1. Qual a lógica do jogo, como funciona?

A ideia do jogo é misturar dois jogos bastante comuns e simples, de maneira que apareça nessa junção uma complexidade interessante tanto para os jogadores quanto para lógica computacional necessária para implementá-la. Os dois jogos comuns são jogo da velha - que dispensa apresentações - e *dots and boxes*, exemplificado abaixo.

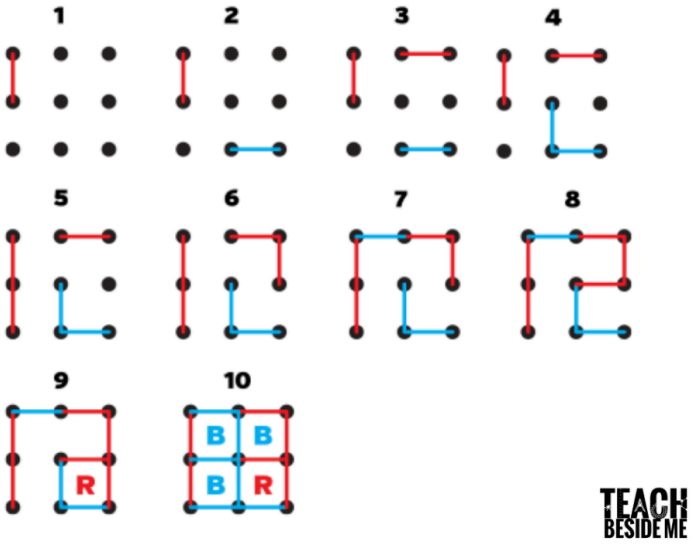

Figura 1 - *Dots and Boxes*

O resultado dessa mistura é um tabuleiro que tem uma grade 3x3 de partidas de *dots and boxes*, onde cada partida tem uma grade de 4x4 pontos, como na imagem abaixo.

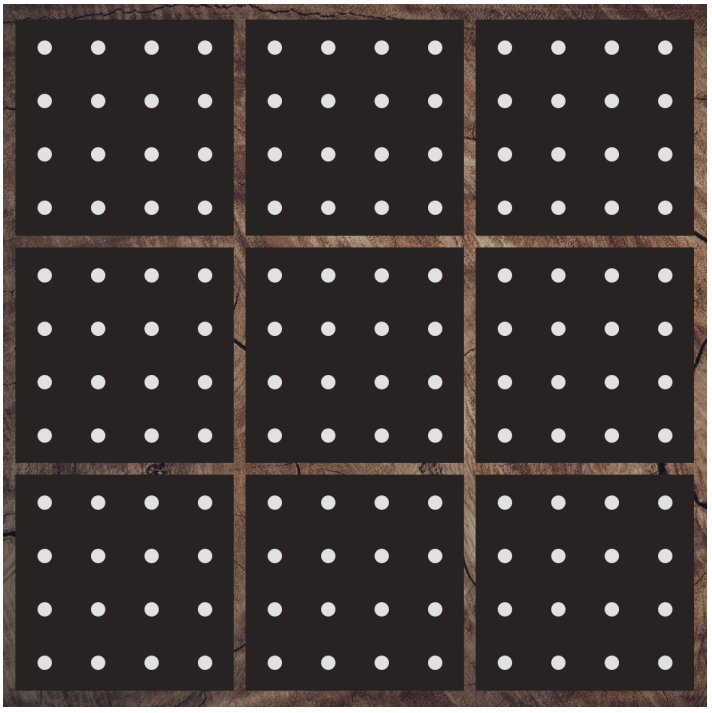

Figura 2 - Mockup de UI do nosso jogo

Para fins de notação, vamos chamar cada matriz pequena de *dots and boxes* de microjogo 1, microjogo 2, …, microjogo 9, na ordem que estão apresentados, enquanto que o macrojogo será o tabuleiro completo de *Ultimate Dots and Boxes*.

A partir daí os jogadores começam a intercalar suas jogadas, jogando apenas no quadrado central da matriz grande (microjogo 5), como se fosse uma partida normal de *dots and boxes*. Cada jogador uma vez, mas pode jogar de novo - no mesmo microjogo - se completar um quadrado. A partir do momento que o primeiro jogador completa um quadrado - depois de sua jogada extra - o outro jogador deve fazer a sua jogada no quadrado que é equivalente ao completado no dentro do microjogo, mas na escala maior.

