Exercício Busca Local – Move Pedras

O objetivo do jogo é colocar as pedras numeradas de 1 a 8 no grid abaixo de forma que não sejam sequenciais em posições adjacentes nas linhas, colunas e diagonais.

Exemplo:

Estado inicial

	2	3	
1	7	5	8
	4	6	

Violações

- 2 e 3 são sequenciais na mesma linha
- 2 e 1 são sequenciais na diagonal
- 5 e 6 são sequenciais na coluna
- 5 e 4 são sequenciais na diagonal
- 7 e 6 são sequenciais na diagonal

Exemplo de estado objetivo: não viola as restrições do problema

	5	3	
2	8	1	7
	6	4	

Objetivo

Implementar a **função objetivo/avaliação (`método avaliarSolucao() da CLASSE Tabuleiro.java`)** para o algoritmo de subida de encosta e avaliar o desempenho temporal do algoritmo em função do número de vezes que a função objetivo/avaliação é invocada. Também avaliar a convergência do algoritmo em direção à solução plotando o valor da função objetivo para o estado atual por iteração.

O programa gera dois arquivos de saída: SE_Resultados.txt (por execução) e (detalhado)

SE_Resultados.txt contém uma linha por execução, tal que cada linha contém

- Exec: número sequencial da execução;
- K-viz: parâmetro de configuração utilizado para o número de vizinhos;
- K-iter: parâmetro de configuração utilizado para o número máximo de iterações;
- Iter-fim: iteração no qual o subida de encosta parou por não ter encontrado um vizinho melhor;
- Aval: número de chamadas a função objetivo/avaliação;
- VlrObj: valor da função objetivo/avaliação no momento da parada;

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	ARQUIVO SE_Resultados.txt																	
2	exec	K-viz	K-iter	iter-fim	aval	vlrObj	_	T	A	B	U	L	E	I	R	O	_	_
3	1	5	50	2	11	3	0	1	8	0	5	6	4	3	0	2	7	0
4	2	5	50	1	6	2	0	5	1	0	3	7	2	6	0	8	4	0
5	3	5	50	1	6	3	0	6	4	0	3	2	7	1	0	5	8	0
6	4	5	50	3	16	3	0	8	6	0	4	3	5	1	0	2	7	0

O arquivo SE_MelhoresPorIter.txt contém os melhores tabuleiros (menor valor da função objetivo/avaliação) por iteração. Assim, em cada linha encontra-se:

- Exec: número sequencial da execução
- k-viz: parâmetro de configuração de número de vizinhos por estado
- k-iter: parâmetro de configuração de número de iterações por execução
- iter: número sequencial a iteração (dentro da execução)
- vlrObj: valor da função objetivo/avaliação para o tabuleiro

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	ARQUIVO		SE_MelhoresPorIter.txt														
2	exec	K-viz	K-iter	iter	vlrObj	_	T	A	B	U	L	E	I	R	O	_	_
3	1	5	50	1	3	0	1	8	0	5	6	4	3	0	2	7	0
4	1	5	50	2	3	0	1	8	0	5	6	4	3	0	2	7	0
5	2	5	50	1	2	0	5	1	0	3	7	2	6	0	8	4	0
6	3	5	50	1	3	0	6	4	0	3	2	7	1	0	5	8	0
7	4	5	50	1	4	0	8	5	0	4	3	6	1	0	2	7	0
8	4	5	50	2	3	0	8	6	0	4	3	5	1	0	2	7	0
9	4	5	50	3	3	0	8	6	0	4	3	5	1	0	2	7	0
10	5	5	50	1	3	0	5	1	0	8	3	7	2	0	4	6	0
11	5	5	50	2	3	0	5	1	0	8	3	7	2	0	4	6	0

Responda/faça:

1. Qual configuração utilizou para **resolver** (encontrar a solução) o problema com o algoritmo?
    a. Número de execuções
    b. Número de iterações por execução
    c. Número de vizinhos por estado
2. Para esta configuração faça (com os dados do arquivo SE_Resultados.txt)
    a. Plote o gráfico de execução x número de avaliações
    b. Plote o gráfico de execução x valor da função objetivo
3. Para esta configuração faça (com os dados do arquivo SE_MelhoresPorIter.txt):
    a. Escolha a execução na qual o subida de encosta encontrou a solução;
    b. Para esta execução, plote o gráfico de número da iteração x valor da função objetivo (a curva deve ser decrescente em se tratando de um problema de minimização).
4. Qual função de avaliação IMPLEMENTOU? Descreva-a em linguagem natural.
5. Como são calculados os estados vizinhos (observar o código fonte)?
6. Quais as condições de parada para uma execução do algoritmo (como está implementado no código fonte)?