

# **Quebra-Cabeça das 8 peças**

**Integrantes**  
**Caio Cezar e Phillippe Souza**

**Juiz de Fora**  
**2019**

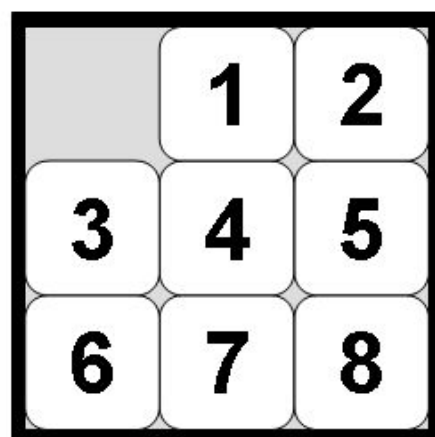
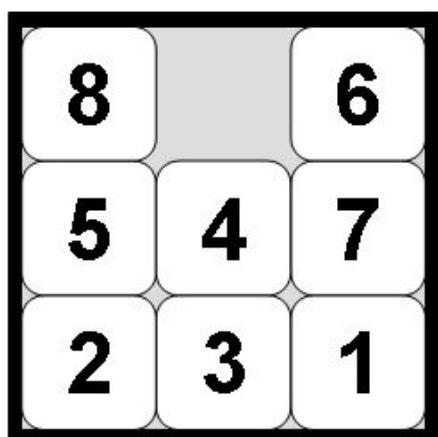
# Sumário

<b>Introdução</b>	<b>3</b>
Apresentação do problema	3
Estado inicial	3
Descrição das ações possíveis	3
Modelo de transição	3
Teste de objetivo	3
Custo de caminho	3
<b>Métodos</b>	<b>4</b>
A*(A-Estrela)	4
Estados e bordas	4
Estados	4
Bordas	4
Bibliotecas	4
Random	4
Datetime	4
OS	4
Heurística	4
<b>Resultados</b>	<b>5</b>
1ª Execução	5
2ª Execução	5
3ª Execução	6
<b>Conclusão</b>	<b>6</b>
Dificuldades	6

# Introdução

## Apresentação do problema

Dado um tabuleiro com 8 peças numéricas e uma peça vazia, em uma ordem aleatória, aplicar uma série de ações que resultem no tabuleiro com a configuração da figura ao lado. A primeira peça deve ser a vazia e as demais deverão conter as peças numéricas ordenadas crescentemente.



## Estado inicial

Qualquer estado pode ser designado como o estado inicial.

## Descrição das ações possíveis

Movimentos do quadrado vazio: Esquerda, Direita, Para Cima ou Para Baixo.

## Modelo de transição

Dado um estado e ação, devolve o estado resultante.

## Teste de objetivo

Verifica se o estado corresponde à configuração de estado objetivo.

## Custo de caminho

Cada passo custa 1. Custo do caminho é o número de passos.

# Métodos

## A\*(A-Estrela)

A \* (conhecido como "A-estrela") é um algoritmo de busca de caminho e de gráfico , que é frequentemente usado na ciência da computação devido à sua completude, otimalidade e eficiência ideal. Uma grande desvantagem prática é sua complexidade do espaço, pois armazena todos os nós gerados na memória. Assim, em sistemas práticos de roteamento de viagem, geralmente é superado por algoritmos que podem pré-processar o gráfico para obter melhor desempenho, bem como abordagens limitadas à memória; no entanto, A \* ainda é a melhor solução em muitos casos.

## Estados e bordas

### Estados

Os estados foram representados através das setas que são escritas no console.

### Bordas

Cima, baixo, direita e esquerda, respeitando o tamanho do tabuleiro.

## Bibliotecas

### Random

Foi escolhido essa biblioteca pois, queríamos que o computador escolhesse os números aleatoriamente.

### Datetime

Nós serviu para pegar a hora inicial e final de execução, assim fazendo com que tenhamos o tempo de execução do algoritmo.

### OS

Utilizamos esta biblioteca para limpar o console.

## Heurística

O algoritmo vai comparar as posições das peças do tabuleiro com as posições das peças da solução. Quanto mais posições corretas melhor.