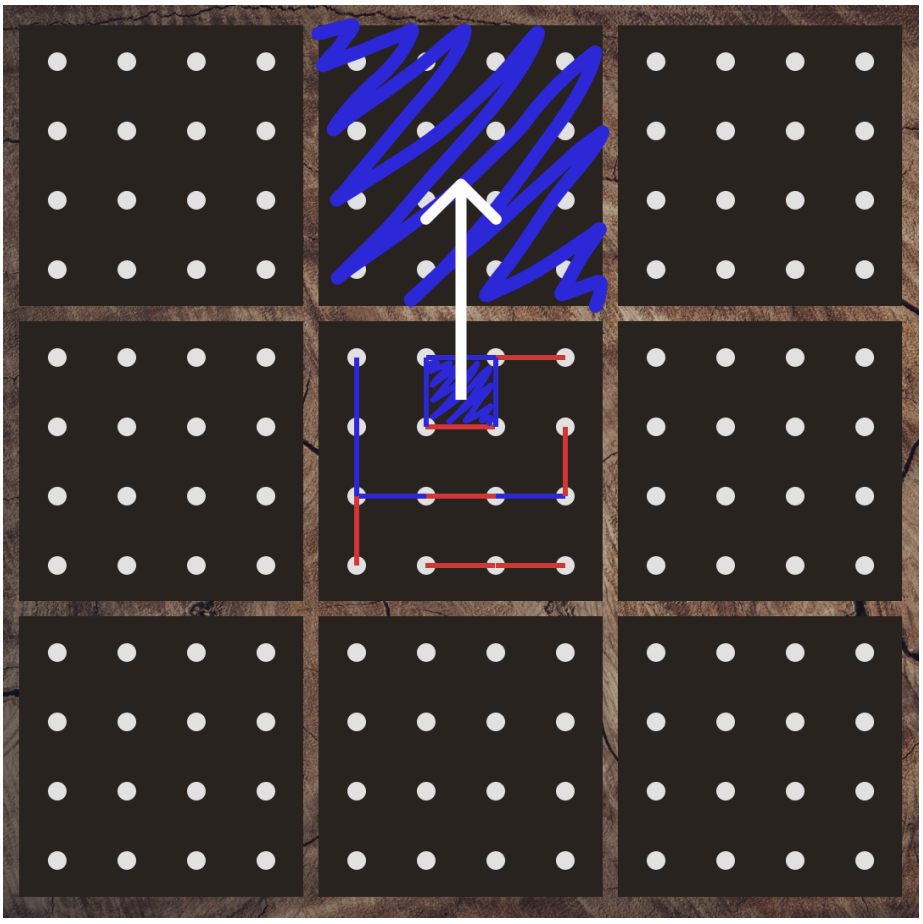

Figura 3 - Quando o quadrado superior central **de um microjogo** é preenchido, o jogador seguinte deve **jogar no microjogo** que ocupa a mesma posição da matriz grande, no caso da imagem, o microjogo 2.

A lógica do jogo se mantém essa ao longo das jogadas, variando em qual microjogo os jogadores se encontram, até que um dos jogadores consiga preencher 5 quadrados de um mesmo microjogo. Nesse momento, o jogador que faz isso ganha aquele microjogo, que fica portanto marcado como ganho por esse jogador.

Para ganhar a totalidade do jogo, o jogador deve ganhar 3 microjogos em linha (vertical, horizontal ou diagonal), **ou** ganhar quaisquer 5 microjogos.

Abaixo, segue uma possível sequência de jogadas em uma partida, onde o jogador azul ganha o jogo após vencer 3 microjogos em linha.

