

Mestrado Integrado em Engenharia Informática e Computação

Monkey Queen

Relatório Intercalar

Programação em Lógica

Grupo Monkey\_Queen\_2:

Ana Rita da Costa Torres, up201406093@fe.up.pt

Rui Pedro Correia Soares, up201404965@fe.up.pt

13 de novembro de 2016

**Resumo**

No âmbito da unidade curricular de Programação em Lógica foi proposto desenvolver um jogo, tendo por base a linguagem PROLOG. O jogo escolhido pelo grupo foi o “Monkey Queen”, jogo da família das Damas.

A realização do projeto colocou um desafio que é trabalhar com uma linguagem funcional, visto que apresenta diferenças significativas relativamente a outras linguagens mais, comumente, utilizadas. A solução para a implementação deste jogo passou pela recorrência a predicados já existentes e outros criados pelo grupo, sendo a ajuda dos professores nas aulas práticas essenciais para esta.

Em suma, os conhecimentos relativos à linguagem em estudo aumentaram consideravelmente e o produto final é amostra disso. Acreditamos ainda, que a execução do programa na linha de comandos é simples e de interpretação fácil para o utilizador comum, permitindo que o jogo seja uma experiência agradável.

## **1.Introdução**

## **2. Jogo**

### **2.1. Descrição do Jogo**

O jogo “Monkey Queen” foi concebido por Mark Steere em janeiro de 2011.

Este encontra-se disponível online nos seguintes sites:

* Ig Game Center ([http://www.iggamecenter.com/info/pt/main.html)](http://www.iggamecenter.com/info/pt/main.html)
* Mind Sports ([http://www.mindsports.nl)](http://www.mindsports.nl/)

#### **2.1.1 Regras do Jogo**

O tabuleiro do jogo tem dimensões 12X12, cada jogador tem vinte peças em pilha, numa fase inicial, sendo um jogador da equipa *cigar* (peças castanhas)e outro da equipa *ivory* (peças brancas).

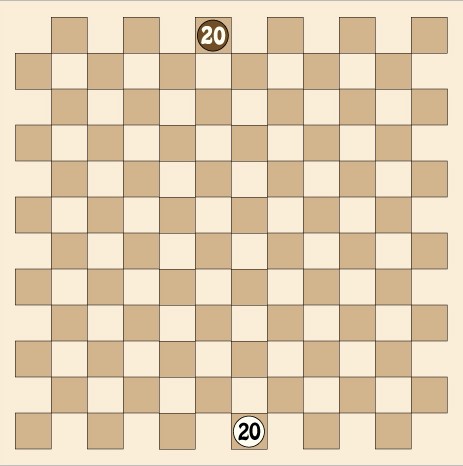


Figura - Tabuleiro no Estado Inicial

O jogador da equipa *ivory* é sempre o primeiro a iniciar o jogo.

Existem dois tipos de peças:

* *Monkey Queen*: pilha de duas ou mais peças (só existe uma de cada jogador no tabuleiro);
* *Baby Monkey*: uma peça (*singleton*).

As peças movem-se em sequências de quadrados não ocupados em linha reta terminada por um quadrado ocupado pelo inimigo, podendo a orientação do movimento ser vertical, horizontal ou diagonal.

No caso da *Monkey Queen*, o movimento com captura pode ter dois desfechos, se o inimigo capturado for um *Baby Monkey*, esta ocupa o seu lugar, por outro lado se for a *Monkey Queen* adversária a equipa que realizou a jogada ganha.

Os movimentos sem captura só podem ser realizados pela *Monkey Queen*, se a sua pilha tiver um número de peças superior a dois e é obrigatório que ao efetuar o movimento esta peça deixe um *singleton* no seu quadrado de partida.

