CCM0118 - Computação I (2024ii)

E06 Oito ladrilhos

Vencimento: quinta-feira, 12 dez. 2024, 23:59

Neste exercício, você deve escrever um programa que resolve o Problema dos Oito Ladrilhos: Exercício 9.6 de

https://www.ime.usp.br/~yoshi/INTRO/EXERCICIOS/caderno-exercicios_2005.pdf

Você pode usar o programa do Cauê, Puzzle.java, como ponto de partida. De fato, Puzzle.java resolve a parte principal deste exercício, mas há ainda algumas coisais adicionais a serem feitas.

O programa que você vai entregar deve chamar-se Eight.java.

Modos de execução. Seu programa deve ter três modos de execução.

Modo simples. Sem nenhum argumento de linha de comando, seu programa deve receber uma configuração inicial dos ladrilhos na entrada padrão, e deve decidir se há ou não solução:

```
$ cat ex1.txt
1 3 4
8 6 2
7 0 5
$ java-introcs Eight < ex1.txt
Solvable
$ cat ex2.txt
3 1 4
8 6 2
7 0 5
$ java-introcs Eight < ex2.txt
Exception in thread "main" java.lang.StackOverflowError
at Eight.neighbour(Eight.java:38)
at Eight.solve(Eight.java:24)
at Eight.solve(Eight.java:28)
at Eight.solve(Eight.java:28)
at Eight.solve(Eight.java:28)
[...]
$ java-introcs -Xss10m Eight < ex2.txt
Not solvable
$
```

Modo que imprime a sequência de configurações. Com o argumento de linha comando 0, seu programa deve imprimir uma sequência de configurações, começando com a configuração-objetivo e chegando na configuração dada na entrada padrão. Se s é uma configuração de sua sequência e t é a configuração seguinte, então um movimento legal deve levar s a t.

```
$ java-introcs Eight 0 < ex1.txt > ex1_sol.txt
$ head ex1_sol.txt
1 2 3
8 0 4
7 6 5

1 2 3
8 4 0
7 6 5
```

?

```
1 2 0
8 4 3
$ tail ex1_sol.txt
7 5 2

1 3 4
8 6 2
7 5 0

1 3 4
8 6 2
7 0 5

$ java-introcs -Xss10m Eight 0 < ex2.txt
$
```

Note que, nas configurações impressas, o dígito 0 codifica o espaço vazio (sem ladrilho).

Modo estatístico. Suponha que escolhemos a configuração inicial uniformemente ao acaso dentre todas as possibilidades. Qual é a probabilidade dessa configuração ser solúvel? Uma forma de estimar esta probabilidade é gerar várias configurações uniformemente ao acaso e contar quantas são solúveis. Seu programa deve ter um modo de execução que recebe um inteiro $N > 0$ e uma semente $seed$ como argumentos de linha de comando. Com tal entrada, seu programa deve gerar N configurações uniformemente ao acaso, usando $seed$ como semente, e deve imprimir quantas configurações solúveis foram geradas.

```
$ java-introcs -Xss10m Eight 100 118
solved / total = 48 / 100 [48%]
$ time java-introcs -Xss10m Eight 100 118118
solved / total = 58 / 100 [58%]

real 0m5.223s
user 0m5.486s
sys 0m0.368s
$ time java-introcs -Xss20m Eight 1000 118118118
solved / total = 487 / 1000 [49%]

real 0m49.683s
user 0m54.899s
sys 0m1.957s
$
```

Verificação. Suponha que seu programa seja executado no modo que imprime a sequência de configurações encontrada. Você pode verificar se a sequência é válida usando o programa Check.java fornecido abaixo. Check.java espera que você dê como argumento de linha de comando a configuração inicial (como pode ser visto abaixo).

```
$ java-introcs -Xss20m Eight 0 < ex1.txt | java-introcs Check 134862705
Moves are legal / no. configs: 3182
Arrived at the right config: true
$ echo 0 1 2 3 4 5 6 8 7 | java-introcs -Xss20m Eight 0 | java-introcs Check 12345687
Moves are legal / no. configs: 74767
Arrived at the right config: true
$ echo 0 1 2 3 4 5 6 7 8 | java-introcs -Xss20m Eight 0 | java-introcs Check 12345678
No sequence to check
$ echo 0 1 2 3 4 5 6 7 8 | java-introcs -Xss20m Eight
Not solvable
$
```

Entrega. Entregue apenas seu programa Eight.java.

E06 6 dezembro 2024, 17:40 PM