

Escola Superior de Tecnologia de Setúbal

Inteligência Artificial - 2020/2021

Manual de utilizador

Projeto N° 1: Problema do Quatro



Realizado por

Número	Nome	Ano
140221101	João Quissenguele	3º
130221040	Izilda Kossy	3º

Docentes

- ♦ Prof. Joaquim Felipe
- ♦ Eng. Filipe Mariano

Índice

1 - Descrição do problema do Quatro	3
2 - Como iniciar à aplicação	3
2.1 - Menu Resolver Problema	6
2.2 - Menu Sair	7
3 - Como utilizar à aplicação	8
4 - Escolha de Algoritmo da Procura	9
5 - Gravação de resultados no ficheiro	13
6 - Limitações do programa	14

1. Descrição do problema de quatro

Este projeto destina-se a resolver o problema baseado no jogo do Quatro, este jogo é constituído por um tabuleiro de 4X4 casas e um conjunto de peças (até 16 dependente dos problemas) que possuem traços característicos de forma e de cor em que existem peças brancas ou pretas, parte delas são altas, a outra parte são baixas, umas são quadradas outras são redondas.

Algumas peças são cheias outras são ocas, deixando para a secção seguinte a explicação do que se pretende que seja desenvolvido no projeto de Inteligência Artificial relativo à resolução de um problema neste contexto por procura em Espaço de Estados e o objetivo é encontrar para um determinado tabuleiro, qual a solução óptima ou de menor custo para chegar o fim do jogo e para isso, é preciso completar uma linha de 4 peças (na horizontal, na vertical ou na diagonal) que compartilhem pelo menos um traço em comum (4 peças pretas, ou 4 peças altas, ou 4 peças ocas, etc).

2. Como iniciar a aplicação

Para iniciar a aplicação necessita de abrir o ficheiro **projeto.lisp** no IDE LispWorks. Uma vez aberto deverá mostrar como a figura 1.

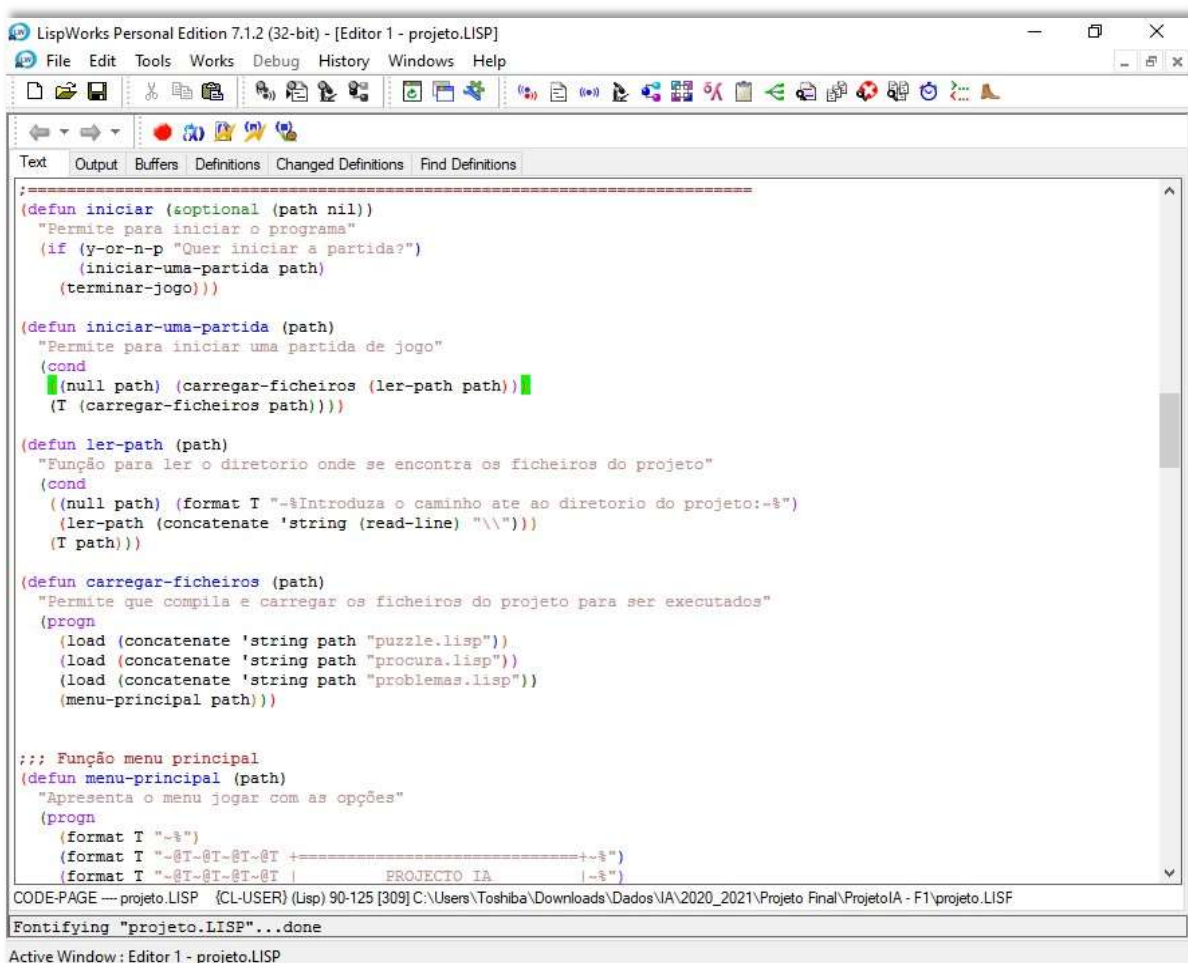



Figura 1 - interface do IDE LispWork

De modo a que consiga iniciar a aplicação, tem que primeiro compilar as suas funções, para isso carregue no 

e uma vez compilado deverá abrir a semelhança da figura 2 e para sair, só precisa clicar no espaço do teclado.

