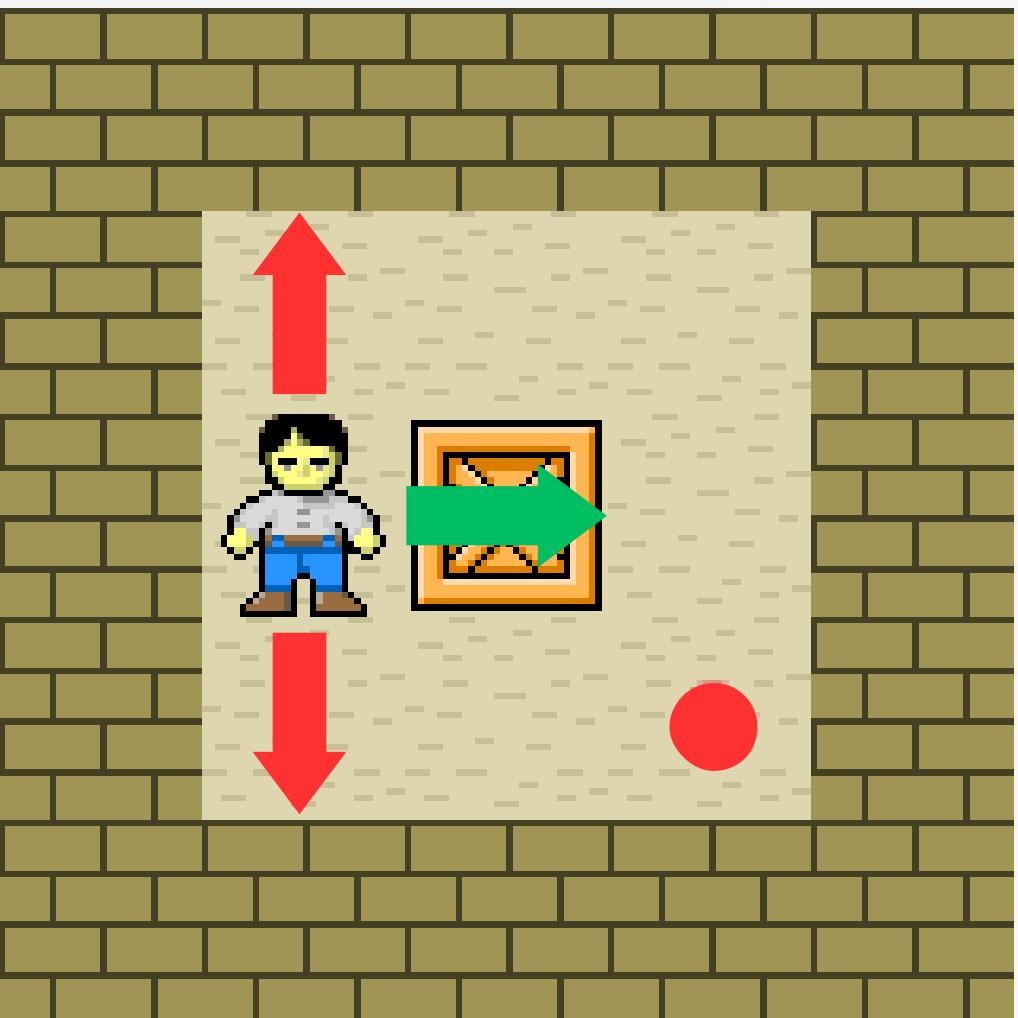
SOKOBAN



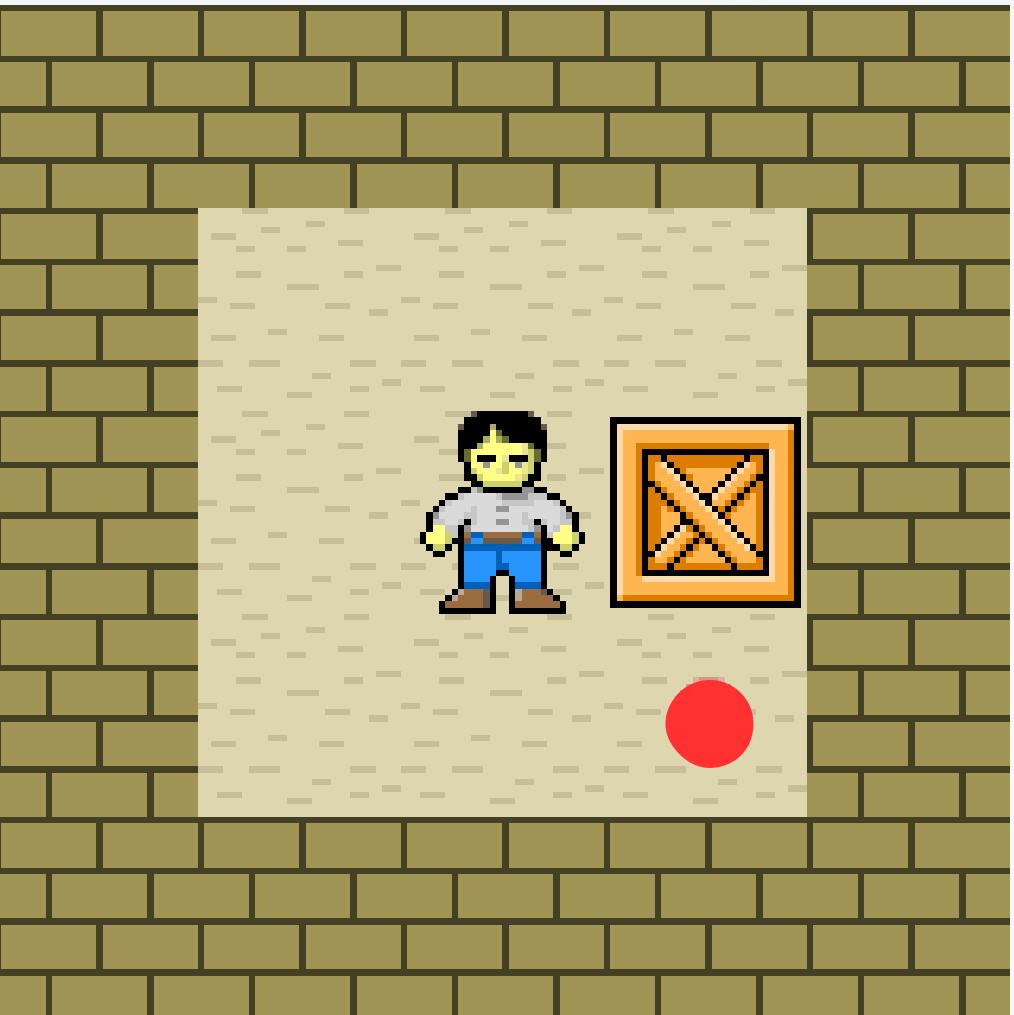
SOKOBAN



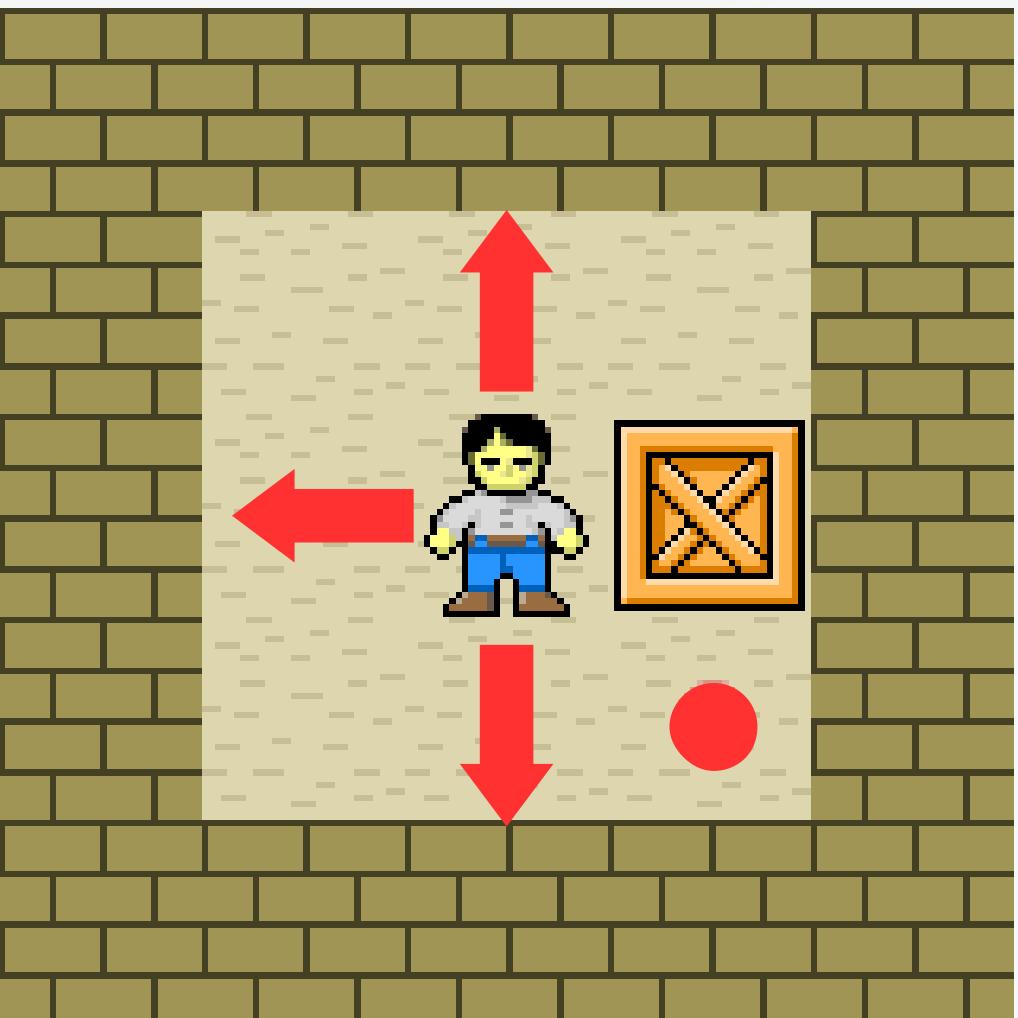
