



Leonardo Luz Fachel

Desenvolvimento de Jogo Sério para Auxílio no Ensino de Conceitos de Estruturas de Dados

Osório

2025

Leonardo Luz Fachel

**Desenvolvimento de Jogo Sério para Auxílio no Ensino de
Conceitos de Estruturas de Dados**

Orientador: Bruno Chagas Alves Fernandes

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul – IFRS
campus Osório

Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

Osório
2025

AGRADECIMENTOS

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Resumo

Este trabalho tem como objetivo apresentar os conceitos de estruturas de dados por meio de um jogo sério. O projeto busca facilitar o aprendizado dos alunos de forma lúdica e interativa. A metodologia utilizada foi baseada no desenvolvimento ágil, com foco em testes práticos. Os resultados demonstraram que a gamificação pode ser uma ferramenta eficaz no ensino de lógica e programação.

Palavras-chave: Jogo Sério. Estruturas de Dados. Ensino. Gamificação.

Abstract

This work aims to present the concepts of data structures through a serious game. The project seeks to facilitate student learning in a playful and interactive way. The methodology was based on agile development, focusing on practical testing. The results showed that gamification can be an effective tool in teaching logic and programming.

Keywords: Serious Game. Data Structures. Education. Gamification.

Lista de ilustrações

Figura 1 – Captura de tela do jogo CodingJob	32
Figura 2 – Captura de tela do jogo CodeBô	32
Figura 3 – Foto do tabuleiro de CodeBô Unplugged	33
Figura 4 – Captura de tela do jogo para auxiliar em estrutura de dados	34
Figura 5 – Captura de tela do tabuleiro de Prog-poly	35
Figura 6 – Captura de tela do jogo Human Resource Machine	37
Figura 7 – Captura de tela do jogo AlgoBot	38
Figura 8 – Captura de tela do jogo MOP’N SPARK	38
Figura 9 – Captura de tela do jogo Iron Ears	39

Lista de tabelas

Tabela 1 – Fatores que contribuem para as dificuldades no aprendizado da disciplina de estruturas de dados	13
Tabela 2 – Comparação entre os trabalhos relacionados	36
Tabela 3 – Comparação entre os jogos relacionados	40
Tabela 4 – Conversão do Sistema de Avaliação da Steam para um sistema numeral de 1 a 5	40

LISTA DE ABREVIATURAS E SIGLAS

GIMP	<i>GNU Image Manipulation Program</i>
GNU	<i>GNU's Not Unix</i>
ILPC	Introdução Linguagem de Programação C
ENgAGED	<i>EducatioNAL GamEs Development</i>
LIFO	<i>Last In First Out</i>
FIFO	<i>First In First Out</i>

Sumário

1	INTRODUÇÃO	11
1.1	Objetivo Geral	12
1.2	Objetivos Específicos	12
1.3	Justificativa	13
2	REFERENCIAL TEÓRICO	14
2.1	Estruturas de Dados	14
2.2	Jogos Sérios	14
2.3	Aprendizagem Baseada em Jogos	15
2.4	Construcionismo	15
2.5	Unity	16
3	METODOLOGIA	17
3.1	Metodologia Científica	17
3.1.1	Classificação Metodológica	17
3.1.2	Etapas da Pesquisa	18
4	METODOLOGIA DE DESENVOLVIMENTO	19
4.1	Fases do Processo ENgaged	19
4.1.1	Fase 1 - Análise da Unidade Instrucional	19
4.1.2	Fase 2 - Projeto da Unidade Instrucional	20
4.1.3	Fase 3 - Desenvolvimento do Jogo Educacional	22
4.1.4	Fase 4 - Execução da Unidade Instrucional	28
4.1.5	Fase 5 - Avaliação da Unidade Instrucional	28
4.2	Síntese da Metodologia	29
5	TRABALHOS RELACIONADOS	31
5.1	Artigos	31
5.1.1	Comparação dos artigos	35
5.2	Aplicativos	36
5.2.1	Comparação dos aplicativos	39
5.3	Síntese	41
6	DESENVOLVIMENTO	42
6.1	Arquitetura e Estrutura do Projeto	42
6.1.1	Estruturas de Dados Implementadas	42
6.2	Sistemas Principais do Jogo	43

