Trabalho 1
Agente Bombeiro

Inteligência Artificial - Prof. Murilo Naldi

Alcides Mignoso e Silva
Matheus Victorello

Visão geral

- Busca em largura
  - Caminho mínimo
- Modelagem
- Duas regras auxiliares
- Cinco regras de mudança de estado
- Representação gráfica

Modelagem

(I, J, Fogos, CargasExtintor, Extintores)

- *I* é a linha atual
- *J* é a coluna atual
- *Fogos* é a lista de fogos não apagados
- *CargasExtintor* é o número de cargas
- *Extintores* é a lista de extintores que não foram pegos

Modelagem

- Elementos estáticos: escadas, bloqueios(parede), entulho(pedra):
  - escada(0, 1).
  - bloqueio(1, 2).
  - pedra(3, 4).
- Limites:
  - limiteI(K) :- K < 5, K >= 0.
  - limiteJ(K) :- K < 10, K, >= 0.
- Meta:
  - *meta((_, _, [], _, _)).*

Auxiliares

- Determina se uma posição está livre:

```
livre(I, J, Fogos) :-
        not(pedra(I, J)),
        not(escada(I - 1, J)),
        not(escada(I, J)),
        not(bloqueio(I, J)),
        not(pertence((I, J), Fogos)).
```

Auxiliares

- Determina se é possível andar com relação às pedras:

```
andarFogo(I, J, Fogos, CargasExtintor) :-
    (
        not(pertence((I, J), Fogos))
        ;
        (
            pertence((I, J), Fogos),
            CargasExtintor > 0
        )
    ).
```

Mudança de estado

- Pega extintor:

```
s(
        (I, J, Fogos, CargasExtintor, Extintores),
        (I, J, Fogos, NovoCargasExtintor, NovoExtintores)
) :-
        pertence((I, J), Extintores),
        NovoCargasExtintor is 2,
        CargasExtintor == 0,
        remove_elem((I, J), Extintores, NovoExtintores).
```

Mudança de estado

- Anda para direita:

```
s(
        (I, J, Fogos, CargasExtintor, Extintores),
        (I, NovoJ, Fogos, CargasExtintor, Extintores)
) :-
        NovoJ is J + 1,
        limiteJ(NovoJ),
        not(bloqueio(I, NovoJ)),
        andarPedra(I, NovoJ, Fogos),
        andarFogo(I, NovoJ, Fogos, CargasExtintor).
```

Mudança de estado

- Anda para esquerda:

```
s(
        (I, J, Fogos, CargasExtintor, Extintores),
        (I, NovoJ, Fogos, CargasExtintor, Extintores)
) :-
        NovoJ is J - 1,
        limiteJ(NovoJ),
        not(bloqueio(I, NovoJ)),
        andarPedra(I, NovoJ, Fogos),
        andarFogo(I, NovoJ, Fogos, CargasExtintor).
```

Mudança de estado

- Desce escada:

```
s(
        (I, J, Fogos, CargasExtintor, Extintores),
        (NovoI, J, Fogos, CargasExtintor, Extintores)
) :-
        NovoI is I + 1,
        escada(I, J),
        limiteI(NovoI).
```