## Resultados

### 1ª Execução

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: Python

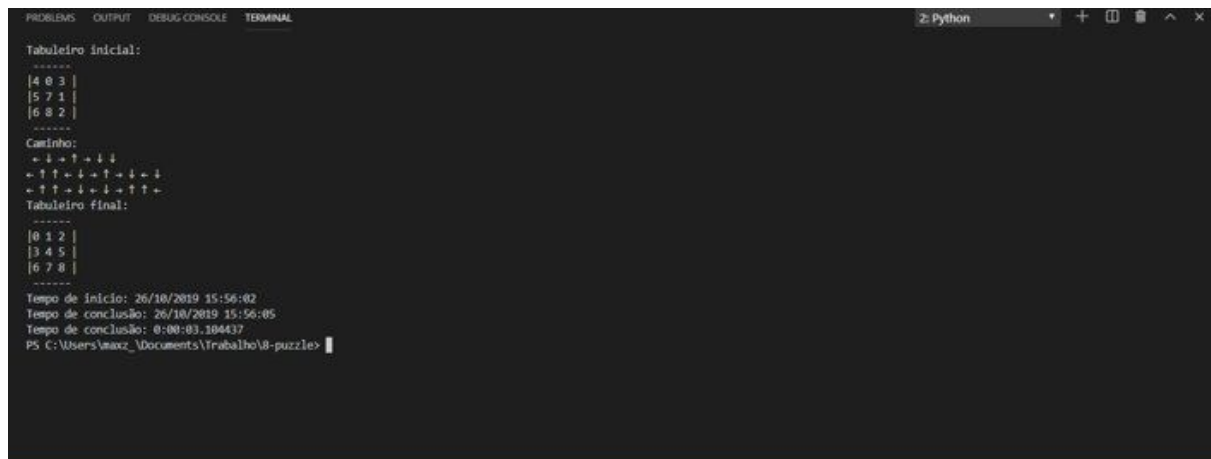
Tabuleiro inicial:
-----
| 5 1 3 |
| 8 0 6 |
| 4 2 7 |
-----
Caminho:
i -> t -> f
+ i -> f -> i -> t -> i ->
f -> i -> t -> i -> f ->
i -> f -> i -> i -> t ->
+ i -> t -> i -> i -> t ->
i -> f -> i -> f -> i -> f
Tabuleiro final:
-----
| 0 1 2 |
| 3 4 5 |
| 6 7 8 |
-----
Tempo de início: 26/10/2019 15:38:44
Tempo de conclusão: 26/10/2019 15:38:57
Tempo de conclusão: 0:00:13.213046
PS C:\Users\maxz_\Documents\Trabalho\8-puzzle>
```

### 2ª Execução

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
2: Python

Tabuleiro inicial:
-----
| 0 4 6 |
| 1 7 2 |
| 5 8 3 |
-----
Caminho:
i -> f -> i -> i -> f -> f
+ i -> i -> f -> f -> i -> f
+ i -> i -> f -> i -> f ->
+ i -> f -> i -> i -> f ->
Tabuleiro final:
-----
| 0 1 2 |
| 3 4 5 |
| 6 7 8 |
-----
Tempo de início: 26/10/2019 15:42:33
Tempo de conclusão: 26/10/2019 15:42:39
Tempo de conclusão: 0:00:05.733907
PS C:\Users\maxz_\Documents\Trabalho\8-puzzle>
```

### 3ª Execução



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Python

Tabuleiro inicial:
-----
| 4 0 3 |
| 5 7 1 |
| 6 8 2 |
-----
Caminho:
+ ↑ + ↑ + ↓ ↓
+ ↑ ↑ + ↓ + ↑ + ↓ + ↓
+ ↑ ↑ + ↓ + ↓ + ↑ ↑ +
Tabuleiro final:
-----
| 0 1 2 |
| 3 4 5 |
| 6 7 8 |
-----
Tempo de início: 26/10/2019 15:56:02
Tempo de conclusão: 26/10/2019 15:56:05
Tempo de conclusão: 0:00:03.104437
PS C:\Users\maxz\Documents\Trabalho\8-puzzle>
```

### Conclusão

Após a criação do código, o script em python consegue chegar ao resultado do tabuleiro desejado utilizando o algoritmo A\*. O script armazena os tabuleiros com um ponteiro para que seja possível percorrer de uma folha até seu ponto inicial.

### Dificuldades

1. Tivemos dificuldades de representar, a parte em que mostra o caminho percorrido pelo algoritmo;
2. Necessitou bastante de pesquisas para entender de fato como o algoritmo se comporta;
3. Contudo o objetivo de solucionar o tabuleiro gerado aleatoriamente foi alcançado.