**Sixteen Stone**

**Relatório Intercalar**



Mestrado Integrado em Engenharia Informática e Computação

Programação em Lógica

**Grupo 2: Sixteen Stone**

Pedro Miguel Vieira da Silva – 201306032

Miguel Guilherme Perestrelo Sampaio Pereira – 201305998

11 de outubro de 2015

Índice

[1. Sixteen Stone 3](#_Toc432281645)

[**História** 3](#_Toc432281646)

[**Detalhes do jogo** 3](#_Toc432281647)

[**Objetivo** 3](#_Toc432281648)

[**Regras** 3](#_Toc432281649)

[2. Representação do Estado de Jogo 6](#_Toc432281650)

[3. Visualização do tabuleiro 7](#_Toc432281651)

[4. Movimentos 8](#_Toc432281652)

# Sixteen Stone

## **História**

Sixteen Stone é um jogo de tabuleiro da categoria da estratégia abstrata lançado em 2015 e criado por Gary Boyd.

## **Detalhes do jogo**

O jogo realiza-se num tabuleiro de dimensões 5x5, em que as casas têm cores semelhantes. Cada jogador começa com oito peças vermelhas ou azuis, colocadas alternadamente em qualquer uma das células vazias. O jogo começa assim que todas as peças tenham sido colocadas.

## **Objetivo**

O objetivo do jogo é remover as peças do oponente do tabuleiro, empurrando-as ou capturando-as.

## **Regras**

Começando pelo jogador vermelho, os jogadores jogam à vez fazendo diferentes ações. Os jogadores podem realizar cada uma das seguintes ações uma vez por turno, por qualquer ordem:

* Empurrar
* Movimentar
* Sacrificar

**Empurrar uma peça**

Os jogadores podem empurrar as peças do oponente se tiverem mais peças numa linha do que o seu oponente. Se uma peça for empurrada do tabuleiro, ela é devolvida ao conjunto de peças do oponente.

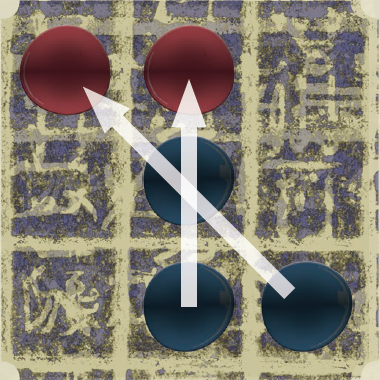
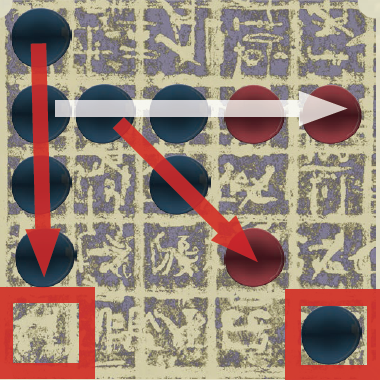


Figura 2 - 3 peças podem empurrar 2 peças. Os jogadores não podem empurrar as suas próprias peças

Figura 1 - As peças podem mover-se na diagonal

**Movimentar uma peça**

Para além de empurrar, os jogadores podem movimentar uma das suas peças para qualquer célula adjacente livre.

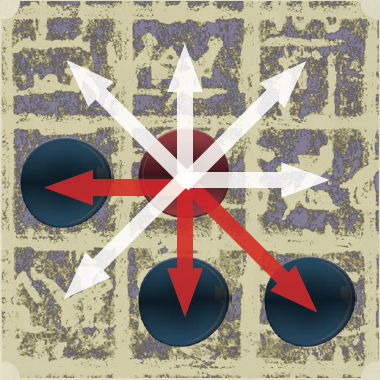


Figura 3 - Uma peça pode ser movida diagonalmente ou ortogonalmente

**Capturar uma peça**

Se um jogador mover uma peça para uma posição que cerque uma peça do oponente em dois lados apostos, essa peça é devolvida ao conjunto de peças do oponente e é substituída por uma peça de outra cor.