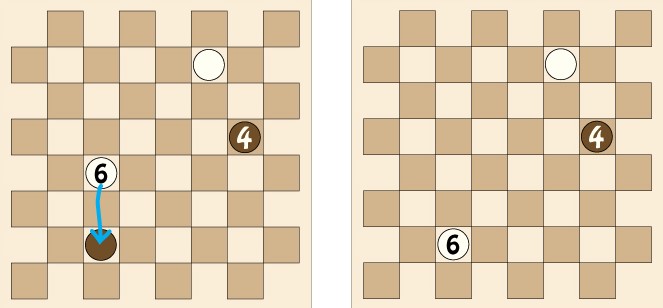


Figura - Ivory Queen captura Baby Cigar

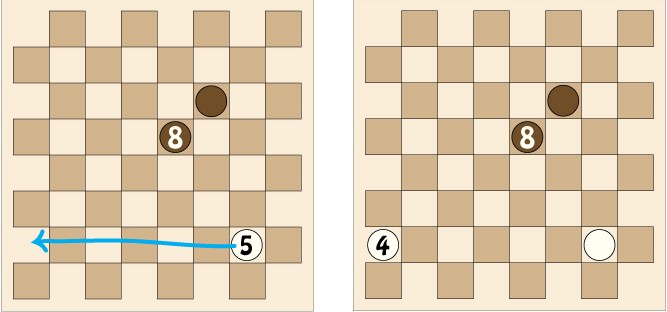


Figura - Ivory Queen executa movimento sem captura

O movimento de captura das peças *Baby Monkey* é semelhante ao movimento das *Queen Monkey*. Já o seu movimento sem captura tem outras condições, ao realizá-lo é imperativo que o *Baby Monkey* se coloque numa posição que se situe a uma menor distância da *Monkey Queen* adversário do que anteriormente.

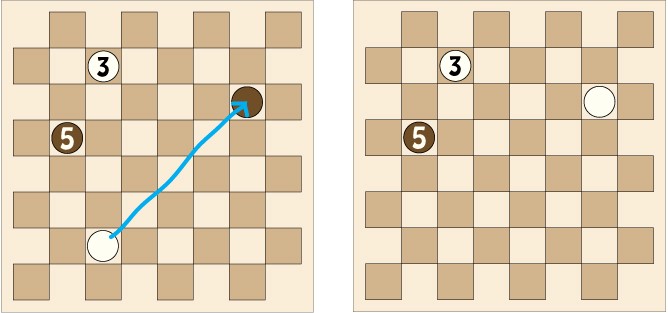


Figura - Baby Ivory captura Baby Cigar

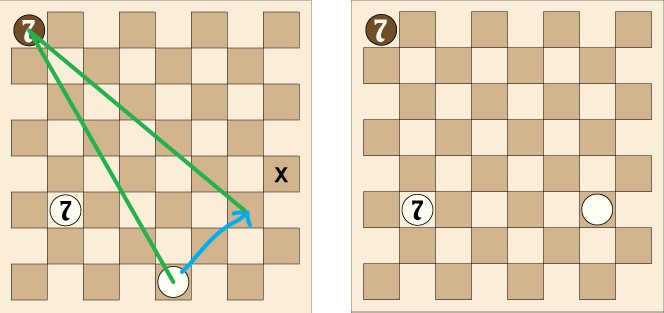


Figura - Baby Ivory executa movimento sem captura

Por fim, existem 2 maneiras de ganhar o jogo, sendo que o empate é um resultado impossível neste jogo:

* Matar a *Monkey Queen* inimiga;
* Posicionar as peças de forma a que o oponente não seja capaz de realizar mais movimentos.

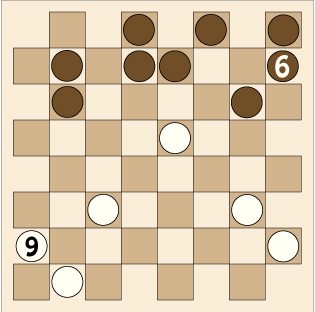


Figura - Cigar Queen não consegue evitar a captura

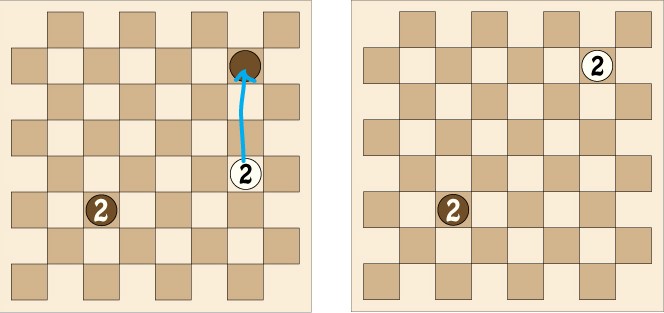


Figura - Baby Ivory só tem uma hipótese de movimento, que consequentemente conduz à sua captura