6.2.1	Sistema de Movimento	43
6.2.2	Sistema de Combate	44
6.2.3	Sistema de Inventário	44
6.2.4	Sistema de Vida e Penalidades	44
6.2.5	Sistema de Progresso e Salvamento	45
6.2.6	Sistema de Feedback	45
6.3	Interface de Usuário	46
6.3.1	Layout Principal do Jogo	46
6.3.2	Menus Implementados	46
6.4	Implementação de Fases	47
6.4.1	Fase 1 - Pilha	47
6.4.2	Fase 2 - Fila	47
6.4.3	Fase 3 - Lista	47
6.4.4	Fase 4 - Consumíveis e Ordenação	48
6.4.5	Fase 5 - Integração	48
6.5	Implementação de Inimigos	48
6.5.1	Tipos de Inimigos	48
6.5.2	Sistema de Vulnerabilidades	48
6.5.3	Estatísticas Randomizadas	49
6.6	Implementação de Consumíveis	49
6.6.1	Ordenação	49
6.6.2	Insert	49
6.6.3	Remove	49
6.6.4	Mana	49
6.6.5	Cura	49
6.6.6	Vida Extra	50
6.7	Desenvolvimentos Técnicos Realizados	50
6.7.1	Persistência de Dados	50
6.7.2	Gerenciamento de Câmera	50
6.7.3	Sistema de Diálogos	50
6.7.4	Animações e Efeitos Visuais	50
6.8	Desafios Enfrentados e Soluções	51
6.8.1	Balanceamento de Dificuldade	51
6.8.2	Clareza das Mecânicas	51
6.8.3	Performance	51
6.8.4	Tratamento de Estados	51
6.9	Testes Realizados	51
6.9.1	Testes Funcionais	51

6.9.2	Testes de Usabilidade	52
6.9.3	Testes Educacionais	52
6.10	Conformidade com Requisitos	52
6.11	Resultados do Desenvolvimento	53
7	CONSIDERAÇÕES FINAIS	54
	REFERÊNCIAS	55

1 INTRODUÇÃO

O ensino e a aprendizagem de conceitos fundamentais da área de computação, como estruturas de dados, constituem um desafio recorrente para educadores e estudantes. De acordo com o autor [Mtaho e Mselle \(2024\)](#), a disciplina de estruturas de dados é altamente exigente para estudantes de ciência da computação, sendo frequentemente associada a uma elevada carga cognitiva e, consequentemente, a altas taxas de reprovação e evasão do curso. Entre os principais fatores que contribuem para essas dificuldades estão a natureza abstrata dos conceitos envolvidos e a baixa motivação dos alunos. Esse cenário se agrava pelo fato de que, tradicionalmente, o ensino desses conteúdos ocorre por meio de aulas expositivas e exercícios de codificação, o que tende a gerar baixa retenção do conteúdo e desinteresse por parte dos estudantes.

Além disso, as novas gerações de estudantes estão cada vez mais habituadas a um fluxo constante de informações e experiências interativas, desenvolvendo um comportamento que valoriza respostas rápidas e estímulos visuais. Isso torna o ensino convencional ainda menos atrativo. **A partir de sua pesquisa, o autor Mtaho e Mselle (2024) recomenda a adoção de novas estratégias de ensino para mitigar as dificuldades e melhorar a experiência de aprendizagem de estrutura de dados.**

Nesse cenário, os jogos sérios surgem como uma estratégia educacional promissora, ao promover o aprendizado ativo e engajado. Diferentemente de abordagens instrucionais diretas, os jogos sérios podem ser projetados para que a aprendizagem ocorra como consequência da interação do jogador com o ambiente, desafios e regras do jogo. Essa perspectiva está alinhada à teoria do construcionismo, proposta por [Papert \(1993\)](#), segundo a qual o conhecimento é construídoativamente pelos alunos quando estes se envolvem com a criação, exploração e manipulação de artefatos significativos.