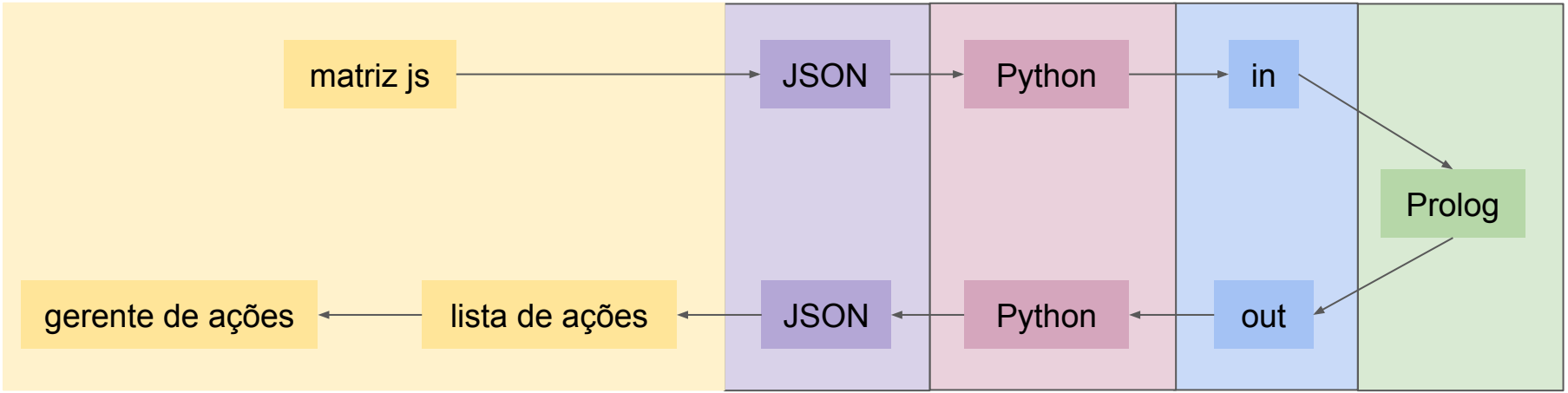
Mudança de estado

- Sobe escada:

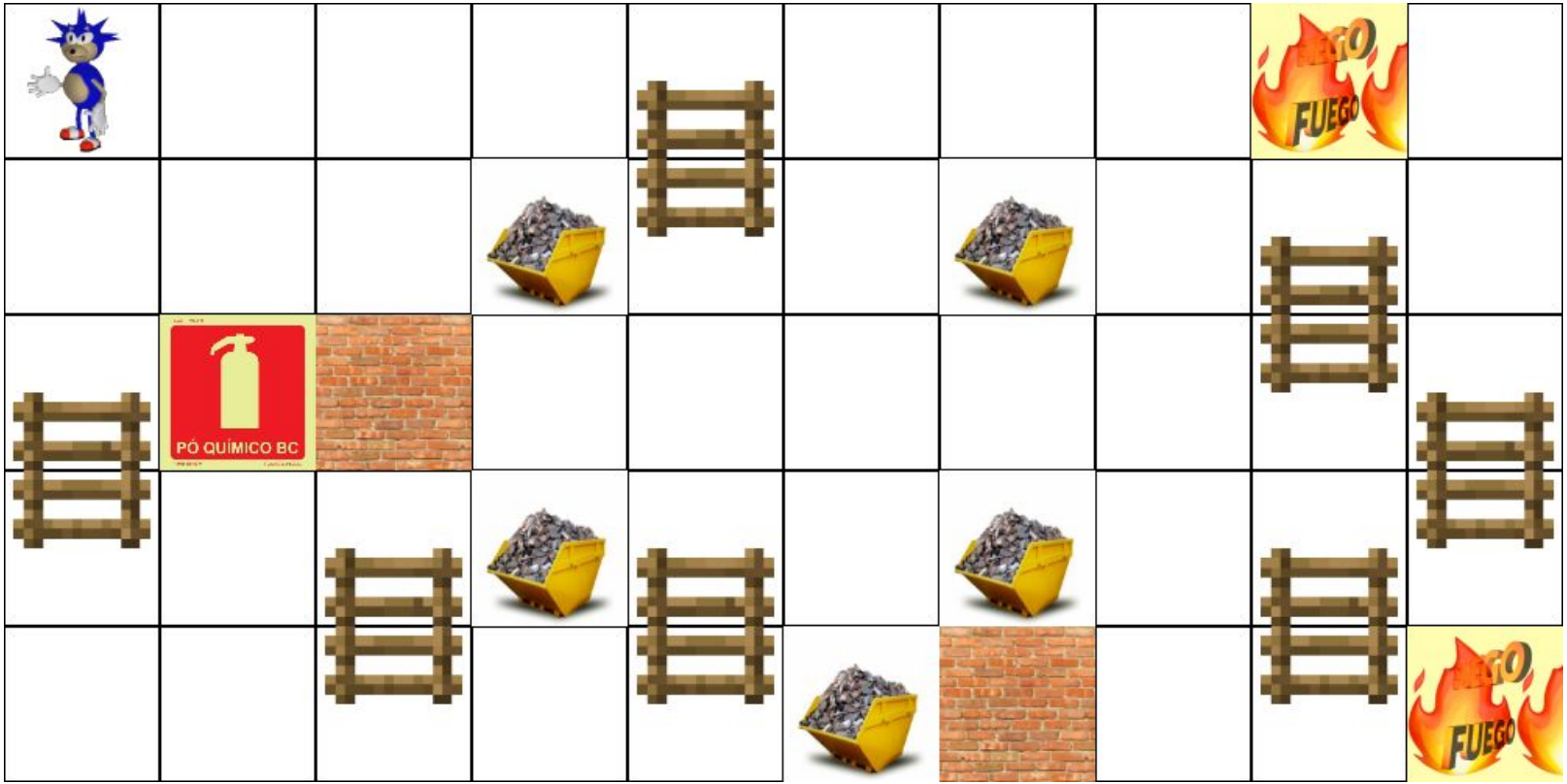
```
s(
        (I, J, Fogos, CargasExtintor, Extintores),
        (NovoI, J, Fogos, CargasExtintor, Extintores)
) :-
        NovoI is I - 1,
        escada(NovoI, J),
        limiteI(NovoI).
```

