# Relatório do Projeto Final

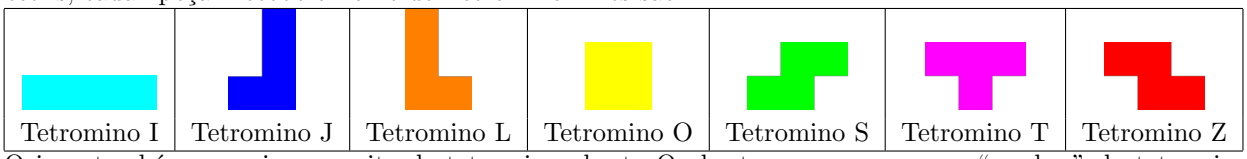
Fabio Luis Arruda Fernandes – 11201721754

## 1 Descrição do Projeto

Esse projeto é uma re-criação do jogo Tetris. O objetivo desse projeto era, a princípio, se manter o mais fiel possível ao jogo original, porém, após uma pesquisa mais a fundo foi percebido que na verdade existem diversas versões do jogo, sendo que recentemente foram lançados alguns padrões para criação de variantes do Tetris. Com isso em mente, o projeto foi feito seguindo a maior quantidade de padrões possível.

### 1.1 Nomenclatura

No tetris, cada "peça" recebe o nome de Tetromino. Eles são:




O jogo também possui o conceito de tetromino ghost. O ghost aparece como um "sombra" do tetromino atual, porém sua posição é onde o tetromino atual cairá se o usuário não apertar nenhum botão.

Outra funcionalidade do jogo é a do hold. A qualquer momento enquanto o tetromino atual cai, o jogador pode efetuar a ação de hold. Ai, se já houver um tetromino na zona de hold, o usuário troca esse tetromino com o atual. Caso contrário o atual vai para a zonda de hold e um novo tetromino é gerado. Quando o usuário faz o hold de um tetromino, ele não poderá fazer mais nenhum hold até que o tetromino atual termine de cair e um novo tetromino seja gerado. A zona de hold sempre começa vazia.

Por fim, o jogo permite que o jogador faça um "hard drop". Essa ação move o tetromino atual para a posição do seu ghost.

### 1.2 Inputs do jogador

Ao abrir o jogo, nada acontecerá até que o jogador precione a barra de espaço uma vez, o que fará com que o jogo comece. A partir desse momento, os possíveis inputs do jogador são:

- Seta para a direita/esquerda/baixo: Move o tetromino um espaço para a direita/esquerda/baixo;
- Teclas Q/R: Rotacionam o tetromino no sentido anti-horário/horário;
- Tecla W: Faz o hold do tetromino atual;
- Barra de espaço: Faz um hard drop no tetromino atual.

Vale dizer que segurar uma tecla não repete o input.

### 1.3 Geração dos tetrominoes que o jogador recebe

Atualmente o tetris não usa um sistema realmente aleatório para gerar a lista dos próximos tetrominoes que o jogador vai receber, ao invés disso, o jogo pega a lista com todos os tetrominoes e embaralha essa lista. Isso garante uma distribuição uniforme dos tetrominoes e impede que o jogador receba repetições consecutivas do mesmo tetromino, tornando uma sequência de 3 ou mais repetições consecutivas do mesmo tetromino impossível de aparecer.

### 1.4 Pontuação

O jogo do tetris trabalha com o conceito de nível, o jogo começa no nível 1 e sobe a cada 10 linhas que o jogador limpa. O nível do jogo determina duas coisas, a velocidade de queda dos tetrominoes e a pontuação que o usuário ganha ao limpar um conjunto de linhas. Além disso, limpar várias linhas em sequência dá uma pontuação adicional.

| Ação | Pontos obtidos |
|---|---|
| 1 linha | 100 ×nível |
| 2 linha | 300 ×nível |
| 3 linha | 500 ×nível |
| 4 linha | 800 ×nível |
| combo | 50 ×quantidade de combo ×nível |

A velocidade de queda segue a seguinte tabela, sendo que 1G = 1 célula por frame.

| Nível atual | Velocidade em G | Nível atual | Velocidade em G |
|---|---|---|---|
| 1 | 0.01667G | 9 | 0.1775G |
| 2 | 0.021017G | 10 | 0.2598G |
| 3 | 0.026977G | 11 | 0.388G |
| 4 | 0.035256G | 12 | 0.59G |
| 5 | 0.04693G | 13 | 0.92G |
| 6 | 0.06361G | 14 | 1.46G |
| 7 | 0.0879G | 15+ | 2.36G |
| 8 | 0.1236G | | |

Note que para descobrirmos quantas frames demorar para uma célula cair os números não são inteiros. Por isso o jogo guarda o tempo desde a ultima vez que a célula caiu e usa essa informação para decidir de quantos em quantos frames o tetromino deve cair uma célula.

### 1.5 *Wall Kicks*

A ultima funcionalidade dessa implementação do Tetris são os wall kicks. Quando o jogador tenta rotacionar um tetromino, caso essa rotação faça ele colidir com algum bloco do tabuleiro ou alguma parede, o jogo tenta transladar esse tetromino em algumas direções, sendo que somente no caso de nenhum desses testes ser válido é que a rotação não acontece. O conjunto dessas translações que são testadas é chamado de um sistema de rotação. Essa versão do tetris implementa o chamado *Super Rotation System*. Para informações sobre ele, clique aqui.

## 2 Como usar

Para usar o projeto, basta fazer um `stack run`, esperar a janela abrir e apertar a barra de espaço quando estiver pronto para o jogo começar.

## 3 Dificuldades, destaques do código

A principal dificuldade do código foi fazer funções impuras. Como estamos tratando de um jogo, diversas ações feitas modificam o estado atual. Sendo assim, muitas das funções implementadas precisavam contornar o fato de que não podemos mudar o valor "guardado" em uma variável. Outra dificuldade foi aprender a usar a biblioteca Gloss, que era essencial para o funcionamento adequado do projeto. No fim, essa dificuldade foi superada com alguns testes e com o uso da documentação.

Dentre os destaques do código está a reusabilidade das funções. As funções de aplicar uma translação, aplicar uma rotação e desenhar o tetromino na tela foram feitas de maneira genérica para que pudessem ser re-utilizadas pelo resto do programa. A função de desenhar o tetromino foi usada para desenhar o tabuleiro, o tetromino atual, o ghost, o tetromino do hold e a lista dos 6 próximos tetrominoes que o jogador receberá. As funções de rotação e translação acabaram ficando bem simples também, já que era só usar a função genérica passando como parâmetro a ação desejada.

Outro destaque está na parte dos vetores. Um dos tipos de dados utilizados no código é um vetor 2d. Para ele, foi criada uma instância de `Num`, fazendo com que possamos somar e subtrair dois vetores sem a necessidade de funções adicionais.

Não consegui encontrar nenhuma oportunidade de implementar Functors e Applicatives, pois a única coleção de dados que definimos foi uma Matriz, porém todas as vezes que chamamos funções que iteravam sobre seus elementos, essas funções tratavam de elementos específicos da matriz, necessitando que parâmetros `i` e `j` fossem passados.

Não foi implementada nenhuma instância de Monad, porém os blocos `do` foram extensivamente usados no código, já que diversas funções precisavam de muitas variáveis auxiliares, ou da aplicação de muitas funções em sequência sobre um valor, portanto esses blocos melhoraram muito a legibilidade do código."

# 4 Video de apresentação

O vídeo de apresentação do projeto pode ser encontrado aqui.