**Jogo de tabuleiro “Mod X”**

Relatório Intercalar



Mestrado Integrado em Engenharia Informática e Computação

Programação em Lógica

Grupo “Mod X” 4:

André Sousa Lago – 201303313

Gustavo Rocha da Silva – 201304143

Faculdade de Engenharia da Universidade do Porto

Rua Roberto Frias, sn, 4200-465 Porto, Portugal

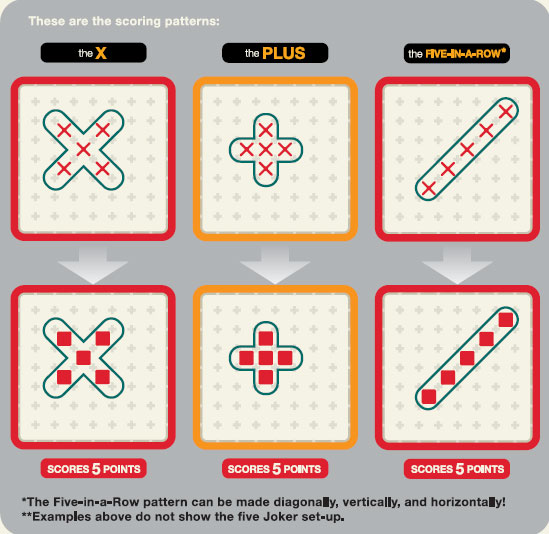
6 de Outubro de 2015

# 1 – O Jogo “*Mod X*”



O “*Mod X*” é um jogo de tabuleiro criado pela “*Cryptozoic Entertainment*” e lançado em Dezembro de 2014, pertencendo à gama de estratégia para 2 a 4 jogadores de idades superiores a 14 anos. Cada jogo possui uma duração aproximada de 25 minutos.

O jogo é jogado num tabuleiro dividido em células (8x8) e tem um funcionamento bastante simples. A cada jogador é atribuída uma cor (vermelho, preto, amarelo e laranja), bem como 14 peças-X e 18 marcadores de pontuação da respetiva cor a cada um. O objetivo do jogo é que cada jogador coloque, à vez, uma peça-X no tabuleiro de forma a criar padrões (X, + ou “5 em linha”). Uma vez criado um padrão, o jogador coloca marcadores de pontuação nas posições onde tinha peças-X, retirando estas para que possam ser reutilizadas. Um jogador que construa um padrão sobre os marcadores já posicionados pode substituir aqueles que as suas peças cobrem por marcadores da sua cor, devolvendo os anteriores ao seu proprietário. Em cada momento, a pontuação de um jogador corresponde ao número de marcadores que possui no tabuleiro, sendo que o jogo termina quando um dos jogadores atinge um valor de pontuação limite pré-estabelecido ou um jogador esgota todas as peças que tem para jogar (neste caso, ganha o jogador com mais pontos).

