UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE COMPUTAÇÃO

MAICON DEQUICY DE SOUZA COSTA
PAULA BEATRIZ LIMA DOS SANTOS
THIAGO DA SILVA MARTINS

TRABALHO PRÁTICO 01 - SOLUÇÃO DO QUEBRA CABEÇA DE 8 PEÇAS UTILIZANDO FILA DE PRIORIDADE

MANAUS
2023

MAICON DEQUICY DE SOUZA COSTA
PAULA BEATRIZ LIMA DOS SANTOS
THIAGO DA SILVA MARTINS

TRABALHO PRÁTICO 01 - SOLUÇÃO DO QUEBRA CABEÇA DE 8 PEÇAS UTILIZANDO FILA DE PRIORIDADE

Trabalho Prático 01 referente à disciplina ICC003 - Algoritmos e Estruturas de Dados II do Instituto de Computação da Universidade Federal do Amazonas.

Professor: Rafael Giusti

MANAUS

2023

SUMÁRIO

Introdução..……...…….……03

1. **8-Puzzle…….....……..……04**
2. **Procedimento de busca.…….……06**
3. **Interface.……..……...…….08**
4. **Instruções..…….....…….……09**

Introdução

O relatório apresentado tem como objetivo analisar um problema de busca e realizar sua solução utilizando fila de prioridades, a fim de pôr em prática os conhecimentos obtidos em sala de aula ao longo do período.

O problema escolhido foi o quebra-cabeça de 8 peças, também conhecido como *8-puzzle*. E vale ressaltar que, junto ao relatório, será submetido um programa em python que, além de possuir uma interface gráfica para o problema, ficará encarregado de resolvê-lo.

8 Puzzle

O jogo

O 8 Puzzle é um jogo no estilo quebra-cabeça. É composto de um tabuleiro 3x3, oito peças e um espaço vazio. O objetivo do jogador é encaixar as peças, que inicialmente estão embaralhadas, na ordem correta. O jogo possui algumas regras:

- A peça vazia se movimenta;
- São quatro movimentos básicos: direita,esquerda, cima e baixo;
- Dependendo da posição da peça vazia, nem todos os movimentos são possíveis.

O algoritmo A*

O algoritmo utilizado para resolver o quebra-cabeça foi o A* (A estrela). Para implementação do algoritmo, foram utilizados oito elementos:

- Estado: no caso do 8 Puzzle, entende-se por estado o novo conjunto de peças que cada movimento forma. No exemplo abaixo, o primeiro tabuleiro representa um estado, e ao movimentar a peça de número cinco, geramos um novo.

Figura 1. Exemplo de transição de estado

- Estado final: é o objetivo final do jogo. É o tabuleiro em que as peças estão ordenadas de forma crescente e o espaço vazio ocupa a primeira posição;

- Função de custo: uma função f(x) = g(x) + h(x), utilizada para organizar a estrutura da fila de prioridades, ou seja, o elemento com o menor custo fica na primeira posição;

- Função g(x): usada para calcular o número de passos de um estado até o outro;

- Função h(x): um palpite do número de passos mínimos para chegar até o estado final. Nesse caso, h(x) é o número de peças fora do lugar, com exceção da peça vazia;

- Função de transição: são os estados gerados a partir de outro estado. As transições são feitas a partir da movimentação para baixo, para cima, para esquerda e para a direita de uma peça.

- Fila de prioridades: utilizada para organizar os estados de acordo com a função de custo.

- Estados passados: é um conjunto de estados já visitados. É utilizado para não passarmos pelos mesmos novamente.

Procedimento de Busca

A implementação do algoritmo de busca para a solução do problema foi feita a partir da teoria das Heaps. Heaps são estruturas de dados especiais baseadas em árvores, onde os elementos dispostos em um array, formam uma árvore binária balanceada. Para esse projeto, foram utilizados os conceitos de Heap Mínima.