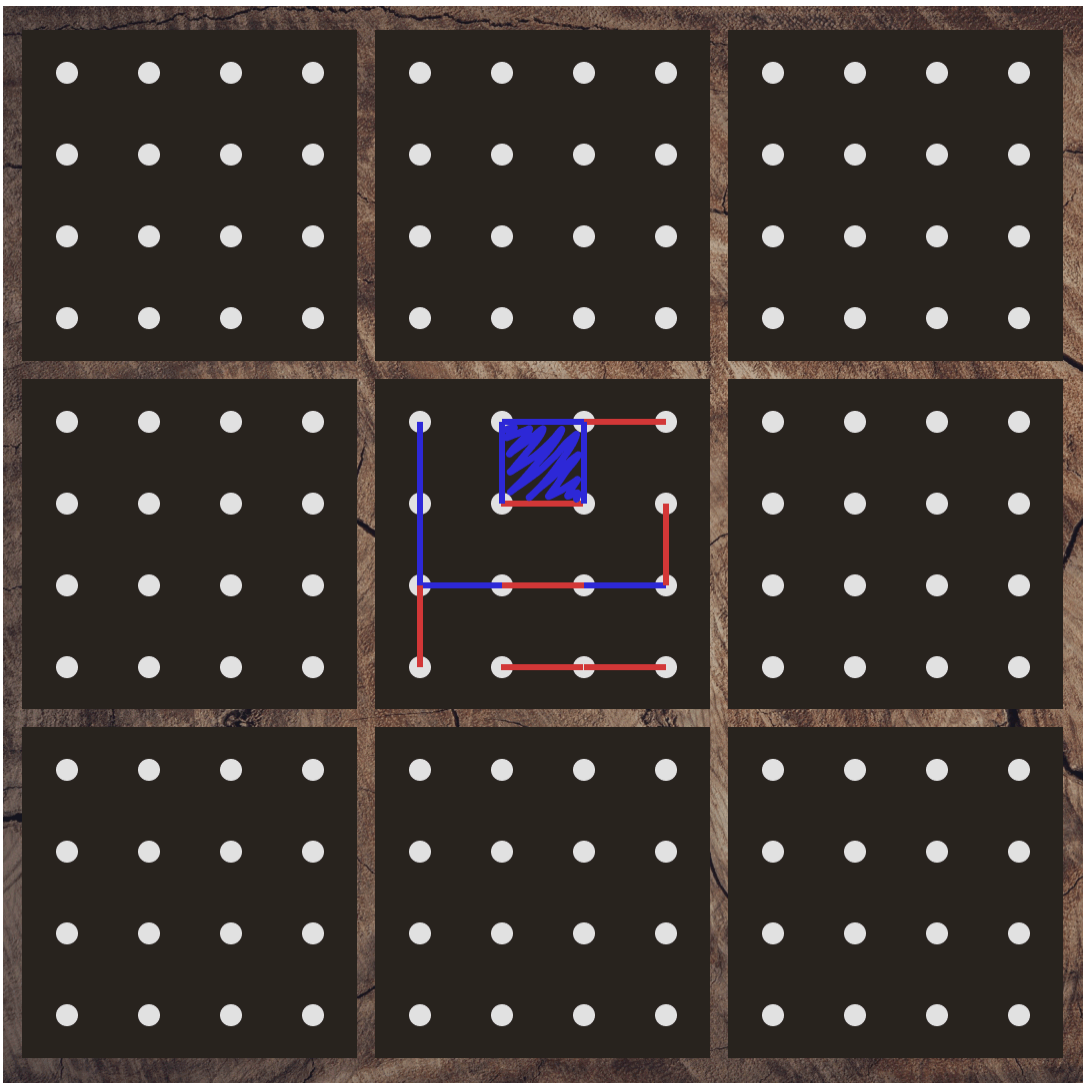

Figura 4 - Partida simulada de Ultimate Dots and Boxes

No demais, vale ressaltar o caso em que um dos jogadores completa um quadrado em um microjogo que direciona a jogada para um microjogo já completo, que seria impossível seguindo a lógica já descrita. Neste caso, o jogador seguinte pode escolher qualquer um dos microjogos incompletos para fazer sua jogada, seguindo a partir daí a lógica normal do jogo.

2. O que faz dele um jogo não simplista? Par ou ímpar, por exemplo, seria simplista. No jogo da velha, por exemplo, é a necessidade de elaboração de um algoritmo para determinação de final da partida e vencedor. E no seu jogo? O que o faz não simplista?

O nosso jogo não será simplista pois há diversas variáveis que devem ser acompanhadas ao longo da jogatina para que a lógica do jogo seja respeitada. Haverão algoritmos para:

- Checar o estado de cada microjogo a cada jogada, entendendo se ele terminou ou não.
- Direcionar o jogador para o microjogo correto dependendo da jogada anterior.

- Checar se o macrojogo está completo, através das condições de linha ou número de microjogos ganhos.
- Entender quando cada quadrado é preenchido dentro dos microjogos, atualizando a interface e o backend do jogo de maneira adequada.