SOKOBAN



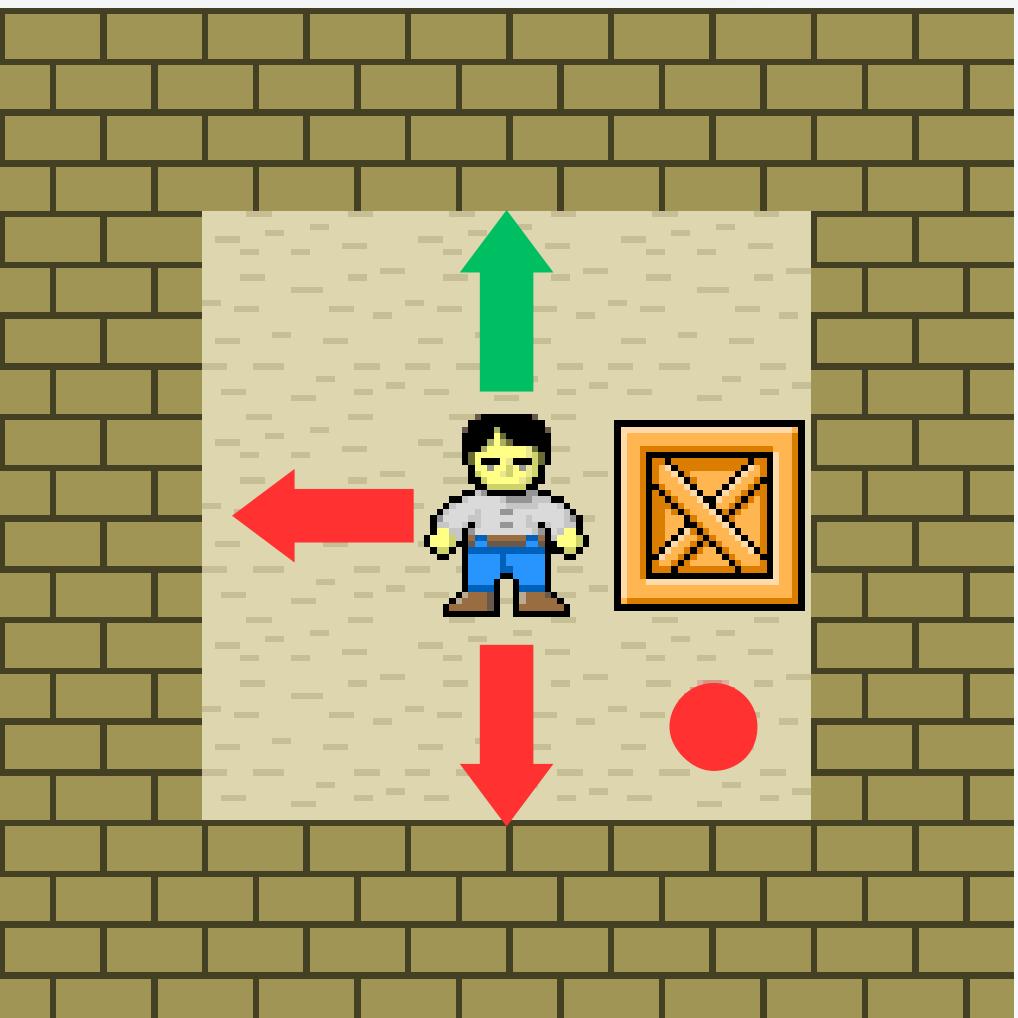
SOKOBAN



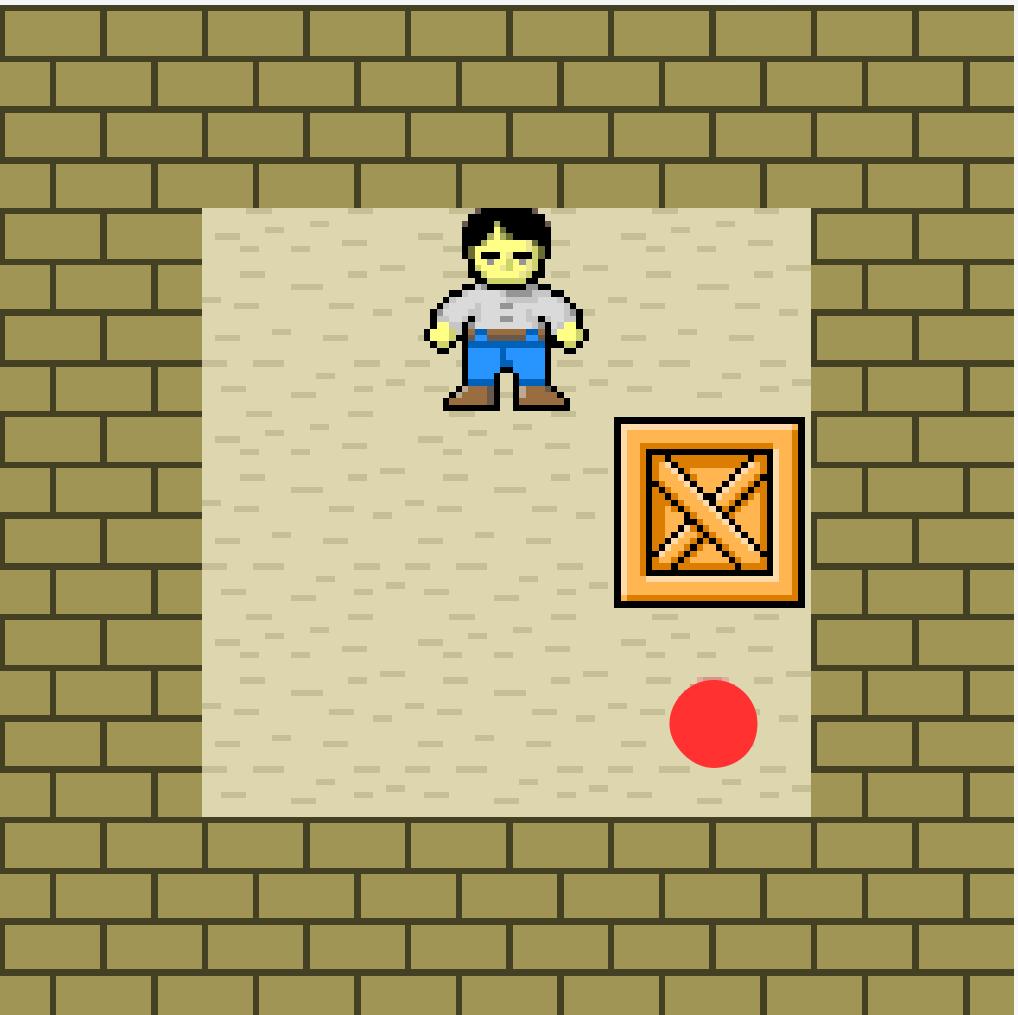
SOKOBAN