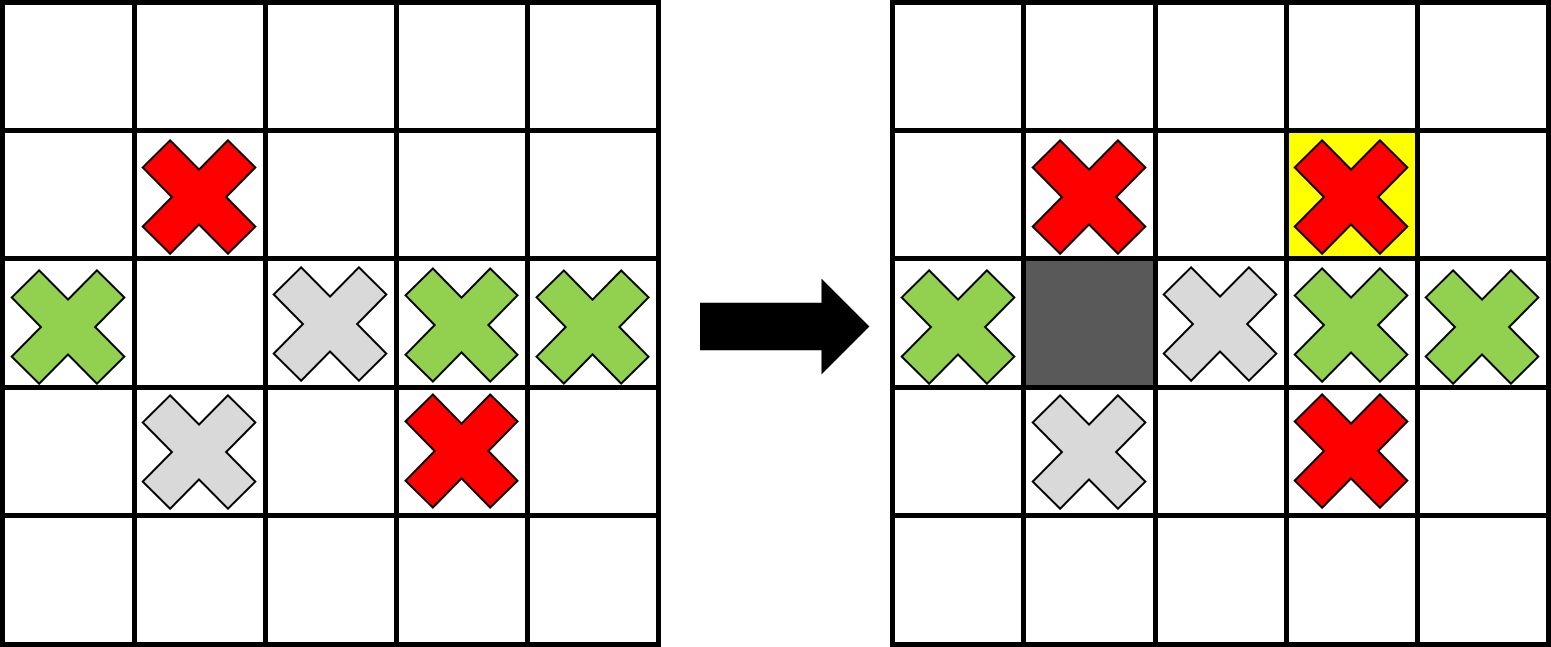


Um padrão tem de ser constituído apenas por peças-X da cor do jogador e por peças-X transparentes. No início do jogo 5 destas peças transparentes são colocadas de forma aleatória no tabuleiro para que possam ser utilizadas por qualquer jogador. Se um jogador conseguir fazer um padrão utilizando uma ou mais peças transparentes, deve substituí-las por marcadores de pontuação da sua cor e reposicioná-las no tabuleiro desde que a nova posição não dê imediatamente origem a um novo padrão.

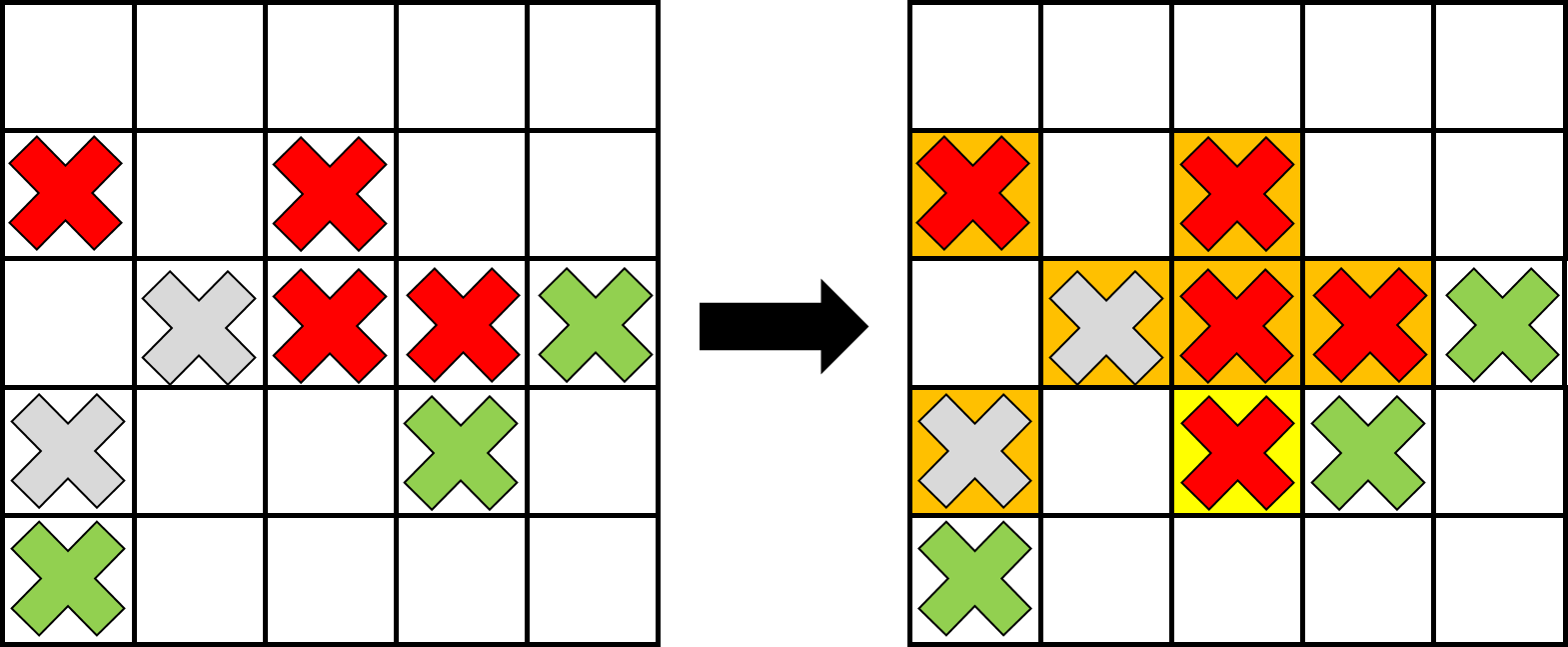
Fig. 1 – Padrões possíveis a realizar. O padrão “5 em linha” é também possível na horizontal ou vertical.