Representação gráfica

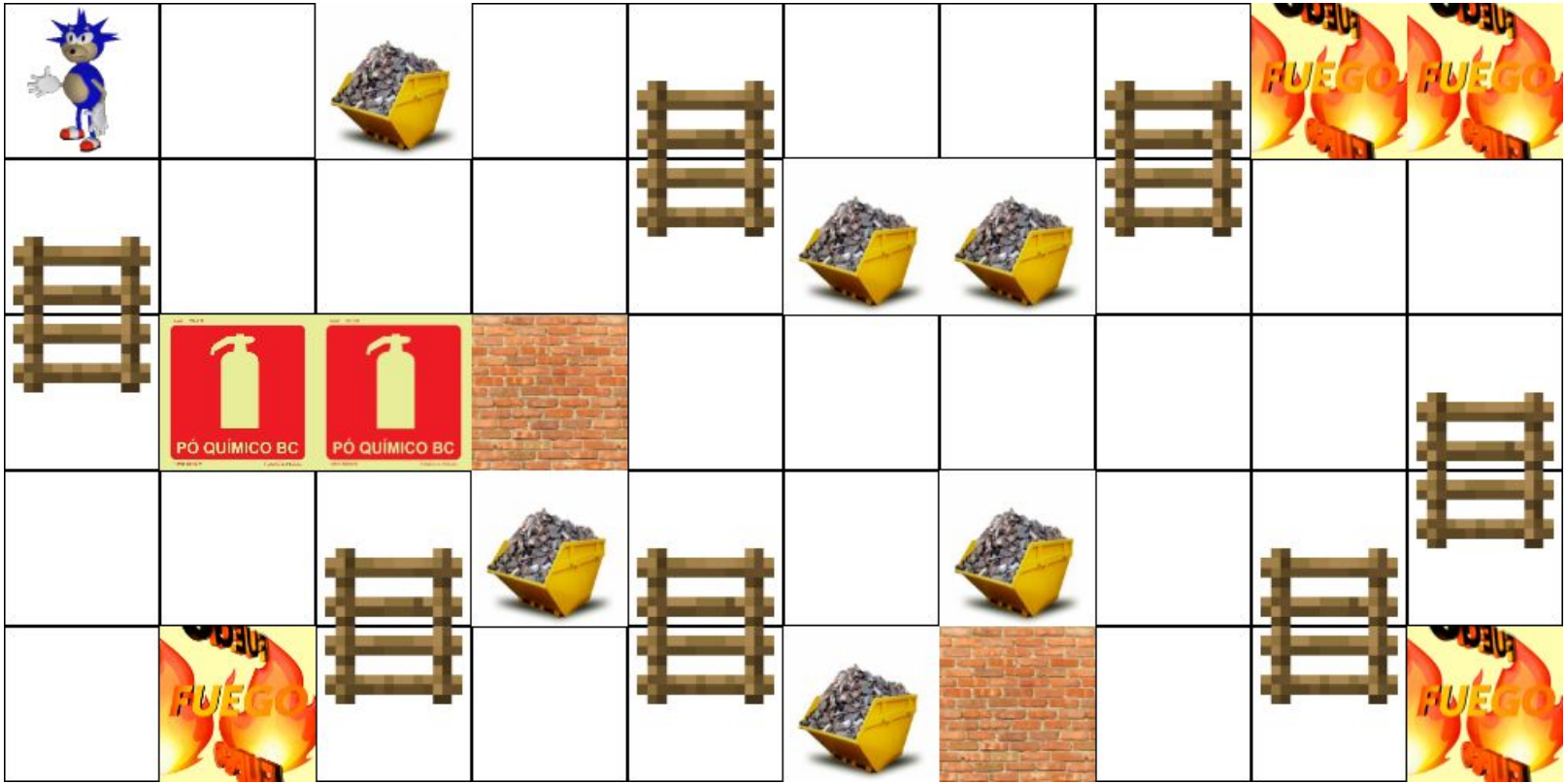
- Javascript (p5.js) e Python (Flask).