Jogos sérios são definidos como uma aplicação de videogames cujo objetivo principal é educar, treinar ou sensibilizar, sem abrir mão do entretenimento ([MOUAHEB et al., 2012](#)). Contudo, uma crítica recorrente a essa abordagem é que muitos desses jogos sérios falham como jogos, priorizam o conteúdo educativo de forma explícita, relegando a experiência lúdica a segundo plano, **quando, na verdade, ensino e entretenimento deveriam caminhar lado a lado (MOUAHEB et al., 2012).**

De acordo com [Araujo e Silva \(2025\)](#), atualmente, grande parte dos jogos sérios se utilizam os conceitos de programação apenas como tema, sem integrá-los verdadeiramente às suas mecânicas. Essa limitação evidencia um modelo que tende a transformar o jogo em um pretexto para ensinar diretamente, por meio de **mecânicas expositivas como** questionários ou simulações superficiais.

O presente trabalho propõe uma abordagem alternativa: utilizar mecânicas de jogo que representem, de forma implícita e interativa, conceitos fundamentais de estruturas de dados. Em vez de apresentar diretamente listas, pilhas ou filas, o jogo incorporará esses elementos em sua lógica e estrutura interna, permitindo que o jogador interaja com tais conceitos de forma intuitiva e contextualizada. Dessa maneira, o aprendizado ocorre como consequência da resolução de problemas e da exploração do sistema, e não como resultado de instruções explícitas ou desafios de programação.

Diferentemente de jogos educativos que simulam exercícios de codificação, o objetivo deste trabalho é projetar um jogo no qual os conceitos ensinados estejam presentes nas ações tomadas pelo jogador, mesmo que ele não os reconheça explicitamente como tais. Na próxima seção, o objetivo geral deste trabalho será aprofundado.

1.1 OBJETIVO GERAL

Este trabalho tem como objetivo geral desenvolver um jogo sério que aborde conceitos fundamentais de estruturas de dados de forma implícita, por meio de mecânicas lúdicas e interativas. A proposta busca promover um processo de aprendizagem mais significativo, intuitivo e motivador.

1.2 OBJETIVOS ESPECÍFICOS

Com base no objetivo geral, este trabalho também visa alcançar os seguintes objetivos específicos:

- Investigar modelos de jogos sérios e sua aplicação no ensino de conteúdos relacionados à computação;
- Projetar e implementar um jogo educacional fundamentado na metodologia ENgAGED;
- Incorporar, nas mecânicas do jogo, representações implícitas de estruturas de dados, como listas, filas e pilhas, além de algoritmos de busca e ordenação;
- Avaliar a usabilidade e a eficácia do jogo no processo de ensino e aprendizagem;
- Coletar e analisar o *feedback* dos usuários com o intuito de orientar futuras melhorias da ferramenta desenvolvida.

1.3 JUSTIFICATIVA

A proposta deste trabalho encontra respaldo na demanda por tornar o ensino de estruturas de dados mais motivador e alinhado às expectativas das novas gerações de aprendizes. O uso de jogos sérios como recurso educacional permite contextualizar os conceitos dentro de uma narrativa envolvente, aumentando o engajamento e favorecendo a construção do conhecimento de forma mais prática e intuitiva ([MOUAHEB et al., 2012](#)).

Além disso, a adoção da metodologia ENgAGED ([BATTISTELLA; WANGENHEIM, 2016](#)) no processo de desenvolvimento garante uma abordagem sistemática e centrada no aprendizado, permitindo que os objetivos educacionais sejam alcançados sem comprometer a experiência lúdica.

Dessa forma, este trabalho justifica-se por buscar uma alternativa para o ensino de estruturas de dados, conforme recomendado por [Mtaho e Mselle \(2024\)](#), pretendendo contribuir para a formação de profissionais mais preparados, criativos e capazes de aplicar o conhecimento de maneira prática e estratégica no mercado de trabalho.

2 REFERENCIAL TEÓRICO

Neste capítulo, serão abordados os conceitos e as tecnologias fundamentais que embasam este trabalho, conectando-os diretamente aos desafios de ensino e aprendizagem de estruturas de dados e à proposta de um jogo sério com aprendizagem implícita.

2.1 ESTRUTURAS DE DADOS

As estruturas de dados representam um pilar fundamental na ciência da computação, sendo essenciais para a organização, gestão e armazenamento eficiente de informações, o que, por sua vez, viabiliza o desenvolvimento de algoritmos otimizados e a construção de *software* robusto e escalável (CORMEN et al., 2022). Contudo, a aprendizagem desses conceitos constitui um desafio significativo para estudantes da área.