Seguem-se alguns exemplos de situações particulares do jogo:

Na sequência à esquerda, o jogador de vermelho pontua 5 pontos por colocar uma peça na posição assinalada a amarelo. Assim, se fizer isto, retira as três peças vermelhas bem como as duas transparentes uma vez que as usou no seu padrão, colocando de seguida os seus marcadores de posição. Por fim, deve reposicionar as peças transparentes em qualquer posição à exceção da que está assinalada a cinzento pois isso causaria o surgimento de um padrão “5 em linha verde”.



Na imagem à direita pode observar-se outra situação possível no jogo: a de se fazerem dois padrões em simultâneo. Do quadro da esquerda para o da direita, se o jogador de vermelho colocar a sua peça na posição assinalada a amarelo consegue obter dois padrões: uma cruz (X) e um “mais” (+) com recurso a 4 outras peças vermelhas e a 2 peças transparentes. Deste modo, com uma só jogada o jogador obtém 7 pontos.



# 2 – Representação do Estado do Jogo

No caso da nossa implementação do jogo o número de peças vai ser um pouco diferente uma vez que apenas será possível jogar com 2 jogadores (humano-humano, humano-computador, computador-computador).

A representação do estado do jogo vai ser feita com recurso a uma lista de linhas do tabuleiro (em que a primeira corresponde à linha de cima). Por sua vez, cada linha é constituída por uma lista de células do tabuleiro (em que a primeira corresponde à célula mais à esquerda). Por fim, cada célula é constituída por uma lista de dois elementos. O primeiro é o número indicativo da base dessa célula enquanto o segundo é o número indicativo da peça-X presente nessa célula (a correspondência entre códigos e peças está representada na secção “Visualização do Tabuleiro”).

Os únicos requisitos impostos por esta implementação são a necessidade de todas as listas que representam linhas terem o mesmo tamanho, que deverá também corresponder ao número total de linhas (elementos da lista principal). Para além disso, os valores de ocupação de cada célula devem corresponder aos valores enunciados para que a representação seja a mais correta.

# 3 – Visualização do Tabuleiro

O tabuleiro será visualizado através da utilização de caracteres ASCII para representar as peças, posições vazias e limites das células. Segue-se a correspondência entre códigos e valores para as peças-X e marcadores de pontuação:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Peças-X** | | **Marcador de Pontuação** | |
|  | **Código** | **Caractere** | **Código** | **Caractere** |
| **Sem peça** | -1 | ‘ ‘ | -1 | ‘ ‘ |
| **Transparente** | 0 | ‘X’ | Inexistente | |
| **Vermelho** | 1 | ‘R’ | 1 | ‘r’ |
| **Verde** | 2 | ‘G’ | 2 | ‘g’ |

Para além dos valores indicados na tabela, existem também caracteres usados na separação horizontal (‘|’) e vertical (‘-‘) de células.

