CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS

Linguagem e Técnicas de Programação 2

ALUNOS:

Davi Augusto Marçal Rodrigues

Matheus Augusto de Carvalho Silva

Samuel Matias Almeida Irineu

TRABALHO PRÁTICO: PONTOS E CAIXAS

Contagem, 2025

Sumário

Sumário **2**
1) Introdução **3**
2) Desenvolvimento **3**
2.1) Implementação 3
2.1.1 Classes Principais 4
2.1.1.1 Classe DotsAndBoxes 4
2.1.1.2. Classe Jogador 4
2.1.1.3. Classe Computador 4
2.1.1.4. Classe Pontos 4
2.1.1.5. Classe Linhas 4
2.1.1.6. Classe Quadrados 4
2.1.2. Relação entre Classes 5
2.2) Estratégias de Desenvolvimento 5
2.3) Principais Dificuldades 6
3) Considerações Finais **7**
4) Referências Bibliográficas **8**
Anexos **9**
ANEXO A - Repositório do Projeto no GitHub 9

1) Introdução

O trabalho tem como foco a criação de um jogo usando Programação Orientada a Objetos (POO) na linguagem de programação Java, juntamente com a biblioteca de recursos audiovisuais LibGDX. O projeto consiste no desenvolvimento de uma versão simplificada do jogo Pontos e Caixas. Os objetivos incluem trabalhar com POO e coletâneas de objetos. Além disso, aprender aspectos do *framework*[1] LibGDX para o desenvolvimento de aplicações multimídia.

Este relatório aborda o processo de desenvolvimento do projeto, explicando as estratégias, estruturas de dados e classes utilizadas no programa. Além disso, apresentará as principais dificuldades durante a execução e o resultado obtido.

2) Desenvolvimento

2.1) Implementação

O projeto criado é uma versão do jogo Pontos e Caixas. Inicialmente a tela apresenta um tabuleiro vazio de 6x6 pontos. O jogador e computador revezam entre si, marcando linhas horizontais ou verticais entre os pontos. A marcação do jogador é controlada pelo usuário, já a do computador é realizada em função da dificuldade selecionada, podendo ser competitiva ou aleatória. Quem completar a quarta aresta de um quadrado 1x1 pontua e ganha mais um turno, além disso, a figura completada é marcada com a inicial do nome do jogador ou do computador.

O jogo utilizado como referência foi o *Dots and Boxes*[2]. A jogabilidade se mantém semelhante, mas foram realizadas alterações gráficas e sonoras no jogo. A implementação se deu através das seguintes classes principais:

[1] Frameworks são estruturas de código reutilizáveis que fornecem elementos para ajudar no desenvolvimento de aplicações.
[2] COOLMATH. *Dots and Boxes*[jogo eletrônico online]. Nova York: Coolmath LLC, 2018. Disponível em: https://www.agame.com/game/dots-and-boxes.

2.1.1 Classes Principais

2.1.1.1 Classe DotsAndBoxes

A classe `DotsAndBoxes` é responsável por gerenciar a janela e os elementos do jogo. As principais funções exercidas pelas demais classes ocorrem nela, além disso, ela renderiza as linhas, quadrados e os pontos. Essa classe herda características da classe `Game` (definida na biblioteca LibGDX) que permite a manipulação de telas.

2.1.1.2. Classe Jogador

A classe `Jogador` é composta de dados e métodos de interação com as linhas. Ela controla o incremento de pontos e a realização de jogadas tanto do usuário como da máquina.

2.1.1.3. Classe Computador

A máquina representada pela classe `Computador` herda as características de `Jogador`, e também adiciona métodos para tornar a jogabilidade da máquina possível, realizando jogadas de forma aleatória.

2.1.1.4. Classe Pontos

A classe `Pontos` funciona como um gerenciador de pontos (classe `Ponto`), responsável por criá-los e desenhá-los na tela.

2.1.1.5. Classe Linhas

A classe `Linhas` é um gerenciador de linhas (classe `Linha`) que cria vetores de linhas horizontais e verticais e as imprime na tela. Além disso, verifica colisão entre elas e o jogador, e se o clique do jogador foi em cima de alguma das linhas definidas.

2.1.1.6. Classe Quadrados

A classe `Quadrados` gerencia quadrados (classe `Quadrado`), sendo responsável por criá-los e desenhá-los na tela. Além disso, verifica se cada um deles já foram desenhados, bem como a última aresta que falta para completá-lo.