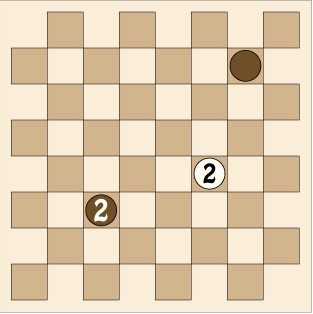


Figura - Ivory Queen não pode realizar qualquer movimento e, portanto, perde o jogo

### **2.2. Implementação do Jogo**

### **2.2.1. Representação do Tabuleiro**

Para a implementação do jogo em Prolog, era necessário escolher que representação de dados usaríamos para o tabuleiro e peças. Como tal, escolhemos uma matriz (ie. lista de listas), de dimensões 12x12, cujos elementos são átomos que representam o estado em que se encontra a célula. De entre os átomos possíveis, temos a célula vazia (*empty*), a Monkey Queen de cada cor (*ivoryQueen, cigarQueen*), e os seus Monkey Babies (*ivoryBaby*, *cigarBaby*).

O tabuleiro começa, tal como referido nas regras, com cada jogador com 20 peças na sua pilha inicial.

O jogador pode usar o predicado *getPiece(+X, +Y, +Board, -Symbol)* para obter a peça correspondente às coordernadas (X, Y) introduzidas. Para além disso pode usar o predicado (ainda não implementado) *setPiece(+X, +Y, +Symbol, +Board, -Board1)*.