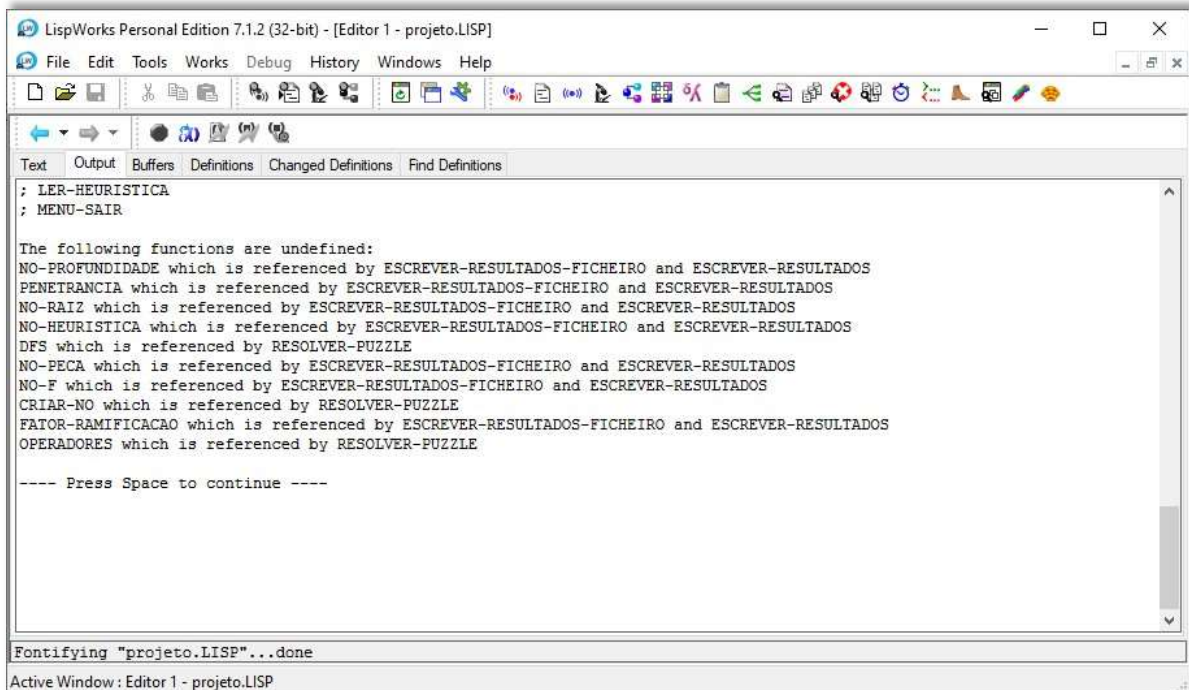


Figura 2 - resultado depois de carregar no compile buffer

E depois é só abrir o **Listener**  para começar a simular o jogo, uma vez aberto, deverá aparecer a semelhança da figura 3.

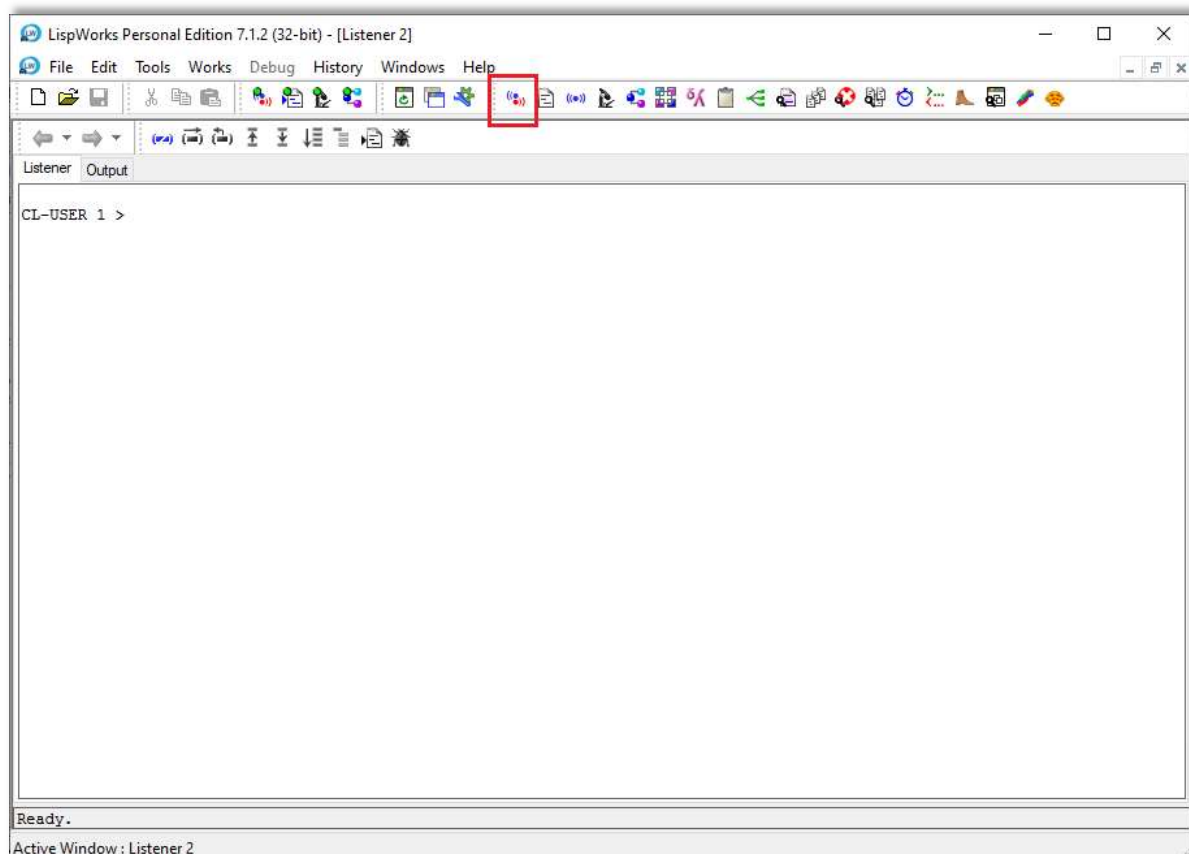


Figura 3 - interface do Listener

Agora está tudo pronto para iniciar a aplicação, no painel da Listener executa a função **(iniciar)**, depois clica no **Enter** e de seguida irá pedir para escolher **Sim** para iniciar uma partida ou **Não** para terminar o programa, como mostra a figura 4.

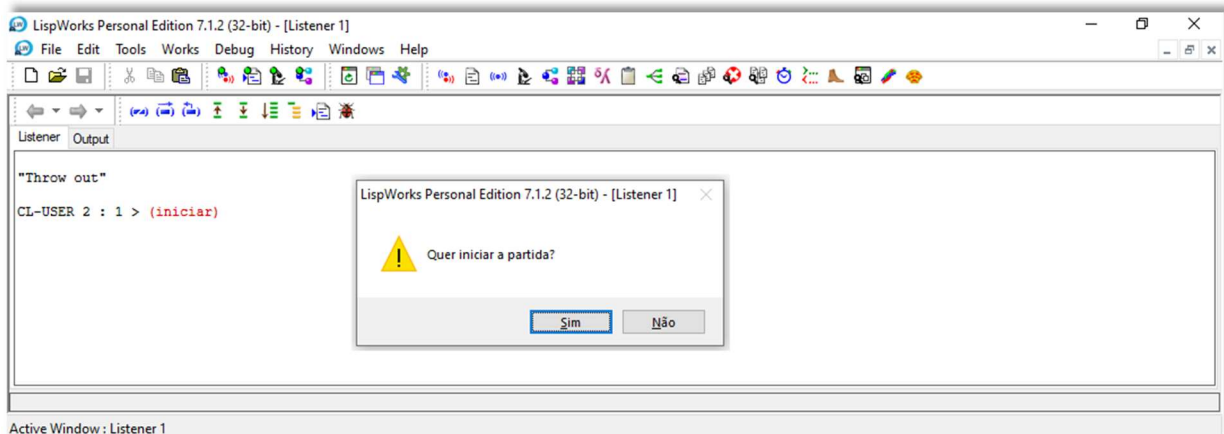


Figura 4 - escolha do inicio ou fim da aplicação

Depois de clicar **Sim**, irá ser pedido inserir o caminho para a diretoria principal da aplicação, como mostra a figura 5 e depois carregue na tecla **Enter** do teclado.