Como exemplo da representação referida na tabela acima, segue-se uma comparação entre o estado do tabuleiro de jogo e a sua representação em texto na consola de Prolog.

-----------------------------------------------------------------

| | | | | X X | | | |

| | | | | X | | | |

| | | | | X X | | | |

-----------------------------------------------------------------

| | rrrrr | | | | | | |

| | rrrrr | | | | | | |

| | rrrrr | | | | | | |

-----------------------------------------------------------------

| | | rrrrr | | | | X X | |

| | | rrrrr | | | | X | |

| | | rrrrr | | | | X X | |

-----------------------------------------------------------------

| | | | rrrrr | | | | |

| | | | rrrrr | | | | |

| | | | rrrrr | | | | |

-----------------------------------------------------------------

| | | | | rrrrr | | | X X |

| | | | | rrrrr | | | X |

| | | | | rrrrr | | | X X |

-----------------------------------------------------------------

| | | ggggg | ggggg | ggggg | ggggg | GgggG | |

| | | ggggg | ggggg | ggggg | ggggg | ggGgg | |

| | | ggggg | ggggg | ggggg | ggggg | GgggG | |

-----------------------------------------------------------------

| G G | X X | R R | R R | X X | G G | R R | |

| G | X | R | R | X | G | R | |

| G G | X X | R R | R R | X X | G G | R R | |

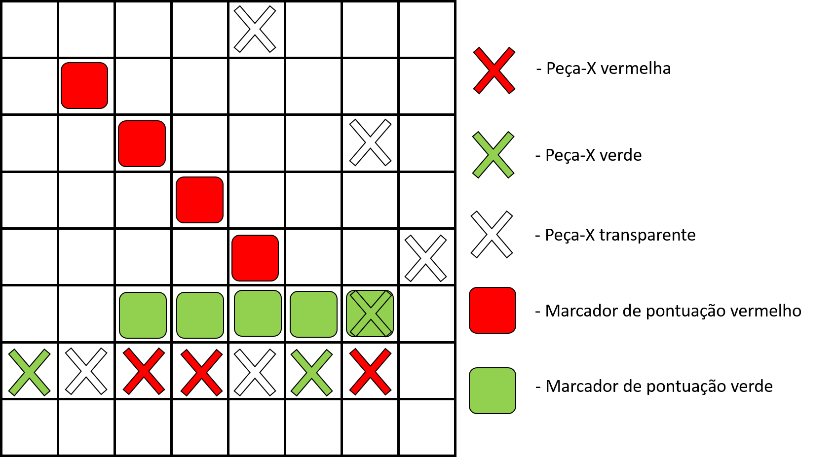
-----------------------------------------------------------------