2.1.2. Relação entre Classes

As relações entre as classes são funcionais. Relacionam-se através das colisões e no sistema de pontuação do jogo. Em algumas classes ocorre herança, como é o caso da classe `Computador` que herda os atributos da classe `Jogador`. Além disso, algumas classes herdam métodos de classes pré-definidas pela LibGDX, como é o caso da classe `DotsAndBoxes` que herda os métodos da classe `Game` ou mesmo das classes `TelaMenu` e `TelaReinicio` que herdam os métodos da classe `Screen`. A LIBGDX já disponibiliza métodos para manipulação de janelas e elementos na tela como `create()`, `render()`, `resize()` e `dispose()`.

2.2) Estratégias de Desenvolvimento

Durante o desenvolvimento do projeto diferentes técnicas foram utilizadas. O código foi separado em classes. Essa separação facilitou o seu entendimento, além disso, permitiu que o gerenciamento dos elementos do jogo fosse feito de forma organizada e intuitiva. O uso de coletâneas de dados em Java como `Array` mostrou-se útil, pois também contribuiu para o gerenciamento eficiente e organizado de tais elementos, que são os objetos.

Outra estratégia, foi nomear as classes, funções e variáveis com nomes significativos, o que facilitou a cooperação entre os participantes do projeto. Além disso, o uso de mecanismos de herança foi importante para reduzir repetições no código e enxugá-lo, bem como reutilizá-lo.

Por fim, a utilização do GitHub, plataforma de hospedagem de código, facilitou a colaboração entre os membros da equipe de desenvolvimento, garantindo que as alterações fossem facilmente acessíveis e documentadas. Isso foi útil para garantir que todos trabalhassem no mesmo código fonte, evitando redundâncias e conflitos de códigos.

2.3) Principais Dificuldades

Durante a criação do jogo, algumas dificuldades foram encontradas. Configurar a biblioteca LibGDX no ambiente de desenvolvimento, bem como compreender seu funcionamento revelou-se complicado. Os problemas foram contornado ao assistir os vídeos *EP2: Instalação e Configuração do Ambiente - Desenvolvendo Jogos em Java com LibGDX* [3] e *LibGdx - Desenhando uma matriz de retângulos na tela*[4] no Youtube que auxiliou na configuração e no entendimento do funcionamento da biblioteca.

Outra dificuldade foi desenhar os quadrados na tela, após a efetuação de um ponto. Para contorná-la, a classe `Quadrados` foi criada, determinando como cada quadrado deveria estar previamente posicionado na tela. Cada um deles é exibido assim que sua última aresta é desenhada.

Randomizar as linhas desenhadas pelo computador, tornando-as realmente aleatórias, também foi trabalhoso. A solução encontrada foi armazenar as linhas disponíveis para uma jogada em um `Array` e utilizar a função `random()` da classe `Math` que faz parte do pacote `java.lang` e realiza a escolha aleatória da linha a ser desenhada.

Além disso, para gerenciar as jogadas entre jogador e máquina em turnos, foi criado uma variável booleana que interagia com uma função booleana na classe `Jogador` e diz se o usuário já jogou, alternando a possibilidade de jogar entre máquina e *player*.

Por fim, fazer a tela de jogar novamente sem reiniciar o programa mostrou-se um desafio. Para solucionar essa situação utilizou-se a mesma lógica de implementação usada na tela inicial. Ambas são telas que sobrepõem o jogo (janela com o tabuleiro) até uma determinada ação. No caso da tela de reinício, a ação é clicar no botão “Reiniciar Jogo” ou “Voltar ao Menu”, na tela inicial é digitar o nome do jogador e a dificuldade do jogo.

---

[3] FERREIRA, Welsiton. *EP2: Instalação e Configuração do Ambiente - Desenvolvendo Jogos em Java com LibGDX*. YouTube, 31 mai. 2017. Disponível em: https://www.youtube.com/watch?v=0GhI_TTxSeM. Acesso em: 25 abril 2025.

[4] SANTOS, Alisson Rodrigo dos. *LibGdx - Desenhando uma matriz de retângulos na tela*. YouTube, 18 mar. 2023. Disponível em: https://www.youtube.com/watch?v=fd1Fa7r47mE. Acesso em: 02 maio 2025.