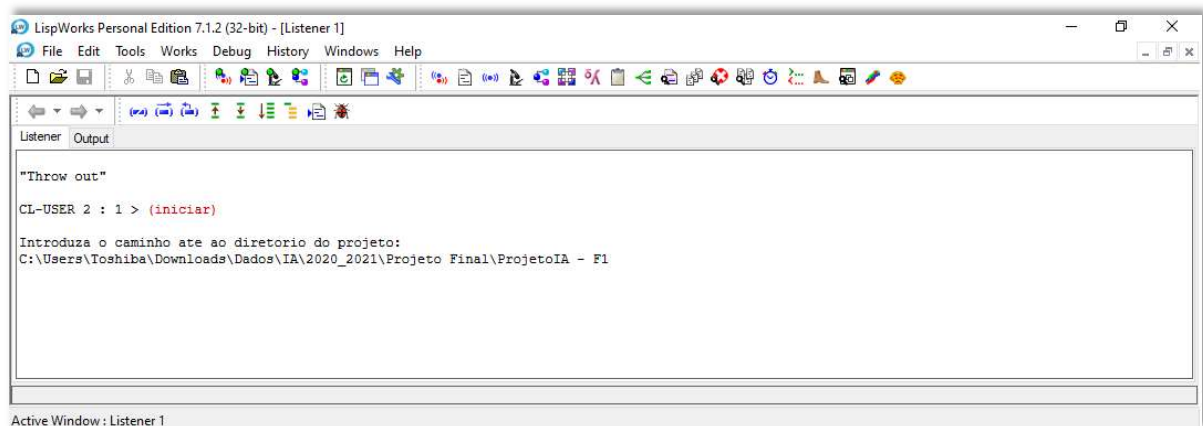


Figura 5 - inserção do caminho para o diretório do projeto

A aplicação irá carregar os ficheiros **puzzle.lisp**, **procura.lisp** e a lista de problemas **problemas.dat** com sucesso e de seguida apresenta o menu principal onde poderá utilizar à aplicação, a semelhança da figura 6.

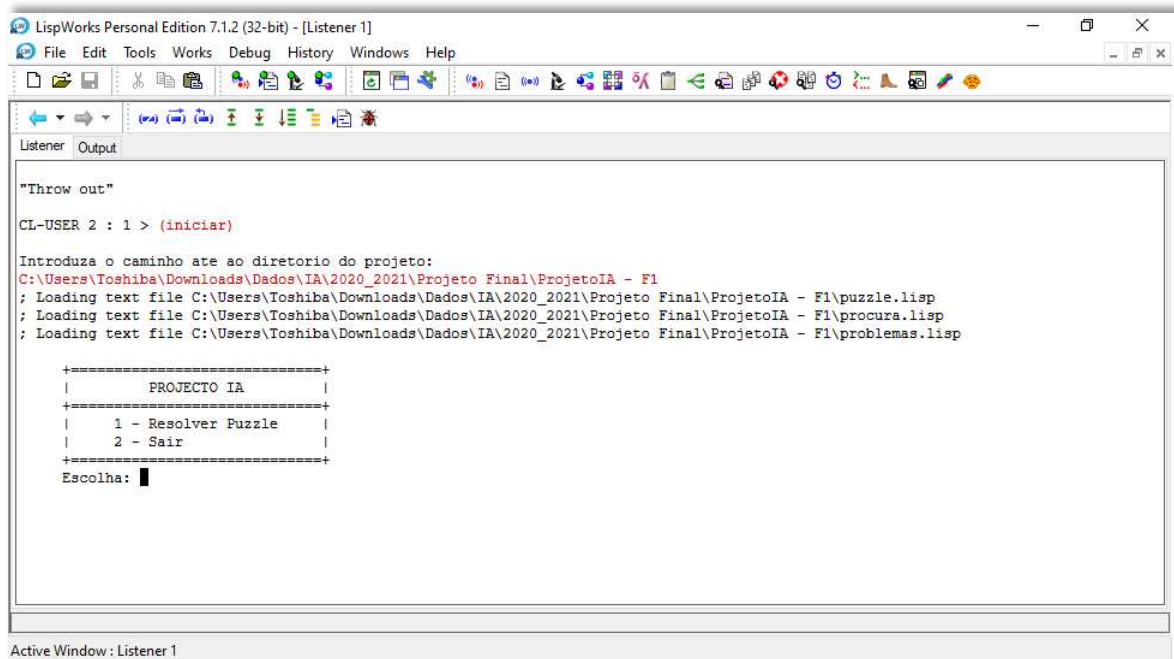


Figura 6 - carregamento dos ficheiros e visualização do menu iniciar

Neste menu principal, pode escolher uma de duas opções:

2.1 - Menu Resolver Problema

Esta opção dará início à aplicação, levando-lhe para outros menus onde tem a possibilidade de escolher qual o **tabuleiro** que deseja, qual o **algoritmo** de procura que deseja, qual a **heurística** que deseja e qual a **profundidade** máxima que deseja (caso tenha seleccionado o algoritmo *Depth First*), como mostra a figura 7.

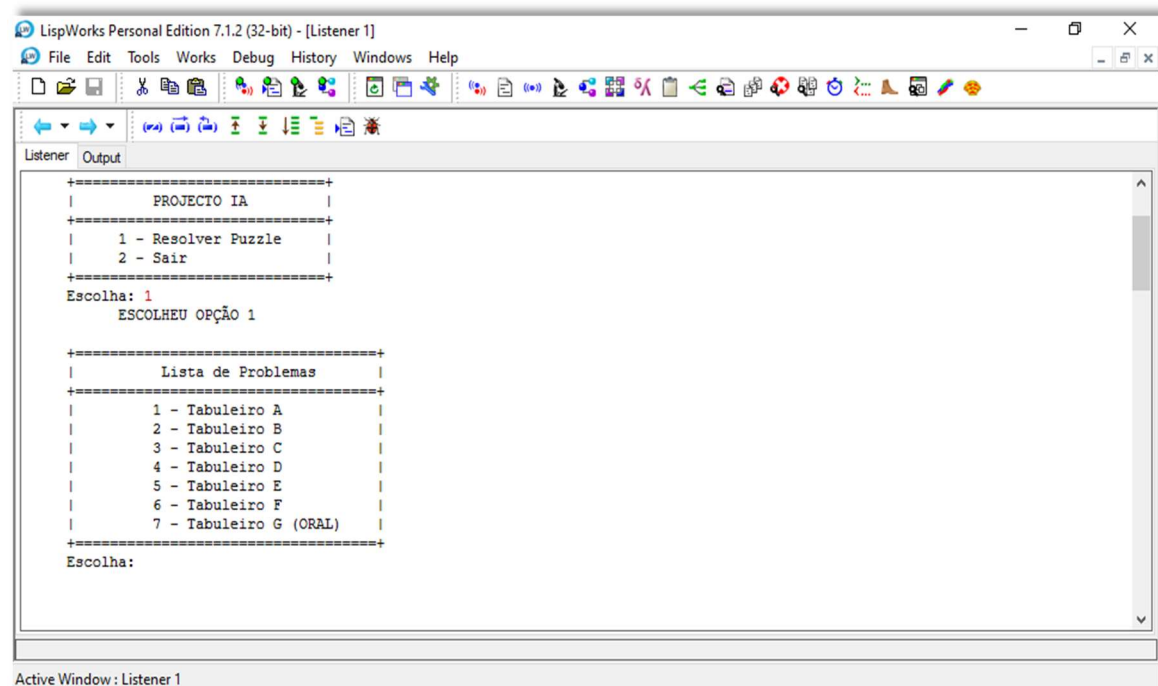


Figura 7 - interface do menu resolver problema com lista de problemas a resolver

2.2 - Menu Sair

Como pode deduzir, esta opção leva ao término da aplicação, parando a sua execução, como mostra a figura 8.