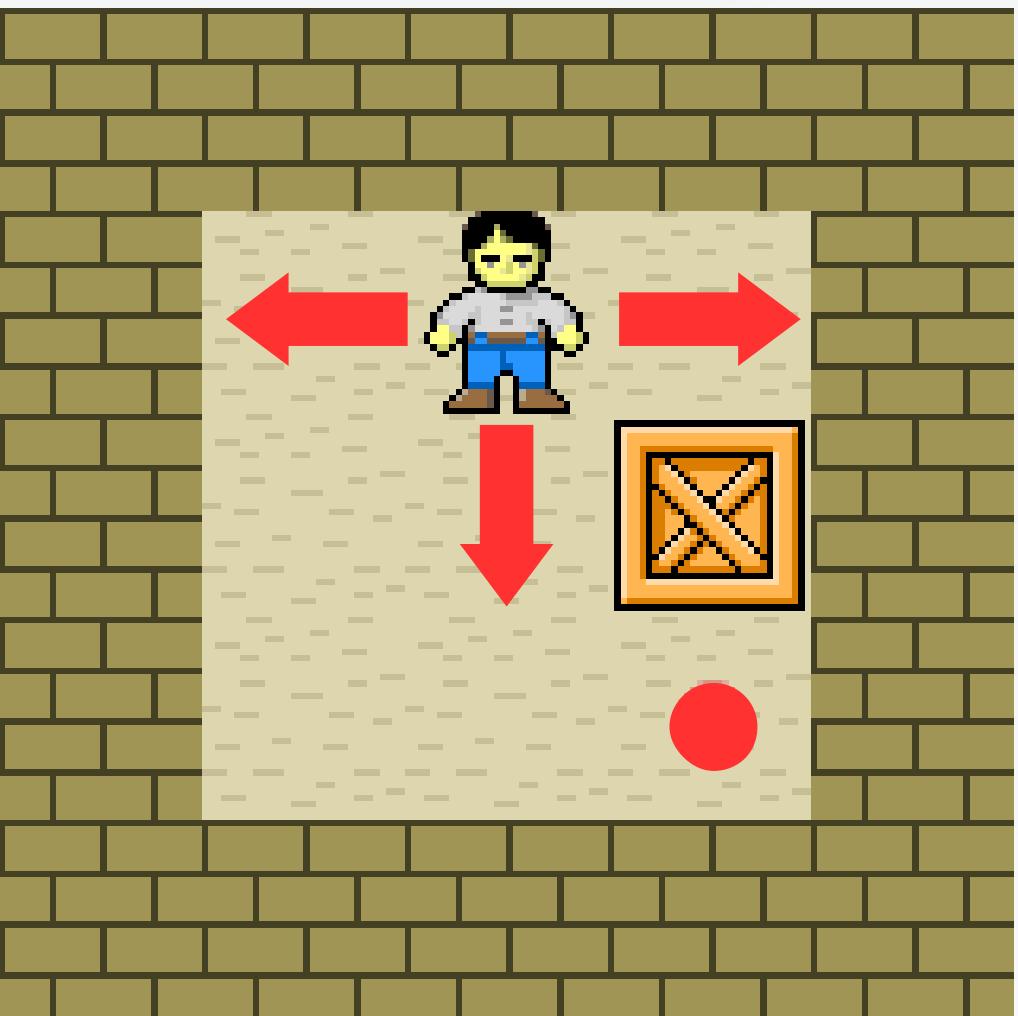
SOKOBAN



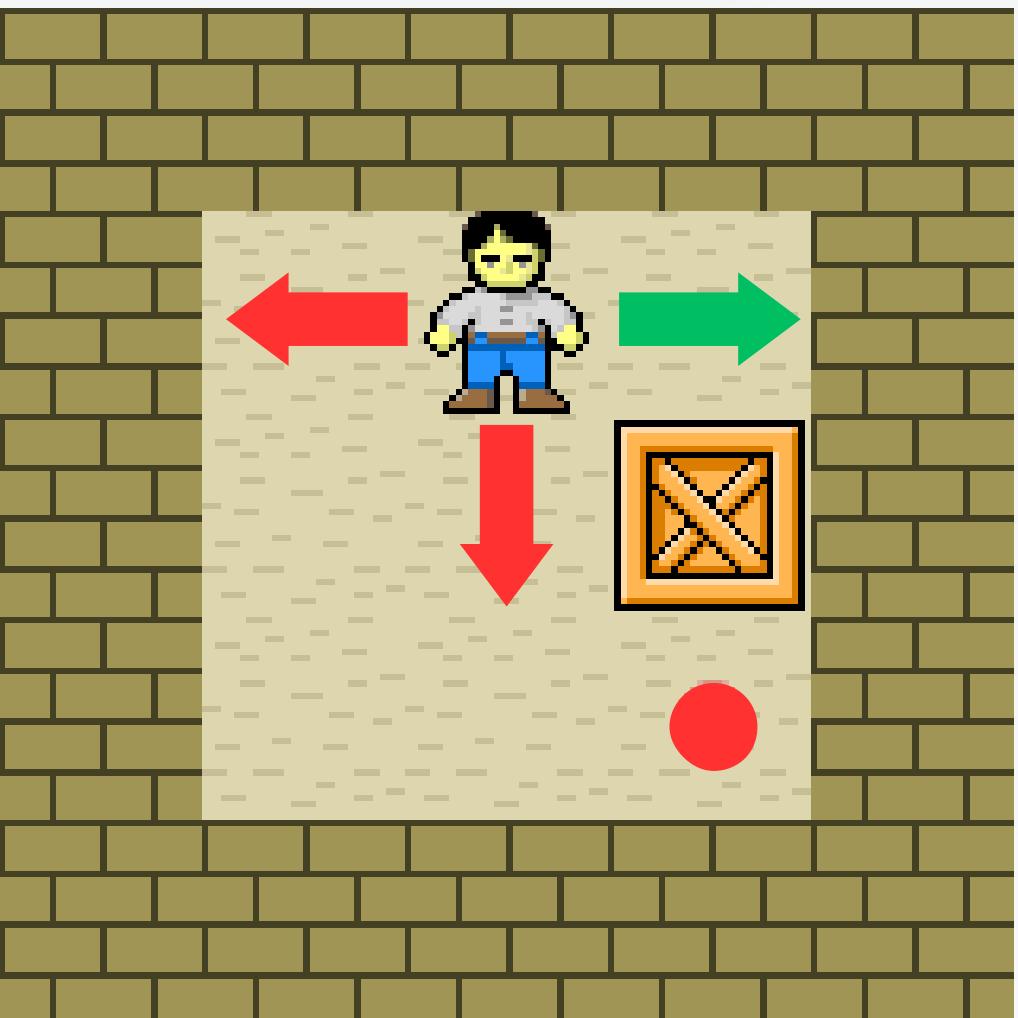
SOKOBAN



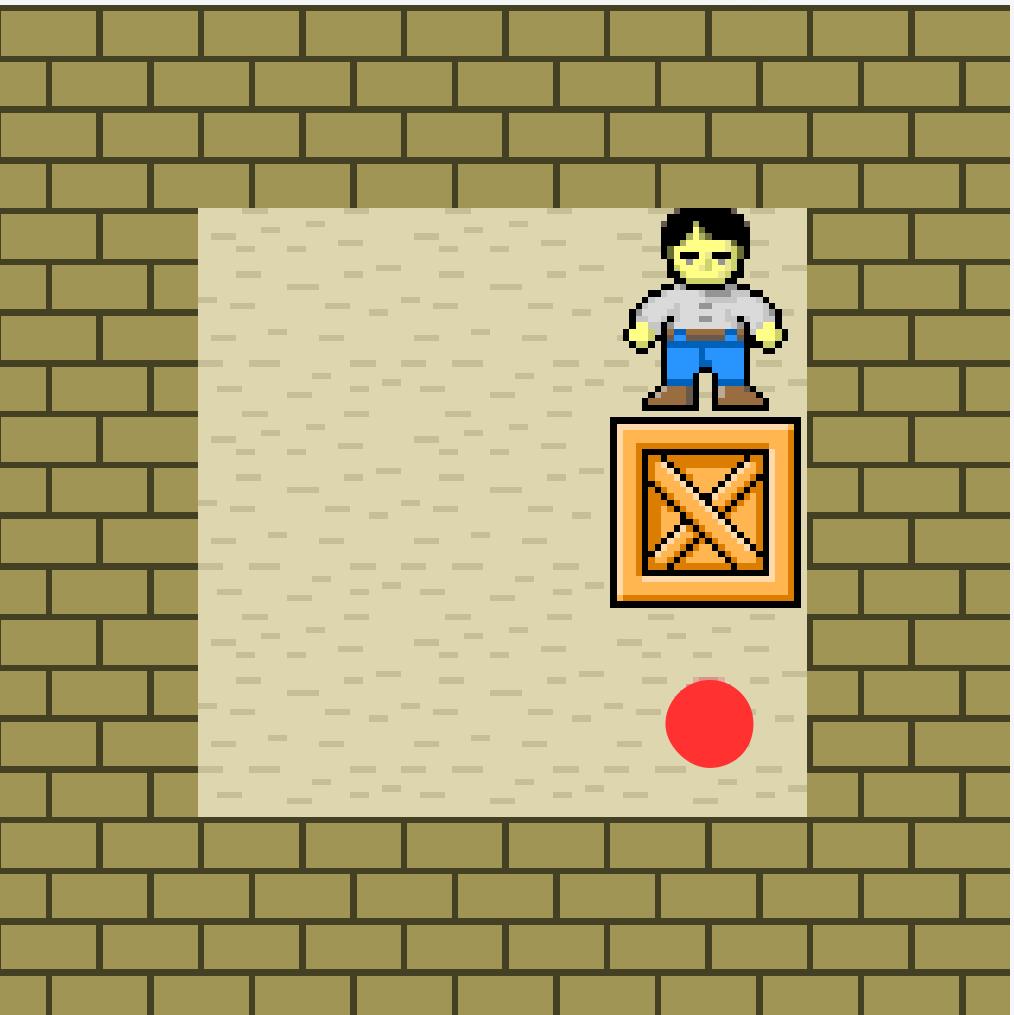