| | | | | | | | |

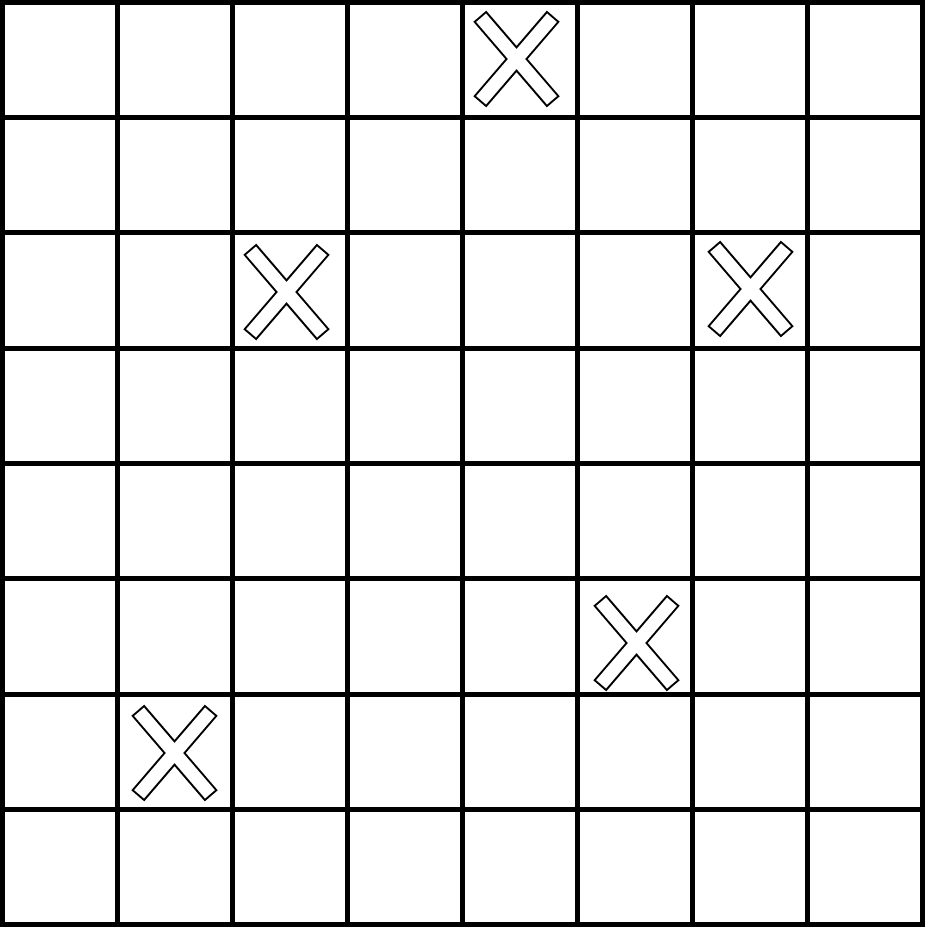
| | | | | | | | |

| | | | | | | | |

-----------------------------------------------------------------



Como se pode ver, as peças-X são representadas colocando o caracter correspondente numa formação em “X”, enquanto os marcadores de posição são representados por quadrados 5x5 preenchidos com o caractere que lhes corresponde. Na coluna 7 da linha 6 é visível o resultado de se colocar uma peça-X sobre um marcador de pontuação.

O estado anterior corresponde a um estado intermédio do jogo, ou seja, a um estado em que o jogo ainda está a decorrer uma vez que todos os jogadores têm peças para jogar e nenhum atingiu a pontuação limite. Assim, segue-se a representação de um estado inicial em que apenas estão colocadas as peças-X transparentes (de notar que a posição de cada uma destas peças é aleatória) e um estado final em que o jogador verde atingiu a pontuação limite (8 pontos). Ambas as representações representam o mesmo jogo:

**Estado inicial do jogo:**

-----------------------------------------------------------------

| | | | | X X | | | |

| | | | | X | | | |

| | | | | X X | | | |

-----------------------------------------------------------------

| | | | | | | | |

| | | | | | | | |

| | | | | | | | |

-----------------------------------------------------------------

| | | X X | | | | X X | |

| | | X | | | | X | |

| | | X X | | | | X X | |

-----------------------------------------------------------------

| | | | | | | | |

| | | | | | | | |

| | | | | | | | |

-----------------------------------------------------------------

| | | | | | | | |

| | | | | | | | |

| | | | | | | | |

-----------------------------------------------------------------

| | | | | | X X | | |

| | | | | | X | | |

| | | | | | X X | | |

-----------------------------------------------------------------

| | X X | | | | | | |

| | X | | | | | | |

| | X X | | | | | | |

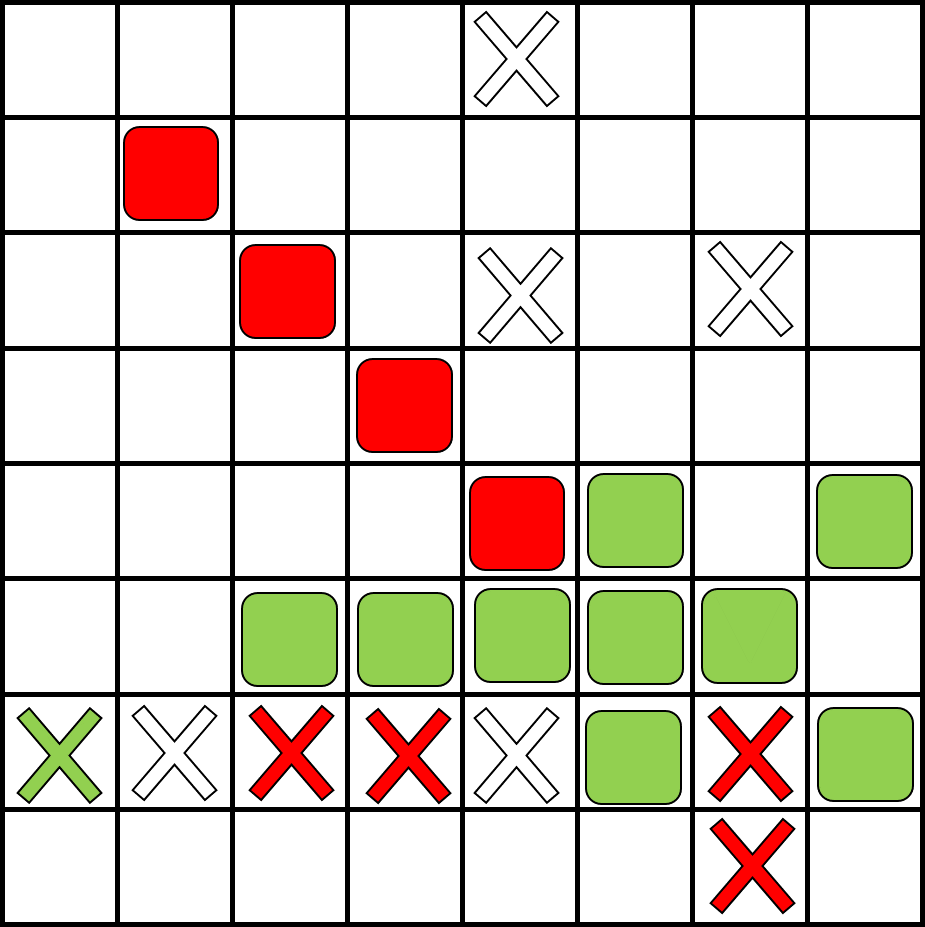
-----------------------------------------------------------------

| | | | | | | | |

| | | | | | | | |

| | | | | | | | |

-----------------------------------------------------------------

Em Prolog, a representação das situações referidas até aqui é feita, como já foi referido, com base em listas e predicados especiais usados para mostrar um tabuleiro. Por exemplo, o tabuleiro do estado final do jogo (imagem acima) seria:

[[[*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *0*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*]],

[[*-1*, *-1*], [*1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*]],

[[*-1*, *-1*], [*-1*, *-1*], [*1*, *-1*], [*-1*, *-1*], [*-1*, *0*], [*-1*, *-1*], [*-1*, *0*], [*-1*, *-1*]],

[[*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*]],

[[*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*1*, *-1*], [*2*, *-1*], [*-1*, *-1*], [*2*, *-1*]],

[[*-1*, *-1*], [*-1*, *-1*], [*2*, *-1*], [*2*, *-1*], [*2*, *-1*], [*2*, *-1*], [*2*, *-1*], [*-1*, *-1*]],

[[*-1*, *2*], [*-1*, *0*], [*-1*, *1*], [*-1*, *1*], [*-1*, *0*], [*2*, *-1*], [*-1*, *1*], [*2*, *-1*]],

[[*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *-1*], [*-1*, *1*], [*-1*, *-1*]]]

**Estado final do jogo:**

-----------------------------------------------------------------

| | | | | X X | | | |

| | | | | X | | | |

| | | | | X X | | | |

-----------------------------------------------------------------

| | rrrrr | | | | | | |

| | rrrrr | | | | | | |

| | rrrrr | | | | | | |

-----------------------------------------------------------------

| | | rrrrr | | X X | | X X | |

| | | rrrrr | | X | | X | |

| | | rrrrr | | X X | | X X | |

-----------------------------------------------------------------

| | | | rrrrr | | | | |

| | | | rrrrr | | | | |

| | | | rrrrr | | | | |

-----------------------------------------------------------------

| | | | | rrrrr | ggggg | | ggggg |

| | | | | rrrrr | ggggg | | ggggg |

| | | | | rrrrr | ggggg | | ggggg |

-----------------------------------------------------------------

| | | ggggg | ggggg | ggggg | ggggg | ggggg | |

| | | ggggg | ggggg | ggggg | ggggg | ggggg | |

| | | ggggg | ggggg | ggggg | ggggg | ggggg | |

-----------------------------------------------------------------

| G G | X X | R R | R R | X X | ggggg | R R | ggggg |

| G | X | R | R | X | ggggg | R | ggggg |

| G G | X X | R R | R R | X X | ggggg | R R | ggggg |

-----------------------------------------------------------------

| | | | | | | R R | |

| | | | | | | R | |

| | | | | | | R R | |

-----------------------------------------------------------------

O predicado utilizado para visualizar um tabuleiro é o “print\_board(Board)”, no qual a variável “Board” representa uma lista semelhante à que se encontra acima, podendo diferir os valores numéricos no seu interior. O predicado e respetivos predicados auxiliares são:

print\_board\_line\_aux([[**B**, **P**] | **T**], **Line**) **:-**

write(' '),

print\_cell(B, P, Line),

write(' |'),

print\_board\_line\_aux(T, Line)**.**

print\_board\_line\_aux([], \_)**.**

print\_dashed\_line([\_ | **T**]) **:-**

write('--------'),

print\_dashed\_line(T)**.**

print\_dashed\_line([]) **:-** write('-')**.**

xpiece\_to\_ascii(*-1*, ' ')**.**

xpiece\_to\_ascii(*0*, 'X')**.**

xpiece\_to\_ascii(*1*, 'R')**.**

xpiece\_to\_ascii(*2*, 'G')**.**

print\_board([**H** | **T**]) **:-**

print\_dashed\_line(H), nl,

print\_board\_content([H | T])**.**

print\_board\_content([**H** | **T**]) **:-**

print\_board\_row(H), nl,

print\_dashed\_line(H), nl,

print\_board\_content(T)**.**

print\_board\_content([])**.**

print\_board\_row([**H** | **T**]) **:-**

write('|'), print\_board\_line\_aux([H | T], *1*),

nl,

write('|'), print\_board\_line\_aux([H | T], *2*),

nl,

write('|'), print\_board\_line\_aux([H | T], *3*)**.**

print\_cell(**Marker**, **Xpiece**, *2*) **:-**

Xpiece > *-1*,

xpiece\_to\_ascii(Xpiece, **X**),

marker\_to\_ascii(Marker, **M**),

print(M),

print(M),

print(X),

print(M),

print(M)**.**

print\_cell(**Marker**, **Xpiece**, *3*) **:-**

Xpiece > *-1*,

print\_cell(Marker, Xpiece, *1*)**.**

marker\_to\_ascii(*-1*, ' ')**.**

marker\_to\_ascii(*1*, 'r')**.**

marker\_to\_ascii(*2*, 'g')**.**

print\_cell(**Marker**, *-1*, \_) **:-**

marker\_to\_ascii(Marker, **M**),

print(M),

print(M),

print(M),

print(M),

print(M)**.**

print\_cell(**Marker**, **Xpiece**, *1*) **:-**

Xpiece > *-1*,

xpiece\_to\_ascii(Xpiece, **X**),

marker\_to\_ascii(Marker, **M**),

print(X),

print(M),

print(M),

print(M),

print(X)**.**

Para além destes predicados, implementamos também um predicado “create\_board(Board)” que verifica se a variável “Board” contém um tabuleiro vazio (sem peças). Assim, pode ser usado para atribuir a uma variável esse tabuleiro.

create\_board(**Board**) **:-**

board\_size(**Size**),

create\_board(Size, Board)**.**

create\_board(**Size**, **Board**) **:-** create\_board(Size, Size, Board)**.**

create\_board(**Width**, **Height**, **Board**) **:-**

Height > *0*,

**H1** is Height - *1*,

create\_board\_line(Width, **L1**),

create\_board(Width, H1, **B1**),

append([L1], B1, Board)**.**

create\_board(\_, *0*, [])**.**

create\_board\_line(**Width**, **Line**) **:-**

Width > *0*,

**W1** is Width - *1*,

create\_board\_line(W1, **L1**),

append([[*-1*, *-1*]], L1, Line)**.**

create\_board\_line(*0*, [])**.**

# 4 – Movimentos

Os movimentos possíveis no jogo e respetivos predicados (apenas o protótipo) são:

* Colocar uma peça-X (transparente ou não)
  + Se for formado um padrão o predicado deve substituir todas as peças-X do padrão por marcadores de pontuação e coloca as peças transparentes usadas no padrão em novos sítios de forma aleatória.
  + “coloca\_peca(Board, Peça, X, Y, Result)”
  + No predicado, Board representa o tabuleiro, X e Y a posição nova da peça e Result contém o resultado da jogada (-2 se a posição é inválida, -1 se não foi feito nenhum padrão, maior ou igual a zero indica que foi feito um padrão e qual é o número de peças transparentes – joker – a colocar).
* Co

# 5 – Bibliografia

* Página oficial do Mod X (<https://www.cryptozoic.com/games/mod-x>)