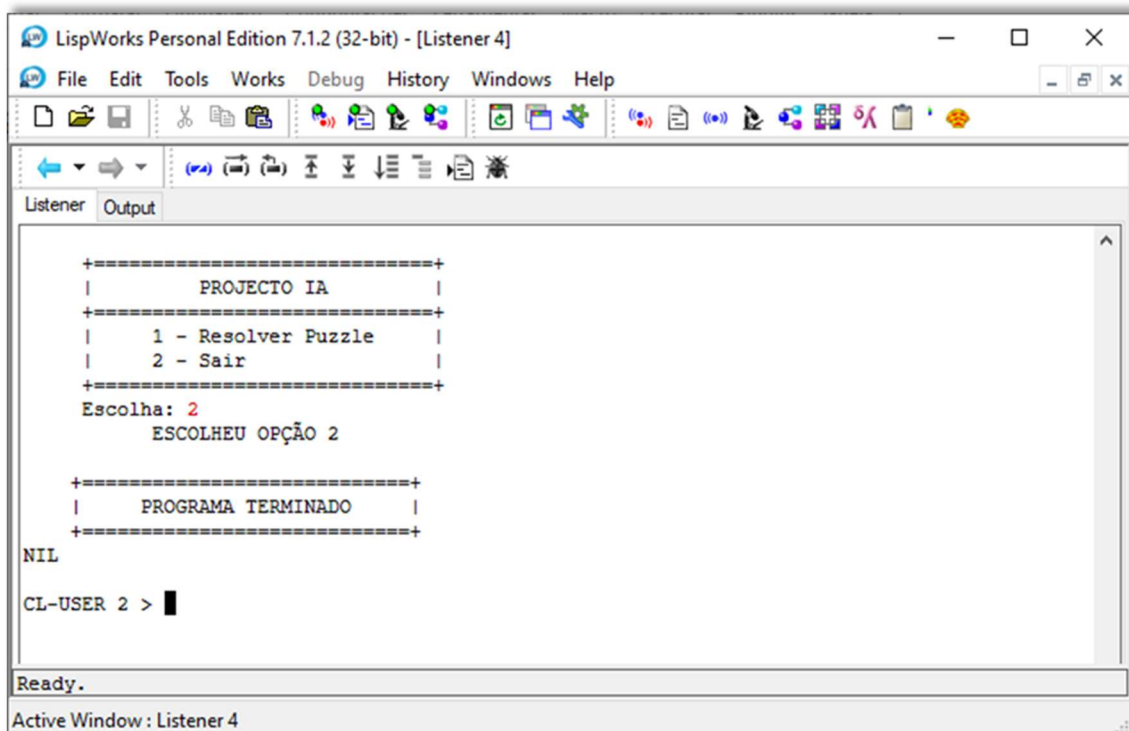


Figura 8 - interface do menu sair para terminar o programa

E se pretender iniciar novamente a aplicação é somente, seguir o mesmo procedimento já descrito na seção como iniciar a aplicação, escrevendo a função **(iniciar)** no Listener, como mostra a figura 9.

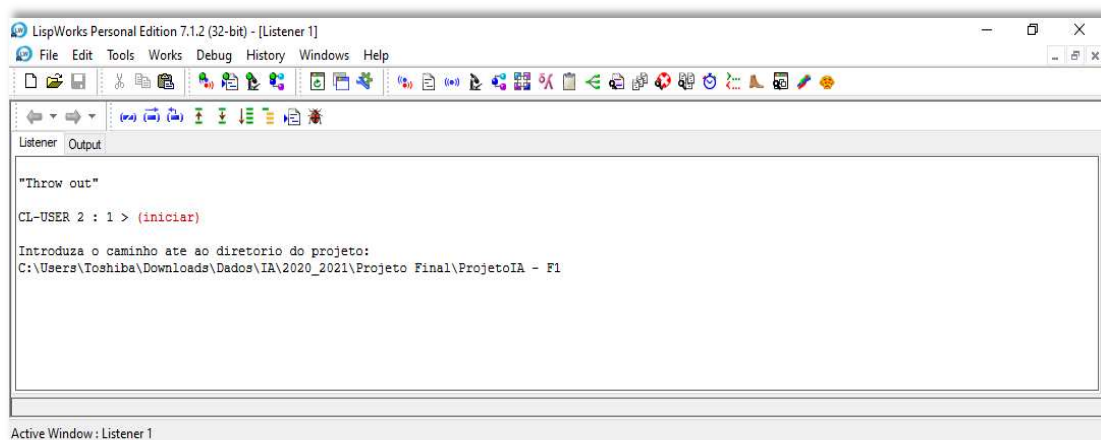


Figura 9 - iniciar a aplicação depois de terminar a aplicação

3. Como utilizar a aplicação

Escolhendo a opção 1 para **Resolver Problema** a aplicação irá mostrar outros menus onde irá escolher o tipo de tabuleiro para simular o jogo, neste problema existem 7 tipos de tabuleiro, como mostra a figura 10 e figura 11.