Conforme destacado por Mtaho e Mselle (2024), a disciplina de estruturas de dados é frequentemente associada a uma elevada carga cognitiva e a altas taxas de reprovação e evasão em cursos de computação. As dificuldades advêm, em grande parte, da natureza abstrata dos conceitos envolvidos e da baixa motivação dos alunos. O modelo de ensino tradicional, que se baseia em aulas expositivas e exercícios de codificação, muitas vezes não consegue proporcionar a retenção efetiva do conteúdo nem despertar o interesse necessário para a compreensão aprofundada. Essa lacuna pedagógica ressalta a urgência de explorar e implementar estratégias de ensino inovadoras que possam mitigar essas barreiras e enriquecer a experiência de aprendizagem.

2.2 JOGOS SÉRIOS

Jogos sérios (*serious games*) são definidos como aplicações interativas que utilizam o design e a tecnologia dos videogames para fins que vão além do puro entretenimento, como educação, treinamento ou conscientização (MOUAHEB et al., 2012). Eles surgem como uma alternativa promissora às metodologias de ensino tradicionais, buscando engajar os alunos em ambientes de aprendizagem imersivos e motivadores.

Contudo, a eficácia de um jogo sério não reside apenas em seu conteúdo educacional, mas na sua capacidade de integrá-lo de forma coesa à experiência lúdica. Uma crítica recorrente a muitos jogos com propósito educacional é que eles falham em ser bons jogos, priorizando a instrução direta em detrimento da jogabilidade (MOUAHEB et al., 2012). De acordo com Araujo e Silva (2025), muitos jogos sérios para o ensino de programação, por exemplo, utilizam os conceitos apenas como tema, sem incorporá-los profundamente em suas mecânicas centrais. Isso resulta em experiências que se assemelham mais a questionários interativos ou a exercícios

de codificação disfarçados, perdendo o potencial transformador que a mídia dos jogos pode oferecer. A proposta deste trabalho busca superar essa limitação, integrando os conceitos de estruturas de dados de forma implícita nas mecânicas do jogo, de modo que o aprendizado ocorra naturalmente pela interação e resolução de desafios, e não por meio de explicações explícitas.

2.3 APRENDIZAGEM BASEADA EM JOGOS

A Aprendizagem Baseada em Jogos, do inglês *Game-Based Learning* (GBL), é uma estratégia pedagógica que emprega jogos completos como ferramentas para alcançar objetivos de aprendizagem específicos. Diferencia-se da gamificação, que apenas aplica elementos de jogos (como pontos, medalhas e *rankings*) a contextos não lúdicos. Na GBL, o aprendizado emerge da própria interação do jogador com as mecânicas, sistemas e narrativas do jogo ([MALONE; LEPPER, 2021](#)).

O potencial da GBL reside na sua capacidade de criar um ciclo de aprendizado motivado intrinsecamente. Um jogo bem projetado desafia o jogador, oferece *feedback* constante e permite a experimentação em um ambiente seguro, onde o erro é parte do processo de descoberta. Em vez de receber informações de forma passiva, o jogador aprende ativamente ao testar hipóteses, resolver problemas e superar obstáculos. Essa abordagem é particularmente relevante para conceitos abstratos como os de estruturas de dados, pois permite que os alunos visualizem e manipulem representações concretas desses conceitos dentro do universo do jogo, promovendo um engajamento que o ensino tradicional muitas vezes não consegue ([MTAHO; MSELLE, 2024](#)).

2.4 CONSTRUCIONISMO

A fundamentação pedagógica deste trabalho está ancorada na teoria do Construcionismo de Seymour Papert. Derivada do construtivismo de Piaget, a teoria de Papert postula que a aprendizagem é mais eficaz quando o aprendiz está conscientemente engajado na construção de um artefato público e tangível, seja ele um castelo de areia, um poema, uma máquina ou um programa de computador ([PAPERT, 1993](#)). Essa perspectiva é crucial para o desenvolvimento de ambientes de aprendizagem que promovam a autonomia e a descoberta.