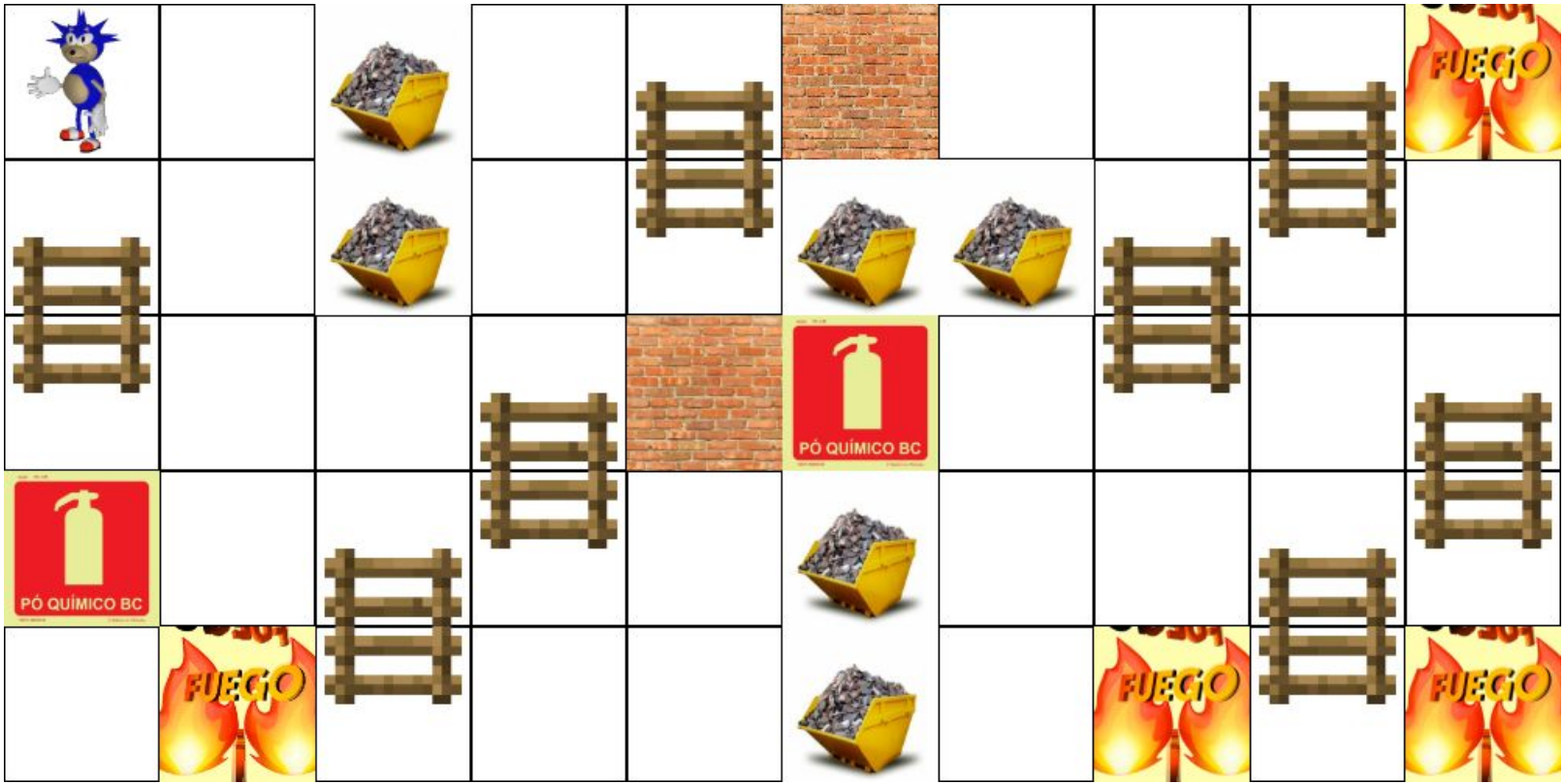
Caso de teste 1