Uma Heap Mínima, é uma heap onde o menor elemento sempre é o primeiro da lista, e seus filhos são sempre maiores que ele. Os filhos de um elemento em uma heap mínima devem seguir a seguinte regra, considerando i o índice do valor na heap, e h[i] sendo o nó pai:

$$h [i] < h [(2 * i) + 1]$$

$$h [i] < h [(2 * i) + 2]$$

Sendo [(2 * i) + 1] o índice do filho esquerdo e [(2 * i) + 2] o índice do filho direito daquele índice i.

Além disso, heaps são ótimas estruturas para controles de filas de prioridades, ordenação e até para achar o menor caminho de um ponto ao outro, e é por isso que é uma ótima ferramenta para a resolução do problema 8-puzzle.

Como ponto de partida, deve ser usado o estado inicial do tabuleiro do jogo, e a partir dele, simular o possível posicionamento das peças para as transições disponíveis. Exemplo:

Estado inicial (h):

3	1	2
4	7	5
6	8	0

Movendo para baixo (h1):

3	1	2
4	7	0
6	8	5

Movendo para direita (h2):

3	1	2
4	7	5
6	0	8

Vale ressaltar que como no estado inicial, o valor zero está em uma quina, o movimento para esquerda e o movimento para cima não são possíveis. Sendo assim, os casos possíveis são adicionados na heap. O valor atribuído a eles, é baseado na quantidade de passos feitos mais a soma do módulo da diferença entre o valor de cada quadrado e o seu índice, por exemplo, em h2, foi realizado apenas um passo até o estado atual, e a soma da distância de cada posição é:

| 3 - 0 | + | 1 - 1 | + | 2 - 2 | + | 4 - 3 | + | 7 - 4 | + | 5 - 5 | + | 6 - 6 | + | 7 - 0 | + | 8 - 8| = 14

somado a um, que é a quantidade de transições realizadas a partir do estado inicial, o valor atrelado ao estado h2 é 15. Obtido esse valor, a tabela é adicionada a heap do solucionador, usando o valor quinze como chave.

Após adicionar, novamente deve ser pego o primeiro valor da heap, retirá-lo e novamente achar suas possíveis transições, até que, o primeiro valor selecionado seja o estado final [0, 1, 2, 3, 4, 5, 6, 7, 8]. Além disso, os estados que geram novas tabelas são retirados da heap principal e adicionados em uma lista de estados proibidos, fazendo com que haja uma verificação de que cada novo estado já não foi consultado, ou seja, não está na lista de estados proibidos.

Interface

Para a construção da interface gráfica do jogo “Quebra-cabeças de 8 peças”, o grupo discutiu acerca de qual seria a biblioteca mais indicada, chegando a utilizar o Pygame em uma primeira implementação, entretanto, após a realização de análises e discussões sobre o seu uso, optamos em utilizar a biblioteca Curses para a composição da interface gráfica.

A interface dispõe de um menu simples com três opções de escolha, sendo elas: resolver, embaralhar e sair. Também é possível visualizar a disposição do tabuleiro do jogo, na figura abaixo, temos a representação da tela inicial do jogo (figura 1).