3) Considerações Finais

O resultado do projeto atendeu o que foi proposto. Uma versão do jogo *Dots And Boxes* foi criada, mantendo funcionalidades e jogabilidade semelhantes à referência proposta. Graficamente, está diferente do jogo de referência, nas fontes e cores. A parte sonora se manteve semelhante, sem alterações significativas.

O jogador consegue inserir o seu nome e a dificuldade que deseja jogar antes de iniciar a partida. Durante o jogo, consegue preencher adequadamente as linhas. Quando preenche a última aresta de um quadrado, esse se torna azul, cor que representa o jogador, além disso, a inicial do seu nome é imprimida no quadrado e ele ganha um ponto. O jogador alterna a vez de realizar a jogada com a máquina.

A máquina realiza jogadas aleatórias, mas quando há a possibilidade de preencher a última aresta de um quadrado, ela preenche, fazendo um ponto. A letra ‘C’ (inicial de computador) é desenhada no quadrado e esse se torna amarelo, cor que representa o computador. Quando a máquina ou o jogador faz um ponto eles ganham direito a mais uma jogada, sem passar a vez.

Por fim, quando o jogo acaba, aparece uma tela de reinício, que permite ao jogador o recomeço do jogo sem fechar e abrir novamente o programa. Quando o jogador clica em “Reiniciar Jogo", todos os elementos voltam ao seu estágio inicial, e na tela aparece novamente o tabuleiro vazio. Quando o jogador clica em “Voltar ao Menu”, ele é direcionado à tela inicial e pode digitar novamente seu nome e dificuldade que ele deseja jogar.

Embora todos os requisitos tenham sido alcançados, houve pretensões que não foram. Os métodos para verificar e desenhar os quadrados não foram feitos de forma satisfatória. Pode-se dizer que é de difícil entendimento a implementação desses métodos. Além disso, os quadrados são pré existentes, no entanto, a intenção era criar cada um deles de fato somente quando todas suas arestas fossem desenhadas.

Portanto, a criação do jogo utilizando Programação Orientada a Objetos e a biblioteca LibGDX foi bem sucedida. Mesmo com dificuldades durante a execução e com uma pretensão não alcançada, o resultado final atende às especificações estabelecidas. As mecânicas e funcionalidades dos elementos do jogo, bem como a parte audiovisual, funcionam conforme o esperado.

4) Referências Bibliográficas

COOLMATH. *Dots and Boxes*[jogo eletrônico online]. Nova York: Coolmath LLC, 2018. Disponível em: https://www.agame.com/game/dots-and-boxes.

FERREIRA, Welsiton. *EP2: Instalação e Configuração do Ambiente - Desenvolvendo Jogos em Java com LibGDX*. YouTube, 31 mai. 2017. Disponível em: https://www.youtube.com/watch?v=0GhI_TTxSeM. Acesso em: 25 abril 2025.

SANTOS, Alisson Rodrigo dos. *LibGdx - Desenhando uma matriz de retângulos na tela*. YouTube, 18 mar. 2023. Disponível em: https://www.youtube.com/watch?v=fd1Fa7r47mE. Acesso em: 02 maio 2025.

DALAL, Dhaval. *Dots-And-Boxes*. Box code. 2017. Disponível em: https://github.com/DhavalDalal/Dots-And-Boxes/blob/master/src/main/java/dots_and_boxes/Box.java. Acesso em: 10 maio 2025.

COOMBS, Jon; COSTA, André; PAUL, John. *Data structure for game Dots and Boxes*. 2015. Disponível em: https://stackoverflow.com/questions/4764787/data-structure-for-game-dots-and-boxes. Acesso em: 10 maio 2025.

PISKEL. Piskel - Free online sprite editor. Disponível em: https://www.piskelapp.com. Acesso em: 20 maio 2025

JUZKUS. DOTS & BOXES - Java Game. 2013. Disponível em: https://www.youtube.com/watch?v=HeGl6JtBsEU. Acesso em: 10 maio 2025.

WIKI. libGDX, [s.d.]. Disponível em: <https://libgdx.com/wiki/>. Acesso em: 13, abril, 2025

Anexos

ANEXO A - Repositório do Projeto no GitHub

O código-fonte completo do projeto está disponível no GitHub.

Link para o repositório: https://github.com/Grupo-8-2025/TrabalhoPratico1