O Construcionismo defende que o conhecimento não é algo a ser simplesmente transmitido, mas algo a ser construído e reconstruído pelo indivíduo por meio de ações e interações com o mundo. Nesse contexto, o jogo proposto neste trabalho pode ser visto como um "micro-mundo" de aprendizagem, um ambiente onde os jogadores constroem seu entendimento sobre estruturas de dados não por meio de instrução direta, mas ao manipular os sistemas do jogo para resolver problemas. As soluções que o jogador cria dentro do jogo são os artefatos que

refletem e solidificam seu aprendizado. Essa abordagem se alinha perfeitamente à proposta de um aprendizado implícito, onde o conhecimento é uma consequência direta da experiência e da ação, e não o seu pré-requisito, conforme a metodologia de ensino que busca desacoplar o conceito ensinado de uma linguagem de programação específica, focando na compreensão conceitual através da interação lúdica.

2.5 UNITY

Para a concretização da proposta pedagógica deste trabalho, que visa o ensino implícito de estruturas de dados por meio de um jogo sério, a *game engine* Unity foi selecionada como a principal ferramenta de desenvolvimento. A Unity é um ambiente de desenvolvimento multiplataforma amplamente reconhecido na indústria de jogos, simulações e aplicações interativas, destacando-se por sua flexibilidade e robustez.

A escolha da Unity é fundamentada em suas características que se alinharam diretamente aos objetivos do projeto. Primeiramente, sua arquitetura baseada em componentes e a utilização da linguagem C# permitem a criação de mecânicas de jogo sofisticadas e a representação abstrata de estruturas de dados de forma eficiente. Isso é crucial para a implementação de um aprendizado implícito, onde os conceitos são integrados à lógica do jogo sem serem explicitamente ensinados. Em segundo lugar, a capacidade multiplataforma da Unity garante que o jogo possa ser acessado por um público mais amplo de estudantes, independentemente do sistema operacional. Por fim, a vasta comunidade de desenvolvedores e a rica documentação disponível para a Unity aceleram o processo de desenvolvimento, permitindo uma maior concentração no design da experiência de aprendizagem.

3 METODOLOGIA

Segundo [Creswell e Creswell \(2021\)](#), um projeto de pesquisa é um plano estruturado que orienta todo o processo investigativo, abrangendo desde a formulação do problema e os objetivos do estudo até os procedimentos de coleta e análise de dados. A definição da metodologia é essencial, pois fornece ao pesquisador um caminho claro para desenvolver o estudo de maneira coerente e fundamentada. Diante disso, este capítulo apresenta os métodos de pesquisa e desenvolvimento utilizados, detalhando as escolhas metodológicas adotadas ao longo do trabalho.

3.1 METODOLOGIA CIENTÍFICA

A metodologia científica adotada neste trabalho é de natureza experimental, com abordagem mista, qualitativa e quantitativa, e classifica-se como uma pesquisa aplicada. A próxima seção detalha e justifica essas escolhas.

3.1.1 Classificação Metodológica

Este trabalho caracteriza-se como uma pesquisa aplicada, pois tem como objetivo solucionar um problema prático relacionado ao ensino de estruturas de dados por meio da utilização de um jogo sério ([MOUAHEB et al., 2012](#)). Diferentemente da pesquisa puramente teórica, a pesquisa aplicada visa gerar conhecimento com aplicação direta em contextos específicos. Neste caso, o foco está no ambiente educacional.

A natureza experimental da pesquisa se deve ao fato de propor uma intervenção concreta, que envolve o desenvolvimento e a aplicação de um protótipo funcional de jogo em um ambiente controlado com usuários reais. O objetivo é observar os efeitos dessa intervenção no processo de aprendizagem.

Adota-se uma abordagem mista, combinando métodos qualitativos e quantitativos com o intuito de proporcionar uma análise mais abrangente e precisa dos resultados. Os dados quantitativos são coletados por meio de instrumentos como questionários estruturados e testes de desempenho, permitindo uma avaliação objetiva. Já os dados qualitativos são obtidos por meio de observações, entrevistas e análise do comportamento dos participantes durante a interação com o jogo. Isso permite compreender de forma mais aprofundada a experiência dos usuários e a eficácia da ferramenta no processo de ensino e aprendizagem.