SOKOBAN



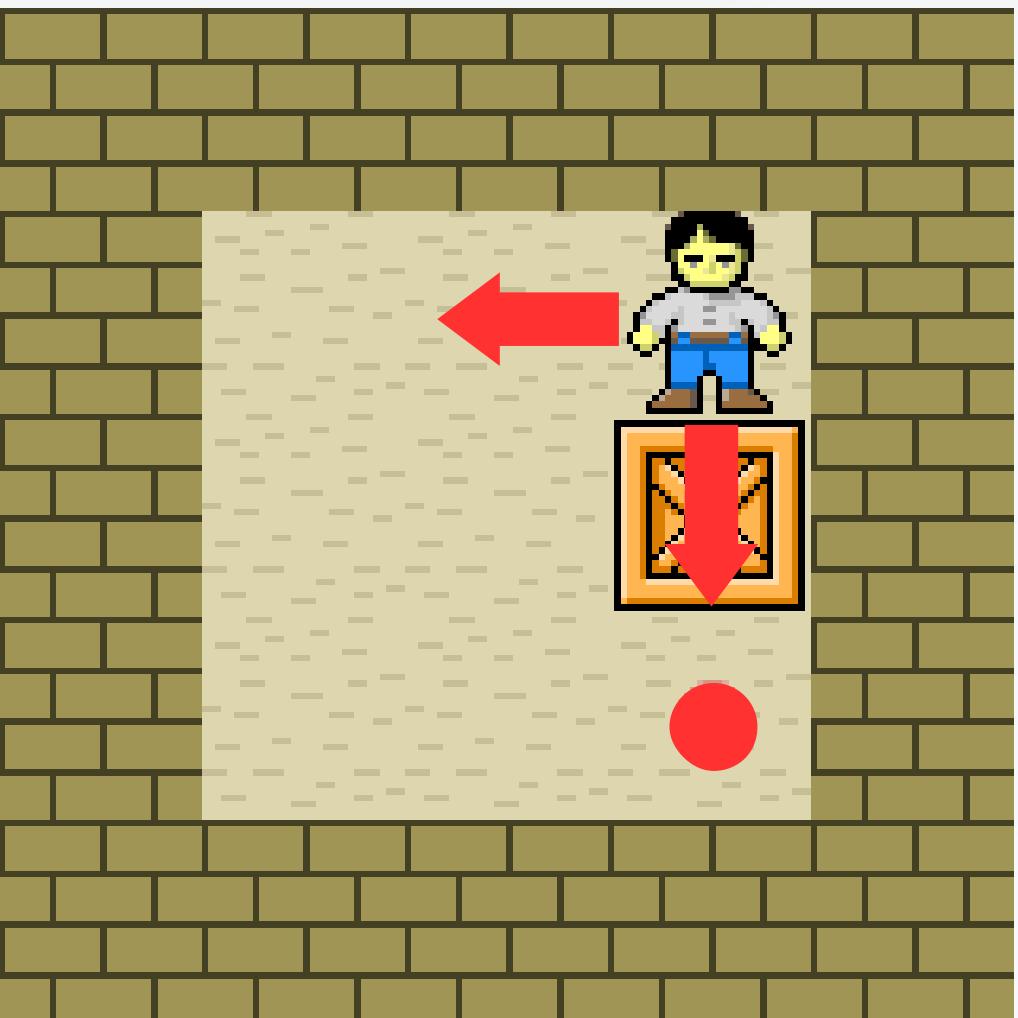
SOKOBAN



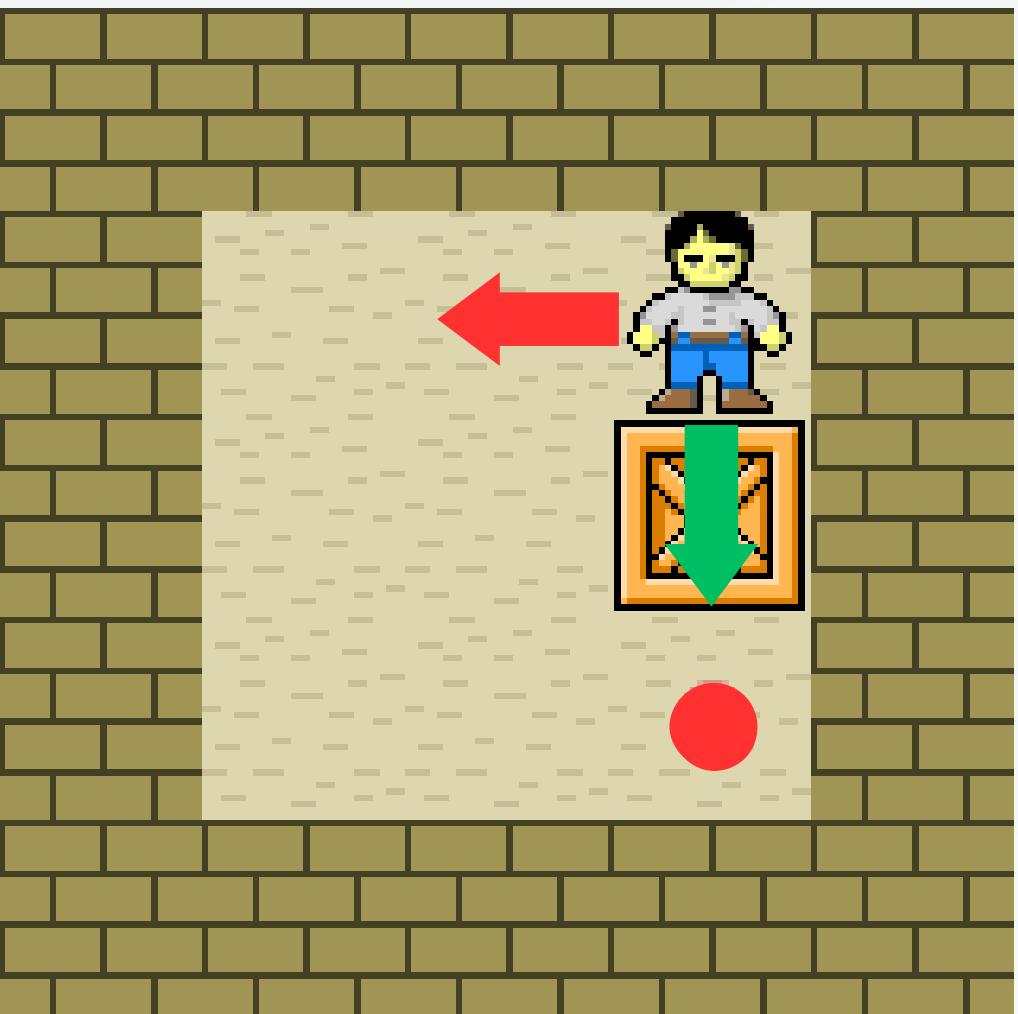
SOKOBAN