Como todas essas checagens e atualizações devem ocorrer de maneira concorrente para que o jogo funcione de maneira adequada, há uma boa quantidade de lógica computacional - e portanto complexidade - na correta implementação do mesmo.

3. Mostre um esboço da interface do seu futuro programa (desenhado a mão, em editor de imagens, imagem obtida ou já uma implementação sob um suporte como Tkinter ou outro).

Para complementar as imagens já mostradas ao longo do relatório, abaixo está uma imagem congelada do GIF animado acima.

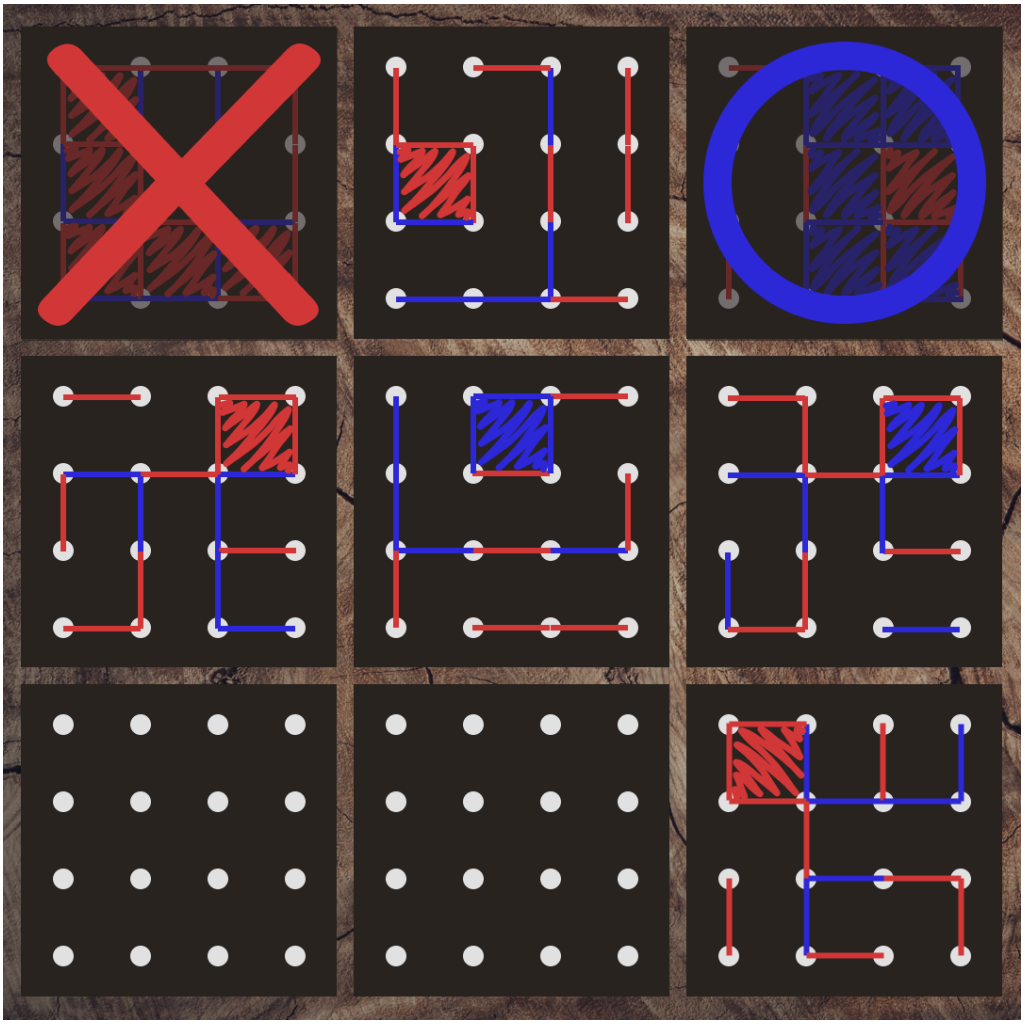

Figura 5 - Frame de uma partida em andamento do jogo.

4. Considerando o esboço da interface, mostre um rascunho de modelagem de casos de uso (quais as funcionalidades?) - usando Visual Paradigm ou não.

Diagrama no repositório do projeto.

5. Considerando o esboço da interface, mostre um rascunho de modelagem de classes (quais os elementos do domínio do problema?) - usando Visual Paradigm ou não.

Diagrama no repositório do projeto.