3.1.2 Etapas da Pesquisa

Esta pesquisa foi composta por diversas etapas e teve início com uma revisão bibliográfica, por meio da qual foram identificadas publicações acadêmicas e aplicações relacionadas ao uso de jogos sérios no ensino de conceitos de programação. Essa etapa proporcionou uma visão das abordagens já utilizadas, seus resultados e os conceitos mais frequentemente aplicados.

Em seguida, foi idealizado e desenvolvido um jogo sério educacional com base na metodologia ENgAGED ([BATTISTELLA; WANGENHEIM, 2016](#)). Esse processo envolveu diversas fases como a concepção, implementação e aplicação do jogo proposto, seguido da coleta de dados por meio de testes com usuários. Nessa etapa, foram empregadas técnicas qualitativas e quantitativas para avaliar a experiência dos participantes e a eficácia da ferramenta como recurso educacional.

Por fim, os resultados foram validados. Esta etapa consistiu no cruzamento dos dados obtidos com os objetivos da pesquisa, com o propósito de verificar se o jogo contribuiu de maneira significativa para o processo de ensino e aprendizagem de estruturas de dados.

4 METODOLOGIA DE DESENVOLVIMENTO

Para o desenvolvimento do jogo educacional foi adotada a metodologia ENgAGED, proposta por [Battistella e Wangenheim \(2016\)](#). O processo integra princípios de design instrucional com design de jogos, estruturando o desenvolvimento em fases sistemáticas que asseguram coerência pedagógica, engajamento sustentado e avaliação contínua. Essa abordagem é particularmente adequada para projetos que buscam equilibrar objetivos educacionais com experiências lúdicas envolventes, evitando a falha comum de priorizar apenas um dos aspectos em detrimento do outro.

4.1 FASES DO PROCESSO ENGAGED

O processo ENgAGED estrutura-se em cinco fases principais, cada uma com objetivos e artefatos específicos. A seguir, cada fase é descrita e contextualizada para o desenvolvimento do jogo proposto.

4.1.1 Fase 1 - Análise da Unidade Instrucional

A primeira fase concentra-se na análise do contexto educacional, da definição do público-alvo e na especificação dos objetivos de aprendizagem que orientarão todo o desenvolvimento.

A1.1 - Especificação da Unidade Instrucional

A unidade instrucional foi definida como o ensino de conceitos fundamentais de estruturas de dados (pilha, fila e lista) no contexto de cursos de graduação em Análise e Desenvolvimento de Sistemas ou disciplinas correlatas de Computação. O foco está em permitir que os aprendizes compreendam o funcionamento dessas estruturas e suas operações básicas (inserção, remoção e ordenação) de forma prática e contextualizada.

Os objetivos de aprendizagem específicos incluem:

- Compreender o funcionamento de estruturas de dados lineares (pilha, fila e lista);
- Identificar as diferenças entre operações LIFO (último a entrar, primeiro a sair) e FIFO (primeiro a entrar, primeiro a sair);
- Aplicar conceitos de manipulação de estruturas em cenários de resolução de problemas;

A1.2 - Caracterização dos Aprendizes

O público-alvo consiste em:

- Estudantes de graduação em Análise e Desenvolvimento de Sistemas ou cursos correlatos de Computação;
- Idade aproximada entre 18 e 30 anos;
- Acesso a computadores pessoais com sistema operacional Windows ou Linux e entrada via teclado e mouse.

Embora o jogo tenha sido projetado com esse público em mente, sua acessibilidade permite que qualquer pessoa interessada em jogos de forma geral possa participar, sem exigir conhecimento prévio explícito de estruturas de dados.

A1.3 - Definição dos Objetivos de Desempenho

O objetivo de desempenho do jogo é que, ao final da experiência, o jogador seja capaz de:

- Executar operações de inserção e remoção respeitando as regras de cada estrutura;
- Entender e aplicar conceitos de ordenação;
- Reconhecer padrões de funcionamento das estruturas através da interação implícita com o jogo.

4.1.2 Fase 2 - Projeto da Unidade Instrucional

A segunda fase dedica-se ao design instrucional, definindo como os conteúdos serão apresentados, como será realizada a avaliação e quais estratégias pedagógicas serão empregadas.