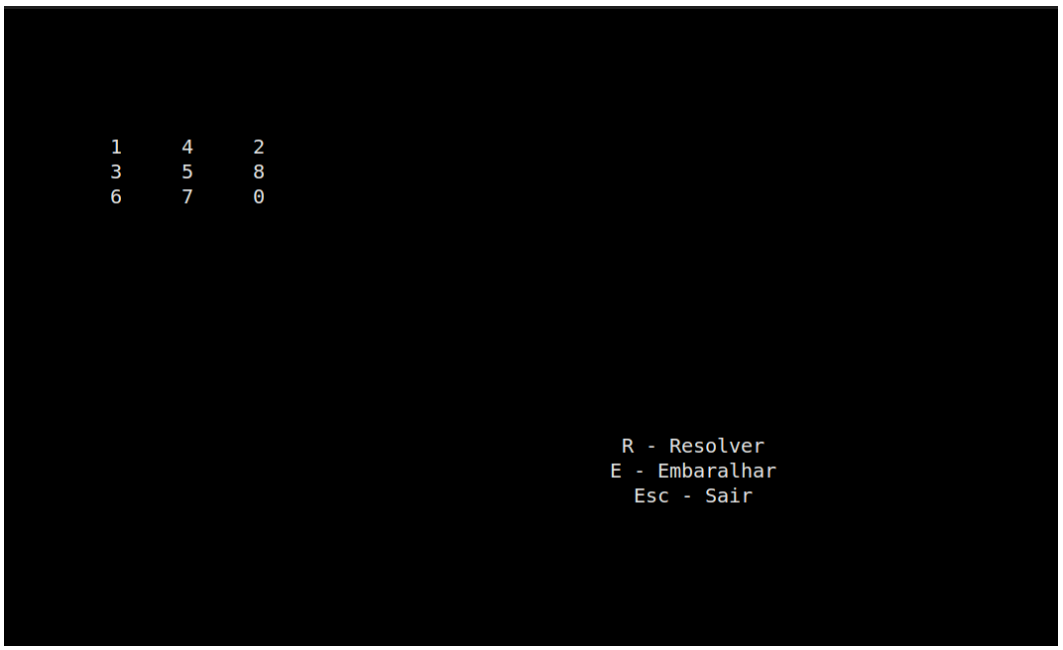

Figura 1 - Interface inicial do jogo.

Instruções

Instalação

Para começar a jogar, primeiro, o usuário deve ter o jogo em sua máquina, para isso, temos uma pequena série de passos:

1. Baixe o arquivo no git que contém o jogo;
2. Abra o terminal do seu computador;
3. Abra a pasta com o jogo baixado no github;
4. Procure pelo arquivo com o nome “curses-interface.py”;
5. Coloque o seguinte comando no terminal “python3 curses-interface.py” (sem aspas);
6. Após isso, será possível notar a interface do jogo;
7. Para saber como jogar, siga as instruções do tópico “Como Jogar” descrito logo abaixo.

Como jogar

O jogo “Quebra-cabeças de 8 peças” dispõe de um tabuleiro que tem peças numeradas de 1 a 8, existe uma posição em vazia que é mostrada no valor 0, as peças numeradas podem ser movimentadas para ocupar a posição do espaço vazio (representada nesta versão do jogo como o valor 0). O objetivo do jogo é movimentar as peças até que todas estejam em ordem crescente.

Movimentação das peças/Escolha no menu de opções

Para movimentar as peças, utilize as teclas direcionais do seu teclado (setas), cada tecla possui um movimento em específico, sendo eles:

- **UP** - movimenta o espaço vazio para a região superior;
- **DOWN** - movimenta o espaço vazio para a região inferior;
- **LEFT** - movimenta o espaço vazio para a região esquerda;
- **RIGHT** - movimenta o espaço vazio para a região direita;

Para escolher uma opção do menu, utilize as teclas referente a letra em destaque, exemplo: R - Resolver.

- **R -** Resolver;
- **E -** Embaralhar;
- **ESC -** Sair;

Tecla R

Esta tecla resolve o problema, caso ele possua solução, na figura 2 ilustra um caso solucionável.