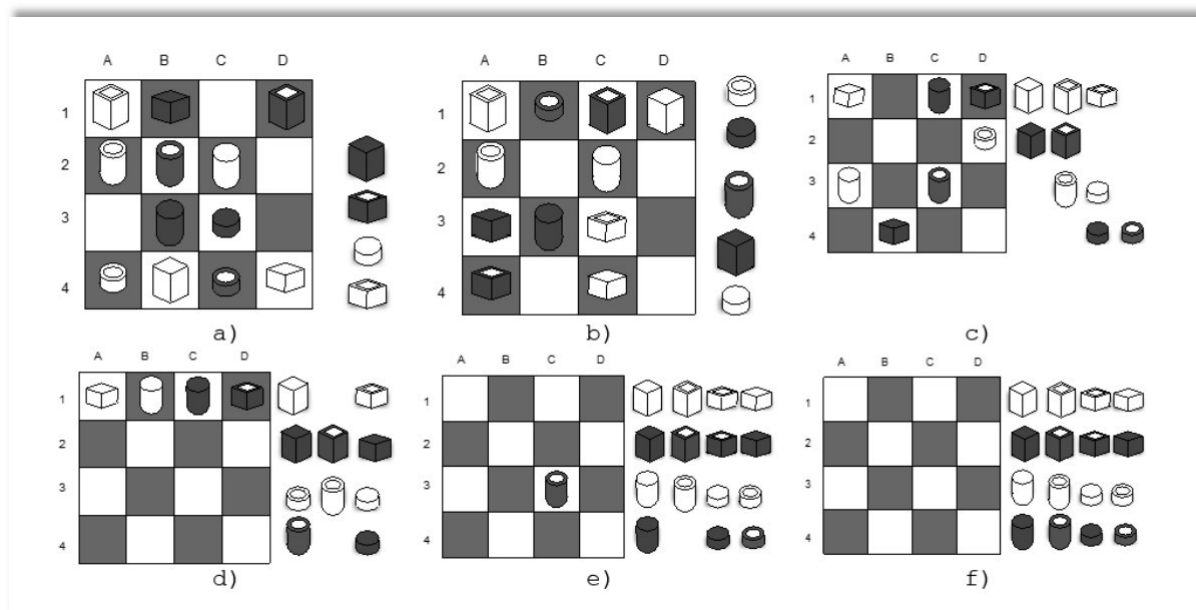


Figura 10 - representação gráfica dos problemas a resolver

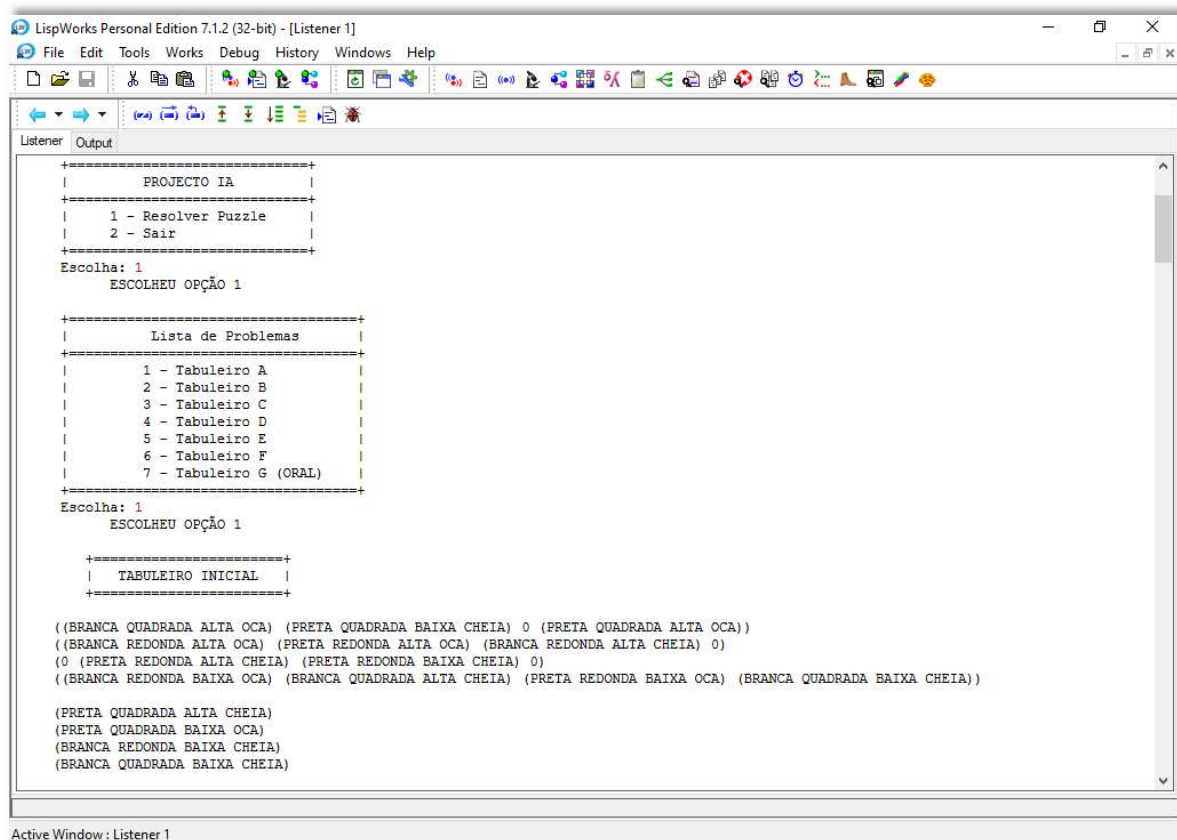


Figura 11 - representação dos problemas a resolver em modo console

4. Escolha de algoritmos da procura

Após a escolha do tabuleiro, o utilizador poderá escolher o algoritmo que deseja utilizar para resolver o problema, como mostra a figura 12.

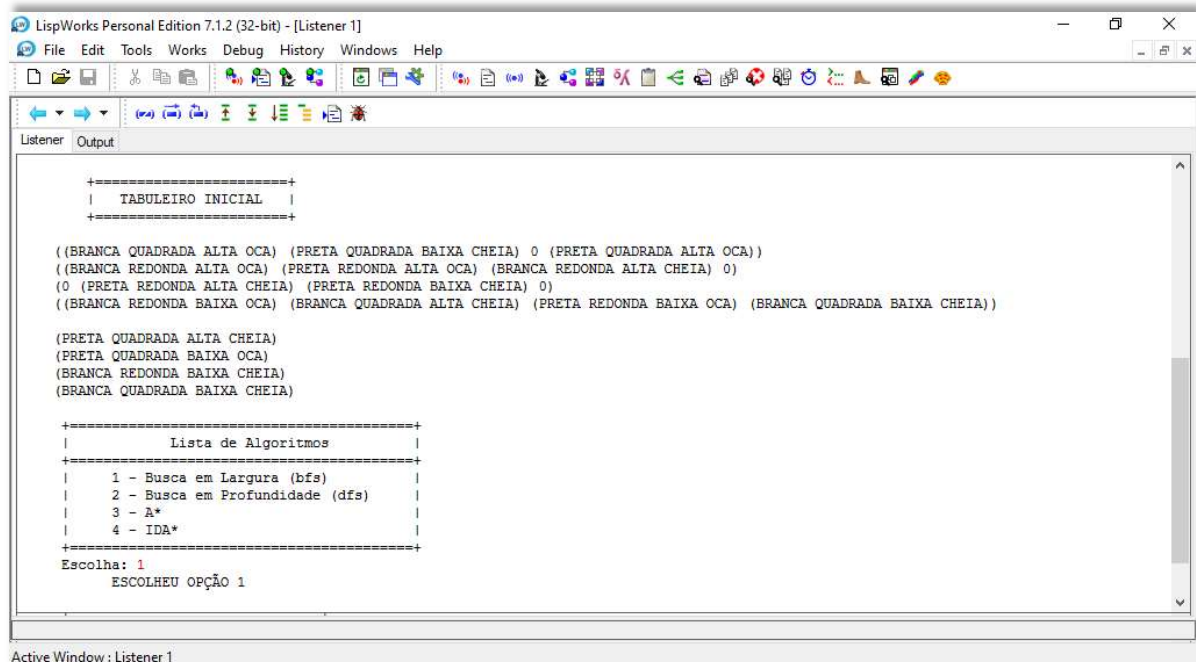


Figura 12 - interface para escolher tipo de algoritmo

Caso escolha o algoritmo **bfs**, o programa irá apresentar logo os resultados para o tabuleiro escolhido, como mostra a figura 13.