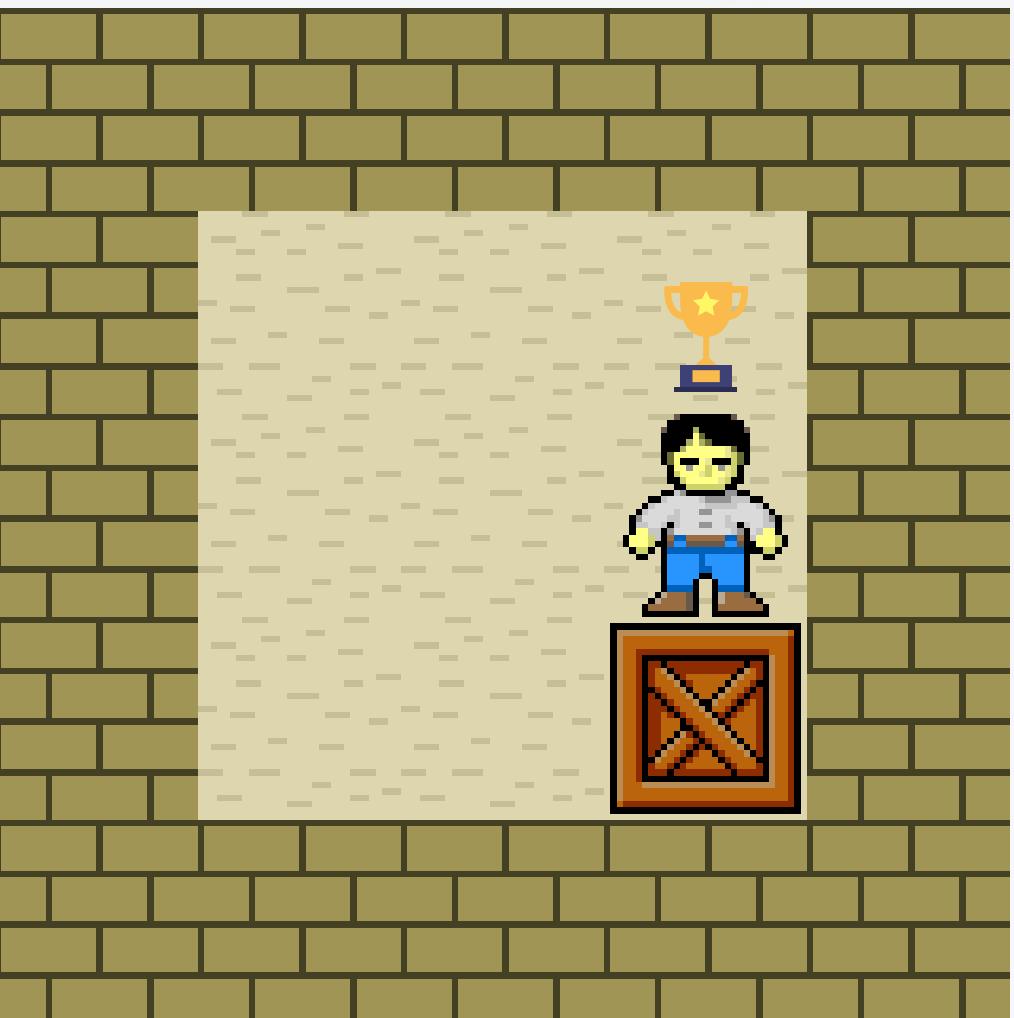
SOKOBAN



SOKOBAN



SOKOBAN

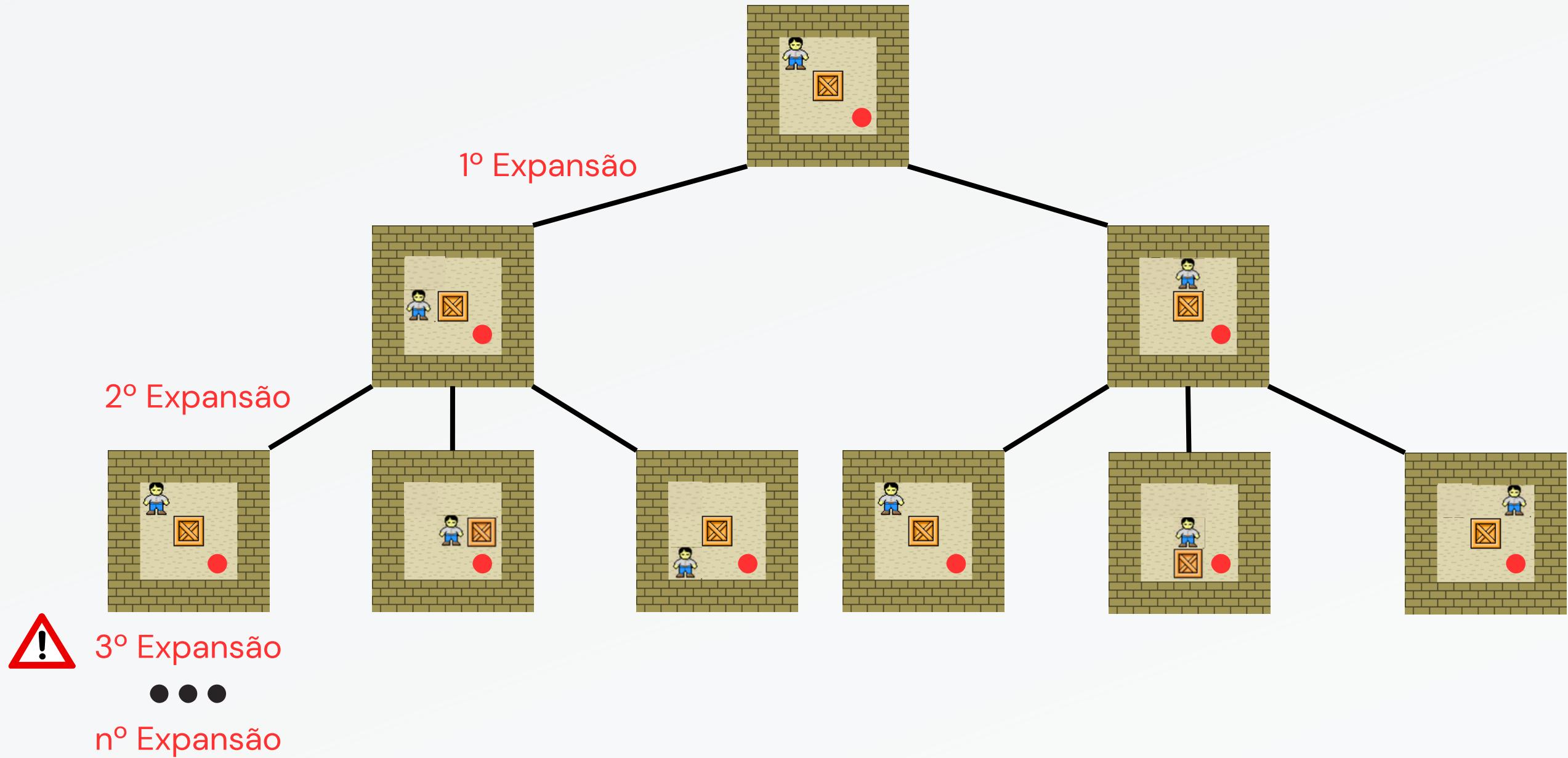




**BUSCA NÃO
INFORMADA**