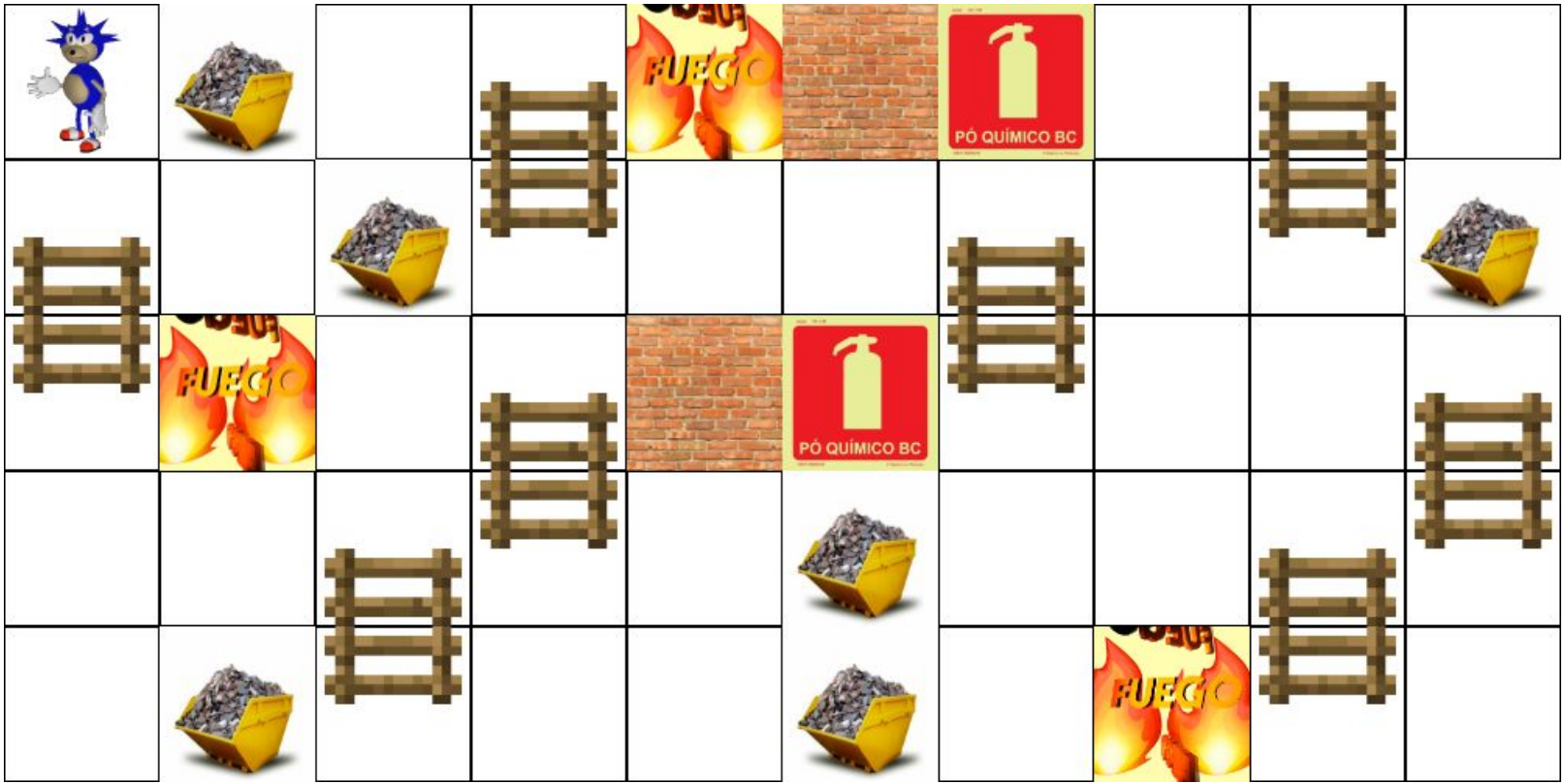
Caso de teste 2



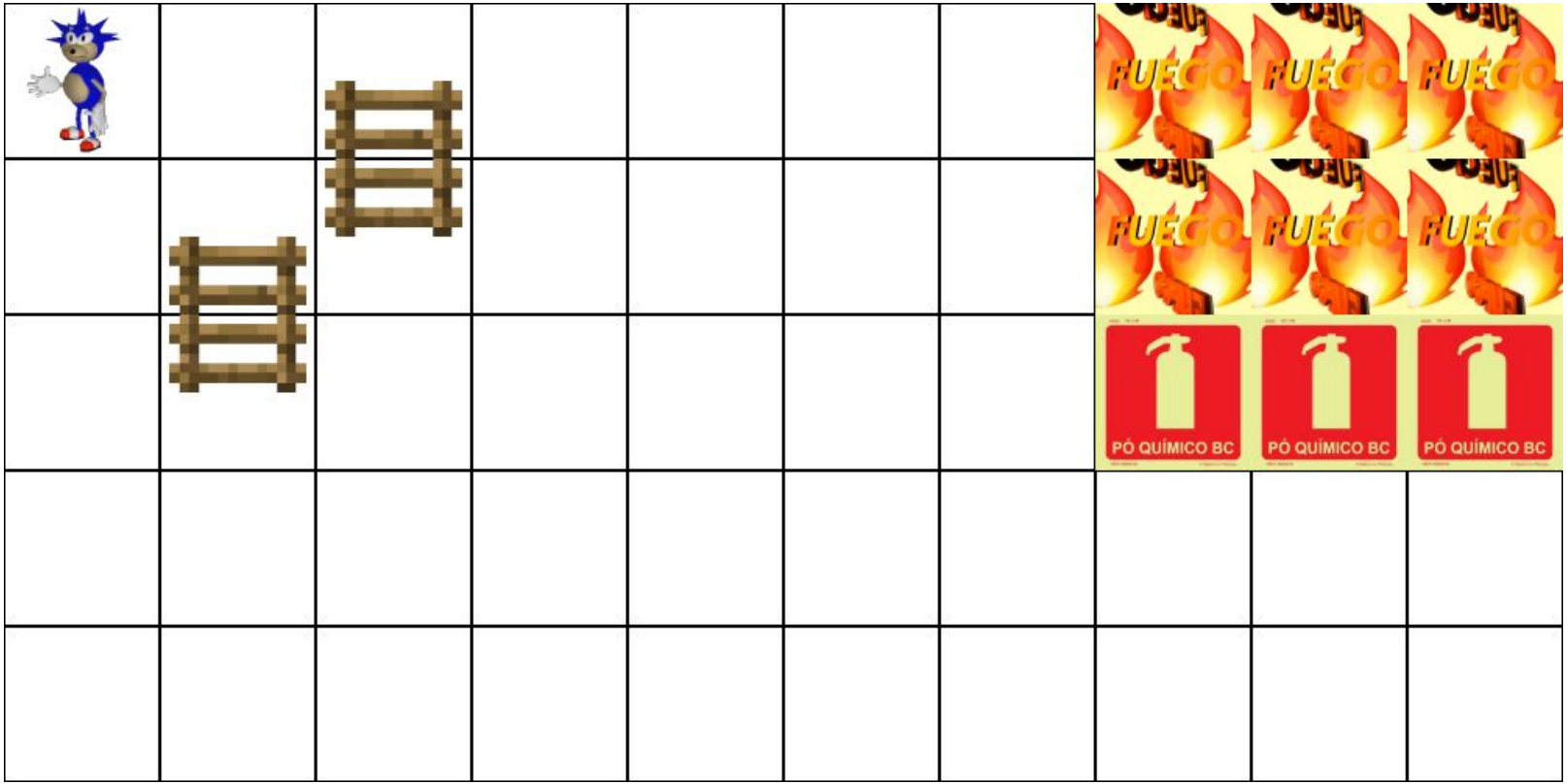