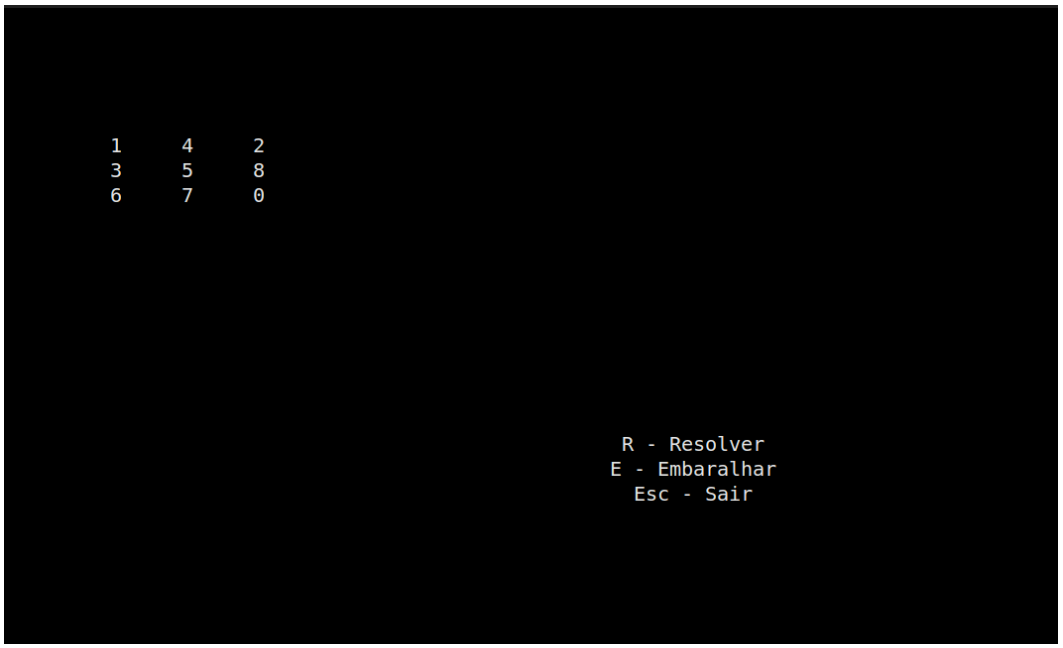

Figura 2 - Problema solucionável.

Após a aplicação da tecla R, o estado final do caso anterior é apresentado na figura 3, logo abaixo.

Figura 3 - Problema solucionado.

Tecla E

Esta tecla executa a função de embaralhar, onde as peças do tabuleiro (Figura 3) são embaralhadas e ganham novas posições no tabuleiro (Figura 4).

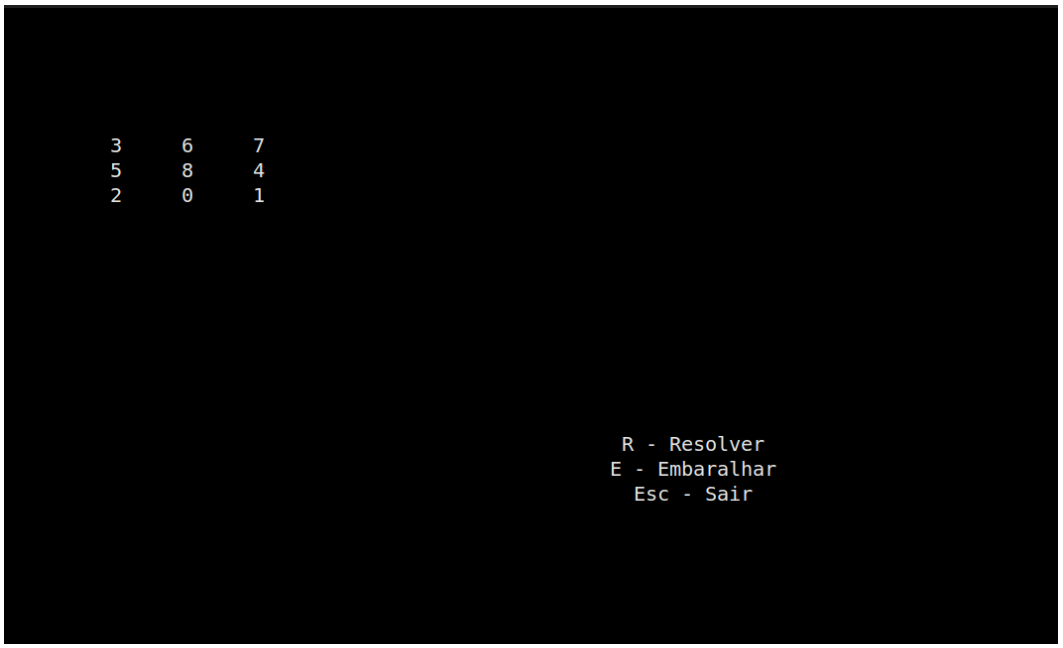

Figura 4 - Tabuleiro após aplicação da função de embaralhamento.