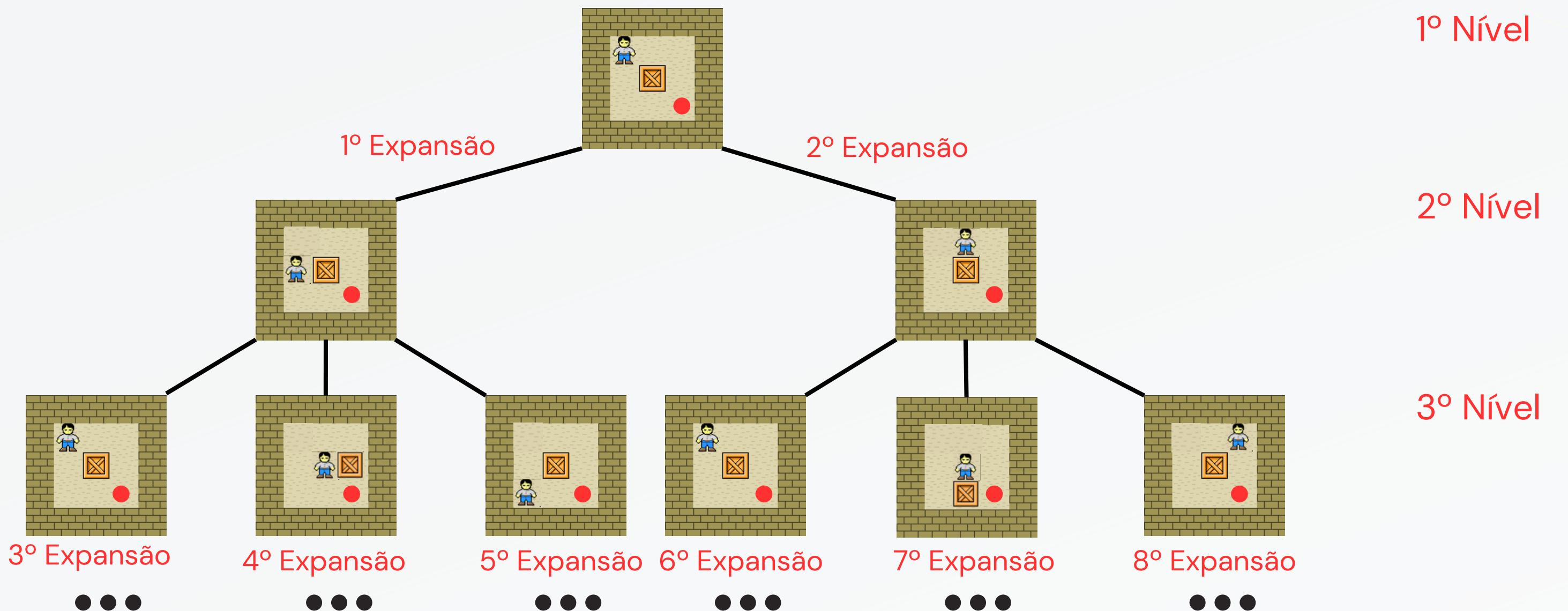
BUSCA EM PROFUNDIDADE

- A busca em profundidade explora tão profundamente quanto possível antes de retroceder.



BUSCA EM LARGURA

- A busca em largura examina minuciosamente todos os nós no mesmo nível antes de prosseguir para o próximo.