A2.1 - Definição da Avaliação do Aprendiz

A avaliação é incorporada diretamente nas mecânicas do jogo, funcionando como mecanismo de feedback contínuo:

- **Avaliação Formativa Implícita:** ao realizar combinações corretas de elementos alquímicos, o jogador recebe feedback positivo instantâneo (sucesso na ação);

- **Avaliação por Penalidade:** combinações incorretas resultam em penalidade (perda de pontos de vida), estimulando o jogador a refletir sobre suas ações e ajustar sua estratégia;
- **Métricas de Desempenho:** ao final de cada fase, são registradas métricas como tempo de conclusão, tentativas, erros e acertos de combinações.

Essa abordagem de avaliação está alinhada ao construcionismo, pois permite que o aprendiz construa seu conhecimento através da exploração, tentativa e erro, em um ambiente seguro onde o fracasso é parte do processo de descoberta.

A2.2 - Definição do Conteúdo e da Estratégia Instrucional

O conteúdo é distribuído progressivamente através de fases de dificuldade crescente:

- **Fase 1 - Pilha:** introduz o conceito LIFO. O jogador manipula uma estrutura de pilha para combinar elementos de fogo, água, ar ou terra. Apenas a remoção de elementos do topo é permitida, refletindo o comportamento de uma pilha.
- **Fase 2 - Fila:** apresenta o conceito FIFO. O jogador utiliza uma estrutura de fila, onde os elementos devem ser removidos na ordem em que foram inseridos, integrando novos inimigos com vulnerabilidades específicas.
- **Fase 3 - Lista:** introduz manipulações mais dinâmicas. O jogador pode inserir e remover elementos em qualquer posição, representando a flexibilidade de uma lista encadeada.
- **Fase 4 - Consumíveis e Ordenação:** apresenta o item de ordenação, que reorganiza a estrutura, e outros consumíveis que auxiliam na progressão. Integra conceitos de algoritmos de ordenação de forma implícita.
- **Fase 5 - Integração:** os desafios finais combinam as três estruturas simultaneamente, exigindo que o jogador compreenda quando aplicar cada uma e como coordenar ataques complexos.

A estratégia instrucional priorizará o **aprendizado implícito**: os conceitos não são apresentados verbalmente ou textualmente, mas descobertos através da interação com as mecânicas do jogo. Dessa forma, a aprendizagem emerge da necessidade prática de resolver desafios, alinhando-se à filosofia construcionista.

A2.3 - Decisão sobre Desenvolvimento ou Reutilização

Optou-se pelo **desenvolvimento original do jogo**. Essa decisão baseou-se em:

- Inexistência de jogos disponíveis que integrem a mecânica de estruturas de dados com temática de alquimia e estilo visual de pixel art inspirado em clássicos como Mario e Mega Man;
- Necessidade de controle total sobre a implementação das estruturas de dados para garantir precisão pedagógica;
- Oportunidade de customizar completamente a narrativa, visual e mecânicas para alinhar com os objetivos educacionais específicos;
- Potencial de reaproveitamento futuro do código e assets desenvolvidos em outros projetos.

A2.4 - Revisão do Modelo de Avaliação

O modelo de avaliação do jogo segue os princípios do **MEEGA+** (Modelo para Avaliação de Jogos Educacionais), abordando dimensões de:

- **Usabilidade:** facilidade de aprender e usar o jogo, clareza da interface;
- **Engajamento:** capacidade de manter o interesse do jogador, imersão, diversão;
- **Aprendizagem Percebida:** percepção do jogador sobre o aprendizado obtido, aplicabilidade dos conceitos;
- **Experiência do Usuário:** satisfação geral, disposição para recomendação do jogo.

Os feedbacks são implementados de forma imediata no jogo, reforçando a aprendizagem e orientando o jogador através de:

- Feedback visual (mudanças de cor, animações, efeitos particulares);
- Feedback sonoro (sons de sucesso ou erro);
- Mensagens textuais contextualizadas que reforçam conceitos sem ser excessivamente expositivas.

4.1.3 Fase 3 - Desenvolvimento do Jogo Educacional

Esta fase abrange as etapas práticas de implementação: levantamento de requisitos, concepção visual e narrativa, design de mecânicas e implementação técnica.

A3.1 - Análise do Jogo e Levantamento de Requisitos

Os requisitos foram estruturados em duas categorias: funcionais e não-funcionais.