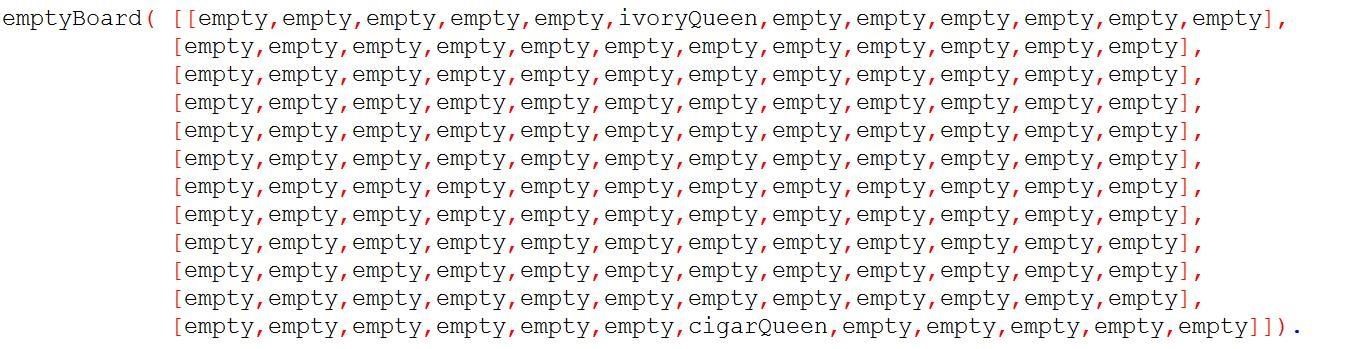


Figura - Tabuleiro Vazio

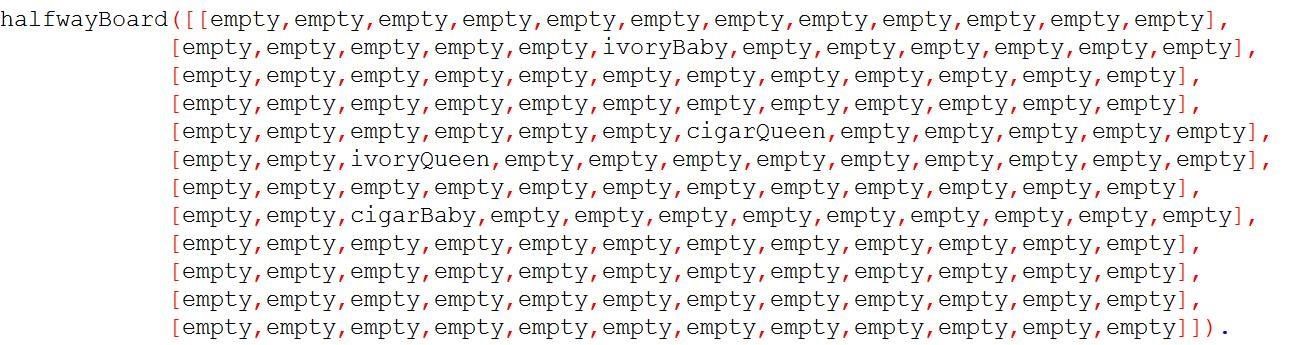


Figura - Tabuleiro Intermédio

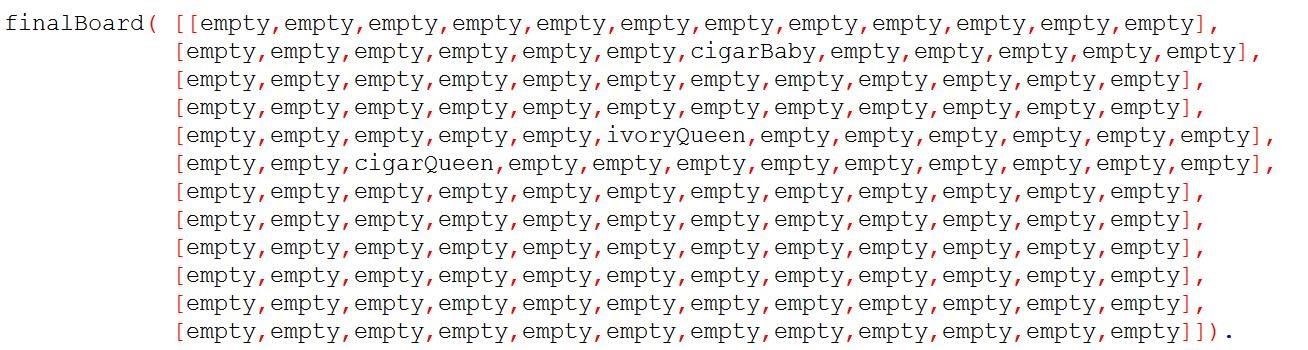


Figura - Tabuleiro Final

### **2.2.2. Visualização do Tabuleiro**

Para imprimir o tabuleiro de jogo para a consola, implementámos um predicado que percorre a representação do tabuleiro e imprime-o, recursivamente e fila a fila. predicado, que para além do estado tabuleiro necessita da quantidade de peças na Queen de cada jogador nesse estado é **printFancyBoard([+IvoryQueenStack, +CigarQueenStack, +Board*])****.*