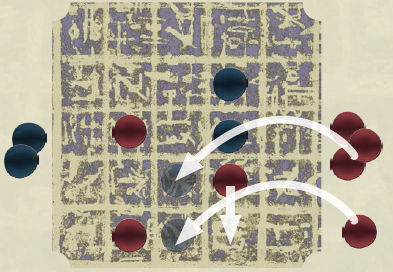


Figura 4 - Esta jogada resultará na captura de duas peças azuis, uma vez que o movimento da peça vermelha causa que ambas fiquem cercadas em lados opostos

Uma peça não é capturada se se mover voluntariamente para uma posição entre duas peças do oponente.

**Sacrificar uma peça**

Os jogadores podem remover permanentemente uma das peças do seu monte para fazer uma jogada adicional.

**Regras adicionais e clarificações**

* Os jogadores podem realizar cada uma das três ações durante a sua vez. É possível um jogador, empurrar, movimentar e sacrificar, de forma a poder empurrar ou movimentar outra vez.
* Um jogador não pode empurrar uma das próprias peças.
* As peças são empurradas apenas uma célula na direção do empurrão. Se saírem do tabuleiro, são colocadas no monte de peças.
* Enquanto se empurra, se uma das peças do jogador for movida para uma posição que não estava previamente ocupada por uma peça e cercar uma peça do oponente, o jogador pode capturar essa peça.
* Se o jogador empurrar uma peça do oponente para uma posição entre duas das peças do jogador, essa peça pode ser capturada.
* Quando uma peça do oponente é capturada, a peça com a qual essa é substituída não pode ser usada imediatamente para cercar outra peça. As peças têm de ser movidas para poderem capturar.
* Se as peças do jogador estiverem todas no tabuleiro, o jogador não pode capturar peças do oponente.

**Final do jogo**

Se um jogador ficar reduzido a uma única peça, o jogo acaba imediatamente e o jogador perde o jogo.

# Representação do Estado de Jogo

Representação do estado inicial do tabuleiro:

[ [empty, empty, empty, empty, empty],

[empty, empty, empty, empty, empty],

[empty, empty, empty, empty, empty],

[empty, empty, empty, empty, empty],

[empty, empty, empty, empty, empty] ]

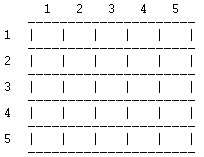


Figura 5 - Estado inicial do tabuleiro visualizado na consola

Representação de um possível estado intermédio:

[ [empty, blue, empty, red, empty],

[blue, blue, empty, empty, empty],

[red, blue, red, empty, empty],

[empty, empty, empty, empty, empty],

[blue, red, red, blue, empty] ]

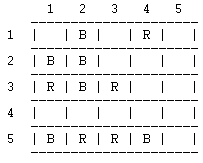


Figura 6 - Estado intermédio do tabuleiro visualizado na consola

Representação de um possível estado final:

[ [empty, blue, empty, red, empty],

[empty, red, empty, empty, empty],

[red, red, red, empty, empty],

[empty, empty, empty, empty, empty],

[empty, red, red, empty, empty] ]

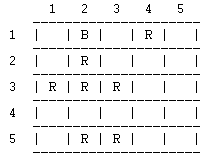
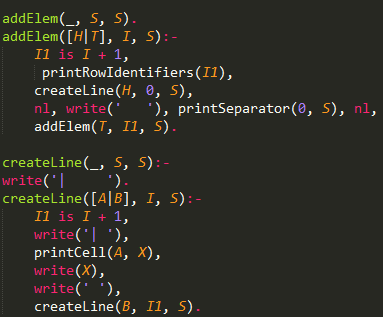


Figura 7 - Estado final do tabuleiro visualizado na consola

# Visualização do tabuleiro

De seguida, apresenta-se o código responsável por mostrar o tabuleiro na consola:



O código acima cria uma lista de listas que representa o tabuleiro de jogo.

# Movimentos

Cabeçalho do predicado para empurrar uma peça:

pushPiece(R, C, dest)

Cabeçalho do predicado para movimentar uma peça:

movePiece(R, C, destR, destC)

Cabeçalho do predicado para capturar uma peça:

capturePiece(R, C, destR, destC)

Cabeçalho do predicado para sacrificar uma peça:

sacrificePiece(R, C)