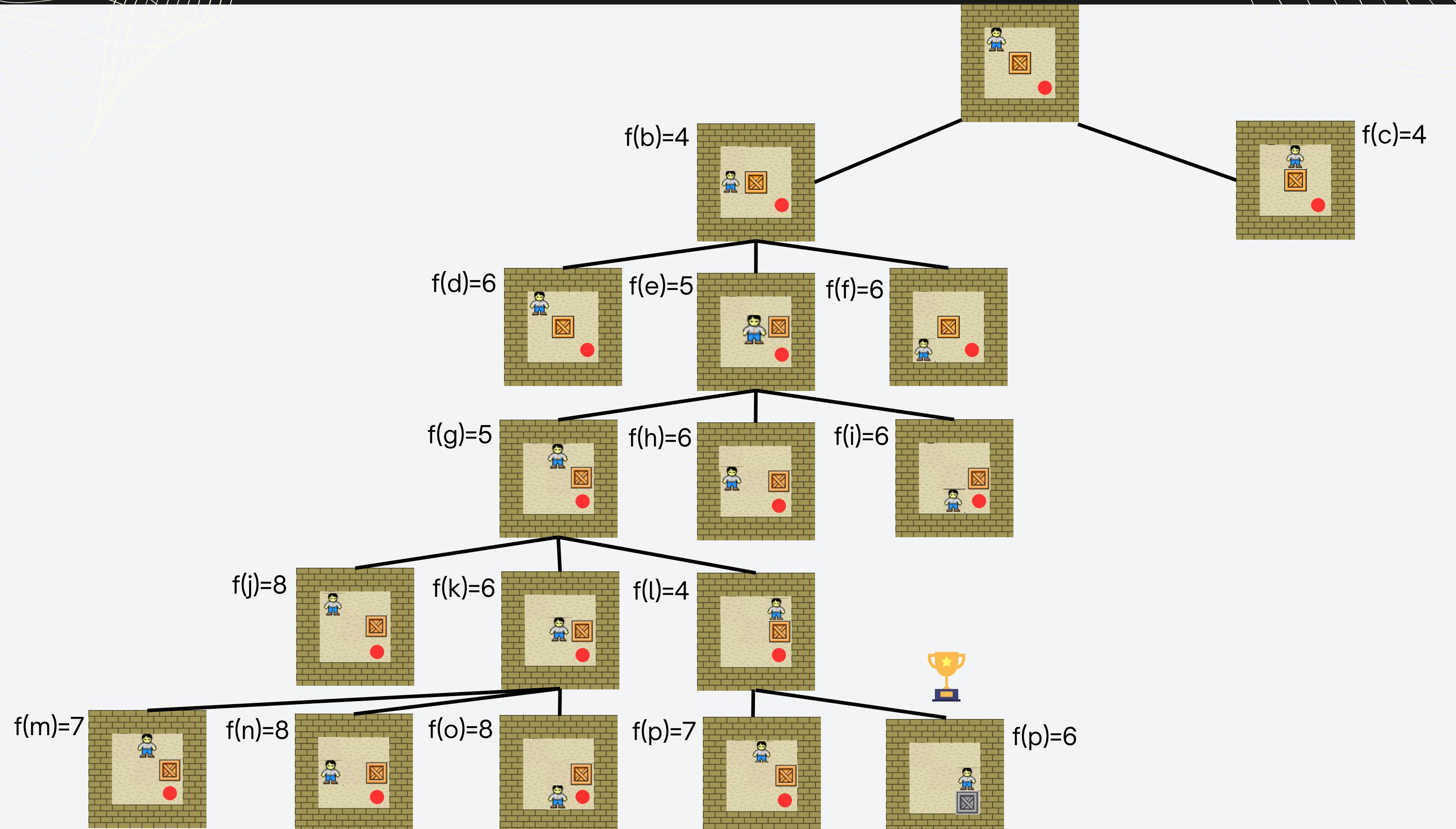


BUSCA INFORMADA

A*

- **Função Heurística** $f(n) = h_1(n) + h_2(n) + h_3(n) + g(n)$:
 - $h_1(n)$ é a FA da distância do **botão até a caixa**;
 - $h_2(n)$ é a FA da distância do **jogador até a caixa mais próxima**.
 - $h_3(n)$ é uma função que concede um bônus para casos em que o jogador, a caixa e o botão estão alinhados horizontal ou verticalmente ou casos adjacentes a esses.
 - $h_1(n) + h_2(n) + h_3(n)$ é equivalente a **função de avaliação**.
 - $g(n)$ é a FC com 1 de **custo para cada movimento**.
- Além disso, no cálculo das distâncias, utilizamos como base a métrica de **distância de Manhattan**.
- Em caso de empate, o caminho escolhido é referente a ordem alfabética.

A *



A *

- Execução do código em python para solução de um tabuleiro.

Starting configuration:
Is solved: False

Type anything to continue

=====

<Current: is_solved(False) d(0) score(4) />

<Trying: is_solved(False) d(1) score(3) />

<Trying: is_solved(False) d(1) score(3) />

Type anything to continue

=====

<Current: is_solved(False) d(1) score(3) />

<Trying: is_solved(False) d(2) score(4) />

<Trying: is_solved(False) d(2) score(1) />


```
<Trying: is_solved(False) d(5) score(2) />

<Trying: is_solved(False) d(5) score(1) />

Type anything to continue
=====
<Current: is_solved(False) d(4) score(1) />

<Trying: is_solved(True) d(5) score(0) />

<Trying: is_solved(False) d(5) score(1) />

Type anything to continue
=====
<Current: is_solved(True) d(5) score(0) />

```

A*

- Extra: com paredes

```
<Trying: is_solved(False) d(4) score(18) />

Type anything to continue
=====
<Current: is_solved(False) d(4) score(18) />

<Trying: is_solved(False) d(5) score(19) />

<Trying: is_solved(False) d(5) score(17) />

Type anything to continue
```

```
Type anything to continue
=====
<Current: is_solved(False) d(12) score(10) />

<Trying: is_solved(False) d(13) score(11) />

<Trying: is_solved(False) d(13) score(11) />

<Trying: is_solved(False) d(13) score(11) />

Type anything to continue
```

```
Type anything to continue
=====
<Current: is_solved(False) d(14) score(8) />

<Trying: is_solved(False) d(15) score(7) />

<Trying: is_solved(False) d(15) score(7) />

<Trying: is_solved(False) d(15) score(7) />

Type anything to continue
```

OBRIGADO!