Caso de teste 3



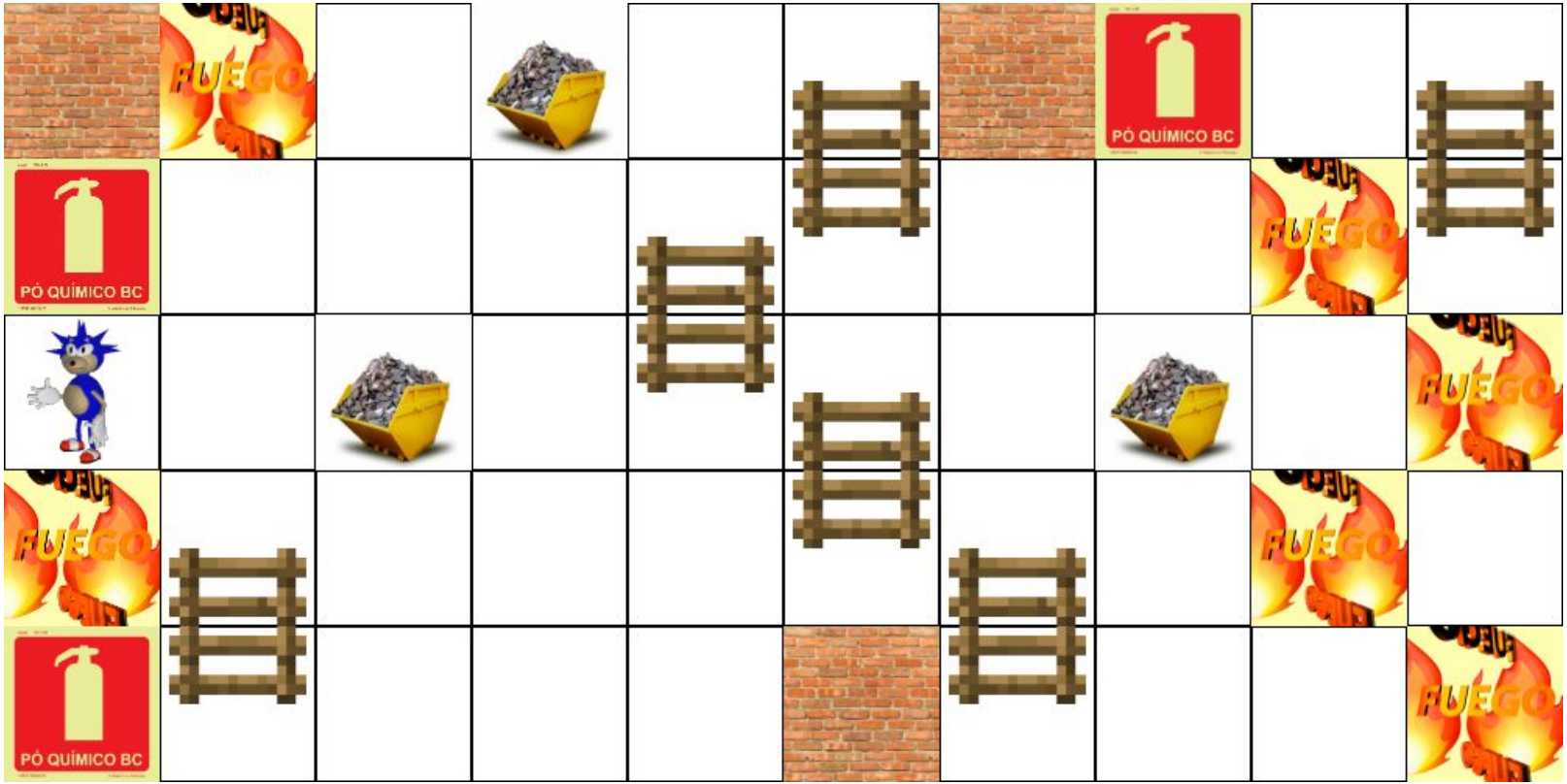
Caso de teste 4



Caso de teste 5



Caso de teste 6



Imagens e gifs