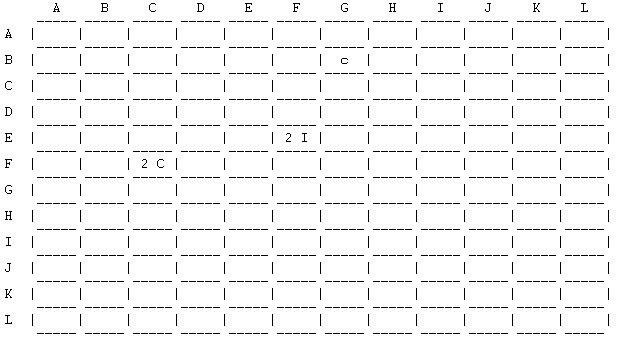


Figura - Tabuleiro Inicial

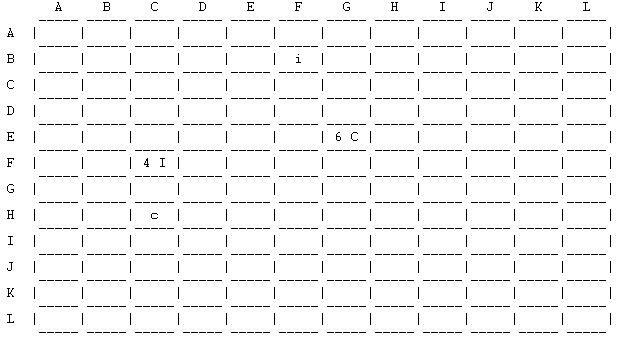


Figura - Tabuleiro Intermédio

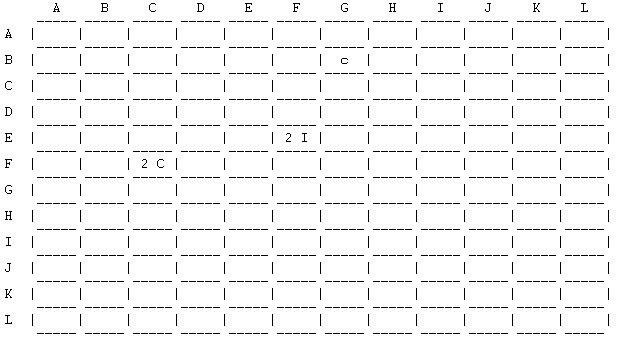


Figura - Tabuleiro Final

### **2.2.3. Movimentos**

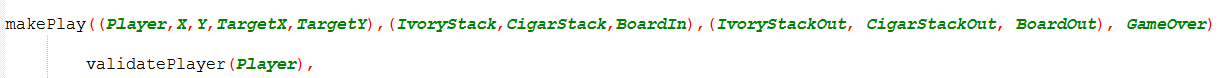


Figura - Predicado makePlay

O predicado acima apresentado verifica se uma jogada é possível e no caso de o ser, executa o movimento. Para além disso, verifica sempre o estado de *Game Over*, isto é, se o jogo terminou. Este recebe o jogador, a sua posição atual, a posição para a qual este se pretende mover, a composição da Queen de cada jogador e o estado de jogo atual. Por fim, retorna a composição atualizada da Queen e o tabuleiro, após a jogada realizada.

## **3.Lógica de Jogo**

**3.1. Representação do estado de jogo**

### **3.2. Visualização do tabuleiro**

O tabuleiro do jogo é representado por uma lista de doze elementos, em que cada elemento é também uma lista de doze elementos, formando um tabuleiro quadrado doze por doze (12x12). As funções relativas à estrutura do tabuleiro e impressão do mesmo, encontram-se no ficheiro *printBoard.pl*. O ficheiro *boards.pl* contém apenas as representações do tabuleiro pedidas na entrega intercalar, isto é, o tabuleiro inicial, intermédio e final.

Para imprimir o tabuleiro basta chamar a função:

**printFancyBoard(+IvoryQueenStack,+CigarQueenStack,+Board)**, em que as variáveis I**voryQueenStack** e **CigarQueenStack** representam o número de peças de cada jogador no estado atual, sendo que, no início o valor será sempre vinte. O **Board** é um tabuleiro de um estado de jogo.

### **3.3. Lista de jogadas válidas**

### **3.4. Execução de jogadas**

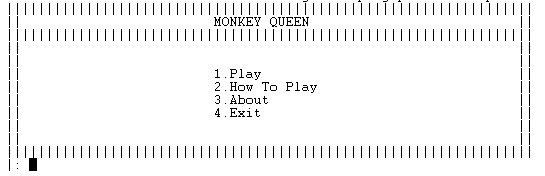
### **3.5. Avaliação do tabuleiro**

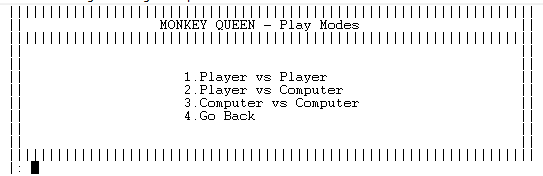
### **3.6. Final do jogo**

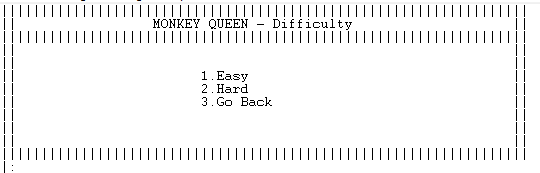
### **3.7. Jogada do computador**

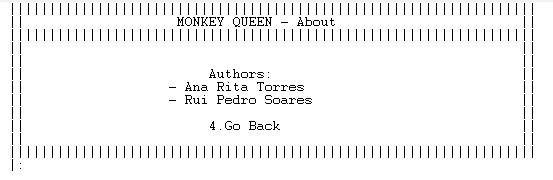
## **4.Interface com o Utilizador**

A interface implementada permite ao utilizador jogar e usufruir do programa de forma simples. O menu inicial tem a seguinte representação:









## **5.Conclusões**