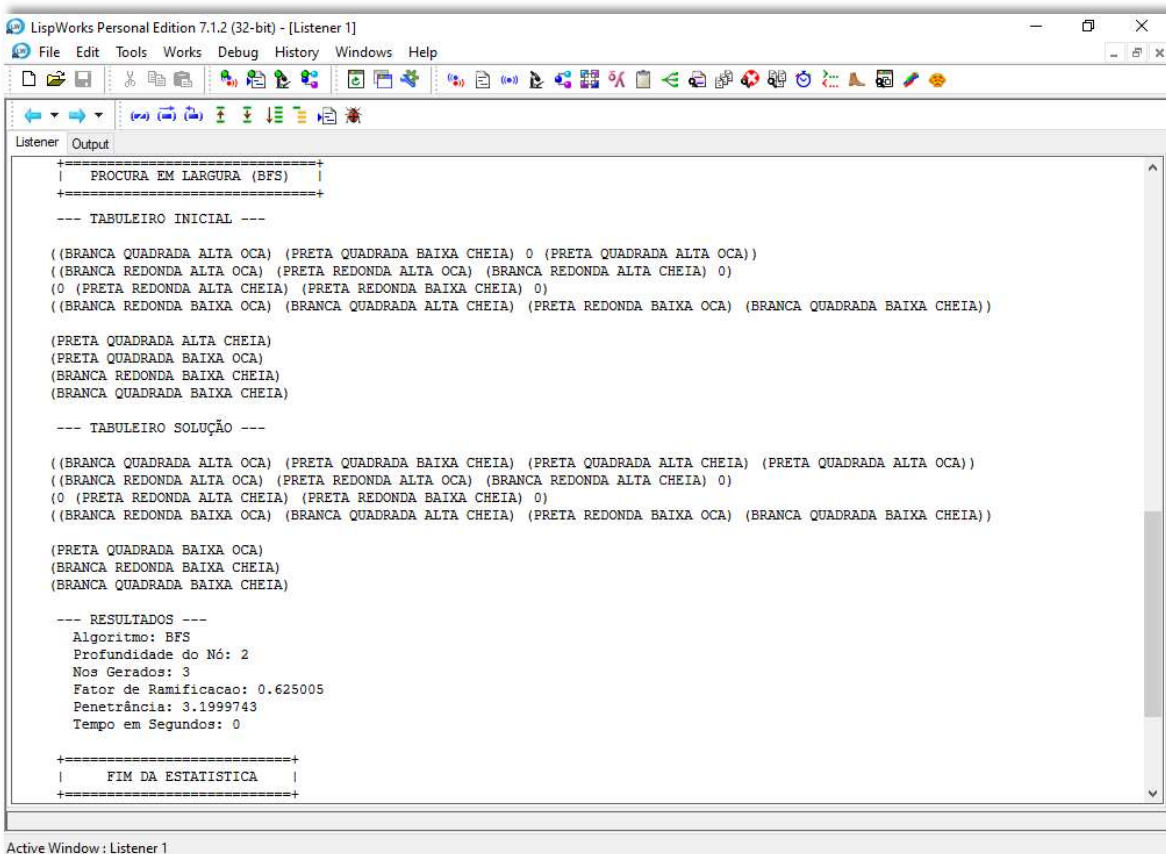


Figura 13 - resultado do problema resolvido com algoritmo bfs

Caso escolha o algoritmo **dfs**, o programa irá pedir a introdução da profundidade máxima para este algoritmo, como mostra a figura 14.

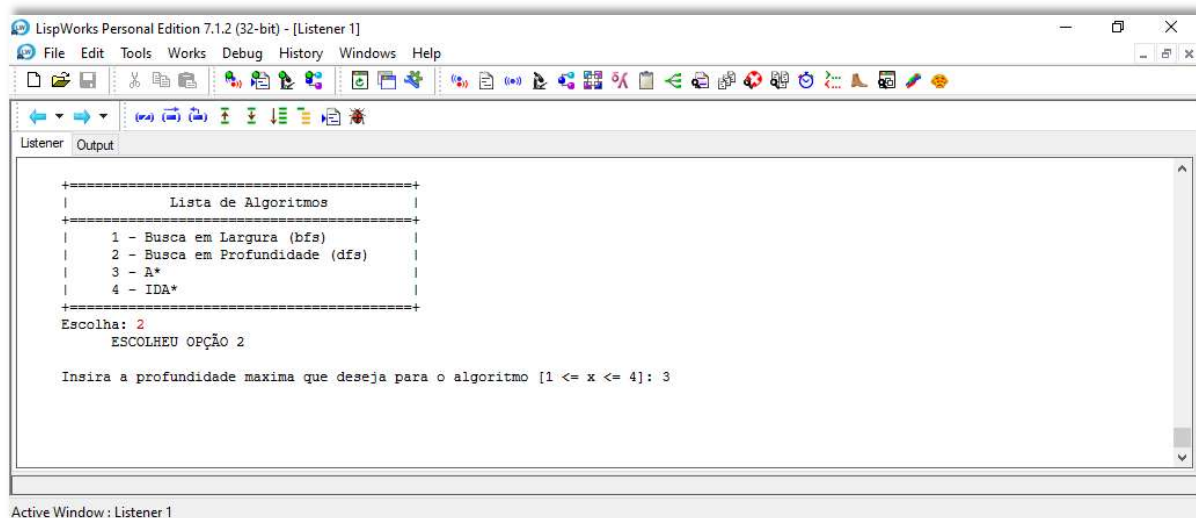


Figura 14 - inserção da profundidade máxima para algoritmo dfs

Quando introduzir um valor na consola, o programa apresentará os resultados do algoritmo **dfs**, como mostra figura 15.

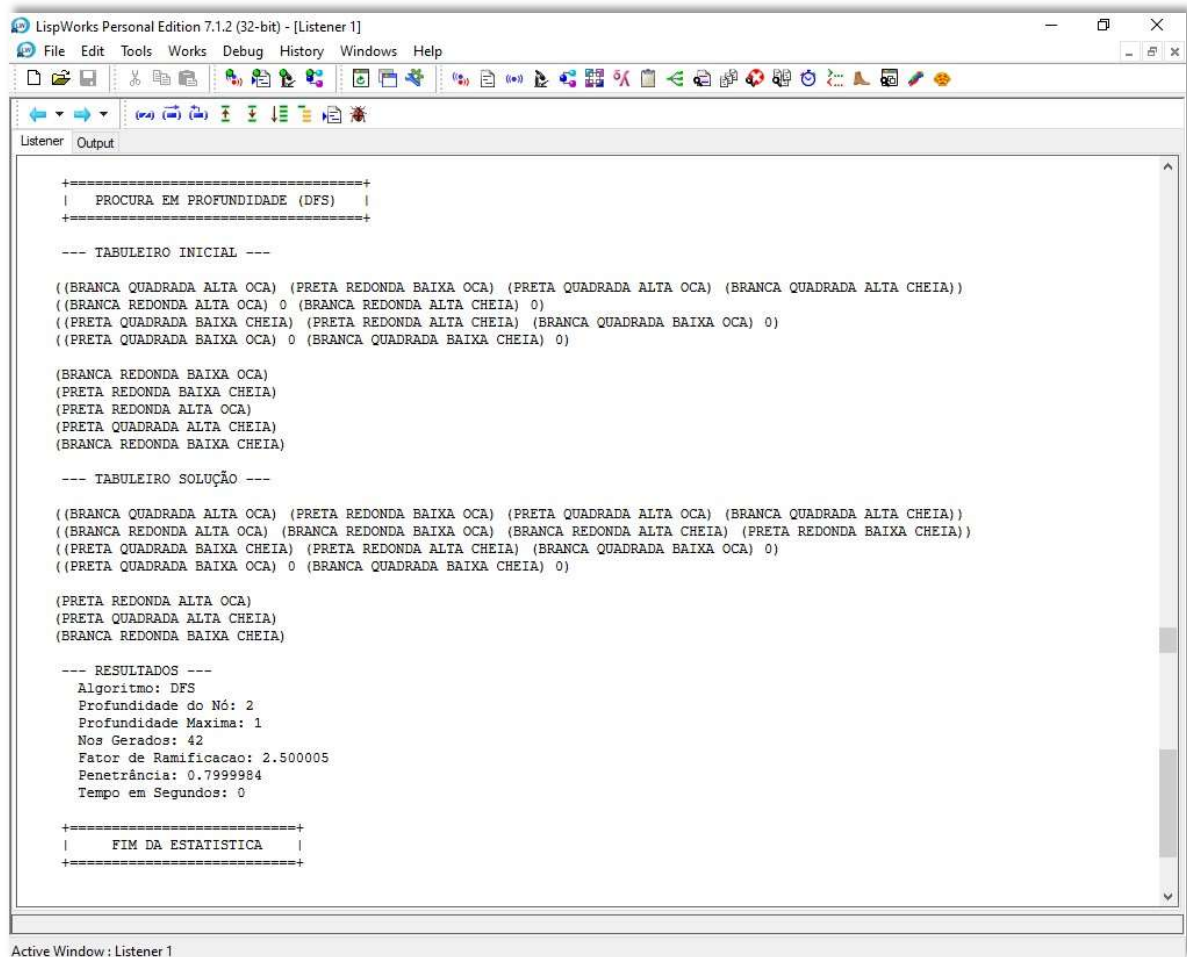


Figura 15 - resultado do problema resolvido com algoritmo dfs na profundidade inserida pelo utilizador

Caso escolha o algoritmo **A*** ou **IDA***, o programa irá pedir que escolha que heurística deseja aplicar no algoritmo, como mostra a figura 16.

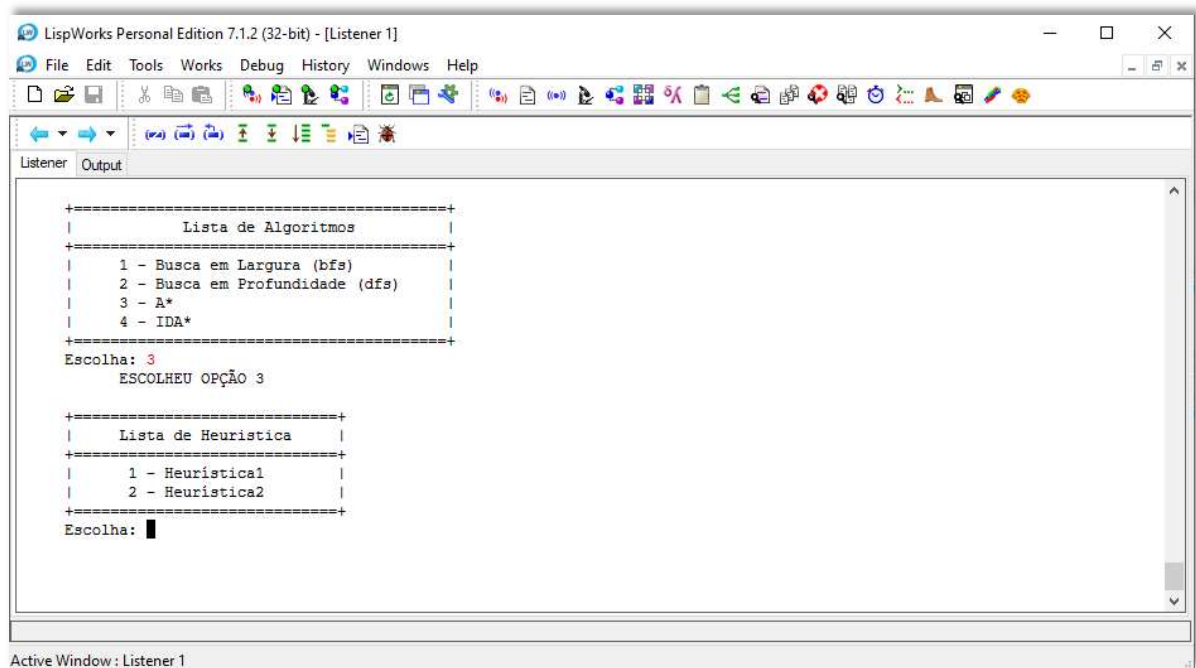


Figura 16 - escolha das heurísticas para algoritmos A* e IDA*

Heurística1

Uma heurística que determina o numero de peças já alinhadas no tabuleiro $h(x) = 4 - p(x)$ em que $p(x)$ é o valor máximo de alinhamento de peças tomando em conta todas as possibilidades em termos de direção e características das peças, isto é, o numero máximo de peças com características comuns já alinhadas na horizontal, na diagonal e vertical.

Heurística2

Foi implementada uma segunda heurística pelos alunos que deverá melhorar o desempenho dos algoritmos de procura informados em relação à primeira fornecida. $h(x) = (4 - p(x)) + q(x)$ sendo $p(x)$ o numero de peças máximas já alinhadas com a mesma característica e $q(x)$ é o número de peças na reserva.

Depois de escolher o tipo de heurística, irá visualizar os resultados da procura, como mostra a figura 17.

```

+-----+
|  PROCURA A*  |
+-----+

--- TABULEIRO INICIAL ---

((BRANCA QUADRADA BAIXA CHEIA) (BRANCA REDONDA ALTA CHEIA) (PRETA REDONDA ALTA CHEIA) (PRETA QUADRADA BAIXA OCA))
(0 0 0 0)
(0 0 0 0)
(0 0 0 0)

(BRANCA QUADRADA ALTA CHEIA)
(BRANCA QUADRADA BAIXA OCA)
(PRETA QUADRADA ALTA CHEIA)
(PRETA QUADRADA ALTA OCA)
(PRETA QUADRADA BAIXA CHEIA)
(BRANCA REDONDA BAIXA OCA)
(BRANCA REDONDA ALTA OCA)
(BRANCA REDONDA BAIXA CHEIA)
(PRETA REDONDA ALTA OCA)
(PRETA REDONDA BAIXA CHEIA)

--- TABULEIRO SOLUÇÃO ---

((BRANCA QUADRADA BAIXA CHEIA) (BRANCA REDONDA ALTA CHEIA) (PRETA REDONDA ALTA CHEIA) (PRETA QUADRADA BAIXA OCA))
(0 0 0 0)
(0 0 0 0)
((BRANCA REDONDA ALTA OCA) (BRANCA REDONDA BAIXA CHEIA) (PRETA REDONDA ALTA OCA) (PRETA REDONDA BAIXA CHEIA))

(BRANCA QUADRADA ALTA CHEIA)
(BRANCA QUADRADA BAIXA OCA)
(PRETA QUADRADA ALTA CHEIA)
(PRETA QUADRADA ALTA OCA)
(PRETA QUADRADA BAIXA CHEIA)
(BRANCA REDONDA BAIXA OCA)

--- RESULTADOS ---
Algoritmo: A*
Profundidade do Nó: 4
Heurística: 2
Custo Heurística: 6
Nos Gerados: 68
Fator de Ramificação: 1.000005
Penetrância: 3.99998
Tempo em Segundos: 2

+-----+
|  FIM DA ESTATISTICA  |
+-----+

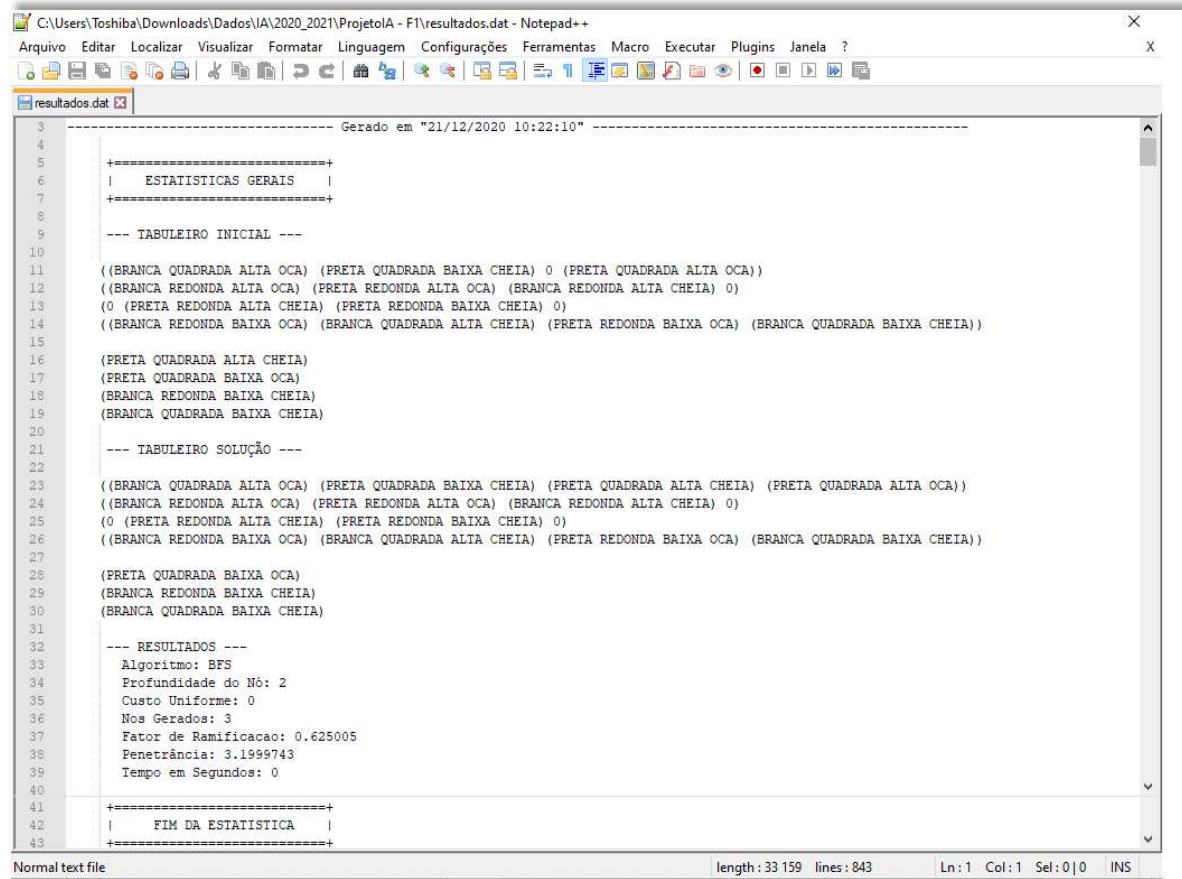
```

Active Window: Listener 1

Figura 17 - resultado do problema resolvido com algoritmo A* e heuristica2

5. Gravação de resultados no ficheiro

Depois de obter os resultados da procura os dados das estatísticas serão gravados num ficheiro na diretoria principal do projeto denominado **resultados.dat** como mostra a figura 18.



```
3----- Gerado em "21/12/2020 10:22:10" -----
4
5+=====+
6|  ESTATISTICAS GERAIS  |
7+=====+
8
9--- TABULEIRO INICIAL ---
10
11((BRANCA QUADRADA ALTA OCA) (PRETA QUADRADA BAIXA CHEIA) 0 (PRETA QUADRADA ALTA OCA))
12((BRANCA REDONDA ALTA OCA) (PRETA REDONDA ALTA OCA) (BRANCA REDONDA ALTA CHEIA) 0)
13(0 (PRETA REDONDA ALTA CHEIA) (PRETA REDONDA BAIXA CHEIA) 0)
14((BRANCA REDONDA BAIXA OCA) (BRANCA QUADRADA ALTA CHEIA) (PRETA REDONDA BAIXA OCA) (BRANCA QUADRADA BAIXA CHEIA))
15
16(PRETA QUADRADA ALTA CHEIA)
17(PRETA QUADRADA BAIXA OCA)
18(BRANCA REDONDA BAIXA CHEIA)
19(BRANCA QUADRADA BAIXA CHEIA)
20
21--- TABULEIRO SOLUÇÃO ---
22
23((BRANCA QUADRADA ALTA OCA) (PRETA QUADRADA BAIXA CHEIA) (PRETA QUADRADA ALTA CHEIA) (PRETA QUADRADA ALTA OCA))
24((BRANCA REDONDA ALTA OCA) (PRETA REDONDA ALTA OCA) (BRANCA REDONDA ALTA CHEIA) 0)
25(0 (PRETA REDONDA ALTA CHEIA) (PRETA REDONDA BAIXA CHEIA) 0)
26((BRANCA REDONDA BAIXA OCA) (BRANCA QUADRADA ALTA CHEIA) (PRETA REDONDA BAIXA OCA) (BRANCA QUADRADA BAIXA CHEIA))
27
28(PRETA QUADRADA BAIXA OCA)
29(BRANCA REDONDA BAIXA CHEIA)
30(BRANCA QUADRADA BAIXA CHEIA)
31
32--- RESULTADOS ---
33Algoritmo: BFS
34Profundidade do Nó: 2
35Custo Uniforme: 0
36Nos Gerados: 3
37Fator de Ramificacao: 0.625005
38Penetrância: 3.1999743
39Tempo em Segundos: 0
40
41+=====+
42|  FIM DA ESTATISTICA  |
43+=====+
```

Normal text file length: 33159 lines: 843 Ln:1 Col:1 Sel:0|0 INS

Figura 18 - resultados gravados no resultados.dat

6. Limitações do programa

Devido à inexistência de um compilador de LISP incorporado na maioria dos Sistemas Operativos, é necessário recorrer a software de terceiros para conseguir compilar código LISP. Neste projeto foi utilizado o software LispWorks para este objetivo, e dado ser a versão gratuita deste software com muitas limitações de memória Stack e Heap o que compromete a execução de alguns algoritmos para alguns problemas. A heurística desenvolvida, para tabuleiros vazios não consegue encontrar solução devido a não descartar nós suficientes para combater a explosão combinatória causada por estes tipos de tabuleiro, levando a um erro de StackOverflow quando utilizada no algoritmo A* e um erro de lista demasiado grande ao tentar aplicar a função mínima a essa lista.