Requisitos Funcionais

Os requisitos funcionais definem as funcionalidades que o sistema deve implementar. Esses requisitos estão detalhados no ?? e incluem aspectos como manipulação de inventários, sistema de combate, progresso do jogador, salvamento de partida e geração de feedback educacional.

Requisitos Não-Funcionais

Os requisitos não-funcionais especificam características de qualidade e restrições técnicas. Esses requisitos estão detalhados no ?? e abrangem aspectos como performance, compatibilidade, segurança, mantibilidade e acessibilidade.

A3.2 - Concepção do Jogo

Gênero e Perspectiva

O jogo é um **platformer 2D** (jogo de plataforma em duas dimensões) com perspectiva *side-scrolling*. Essa escolha baseia-se na familiaridade do público geral com clássicos como Super Mario Bros e Mega Man, facilitando a adoção do jogo e tornando a experiência intuitiva.

Temática e Narrativa

A narrativa é ambientada em um universo de **alquimia** mística e medieval. O protagonista é um *plague doctor* (doutor da peste) que teve sua pesquisa sobre a **pedra filosofal** roubada por um alquimista rival enquanto viajava para a capital para apresentá-la. Seu objetivo é recuperar a pesquisa, navegando por diferentes regiões e enfrentando adversários durante a sua jornada.

Essa narrativa fornece contexto e motivação para as ações do jogador, aumentando o engajamento e criando imersão. O enredo também justifica a presença dos elementos alquímicos como mecanismo central de combate.

Mecânicas Centrais

As mecânicas principais do jogo são:

- **Movimentação e Exploração:** o jogador controla o plague doctor, movendo-se horizontalmente pelos níveis, pulando e interagindo com o ambiente;

- **Sistema de Inventários Estruturados:** o jogador possui três inventários independentes (pilha, fila e lista), cada um representando uma estrutura de dados específica. Os elementos alquímicos (fogo, água, ar, terra) são armazenados nesses inventários;
- **Combinação de Elementos:** para atacar, o jogador deve remover elementos dos inventários e combiná-los corretamente. Por exemplo:
 - Dois elementos fogo combinados geram um ataque de fogo;
 - Dois elementos água combinados geram um ataque de água;
 - A combinação respeitará as regras da estrutura de origem (LIFO para pilha, FIFO para fila, acesso livre para lista).
- **Vulnerabilidades de Inimigos:** cada inimigo possui vulnerabilidades específicas a um ou mais elementos. Ataques com o elemento correto causam dano;
- **Consumíveis:** itens especiais encontrados durante o jogo auxiliam na progressão:
 - **Ordenação:** ordena a estrutura de dados, permitindo reorganizar elementos;
 - **Insert:** insere dois elementos iguais em um inventário específico;
 - **Remove:** remove um elemento de um inventário específico;
 - **Mana:** concede mana infinita por um tempo determinado;
 - **Cura:** cura 1 ponto de vida;
 - **Vida Extra:** aumenta a quantidade de tentativas em 1.

A3.3 - Design do Jogo

Elementos Visuais e Artísticos

O jogo utiliza *pixel art* como estilo visual, em homenagem aos clássicos de 8 e 16 bits, facilitando a conexão com o público-alvo e permitindo desenvolvimento mais ágil. Os sprites do protagonista, inimigos, objetos e cenários foram criados em pixel art original, mantendo coerência visual e legibilidade.

A paleta de cores foi escolhida para refletir a temática do personagem principal, seguindo majoritariamente tons de azul e roxo, tanto mais escuros quanto pastéis.

Interface e Comunicação Visual

A interface foi projetada com princípios de **minimalismo** e **clareza**, evitando sobrecarga visual:

- **Inventário Visual:** os inventários são exibidos na parte inferior à direita, mostrando os elementos armazenados em cada estrutura através de representações visuais claras;
- **Feedback Visual de Erros:** quando uma combinação incorreta é realizada, o jogo exibe feedback visual (mudança de cor do jogador e efeito de penalidade) sem interromper a fluidez do *gameplay*;
- **Indicadores de Vulnerabilidade:** inimigos exibem visualmente seus elementos de vulnerabilidade através de ícones, permitindo que o jogador identifique rapidamente a estratégia apropriada.

Sistema de Pontuação e Progressão

A pontuação é baseada em:

- **Eficiência:** quantos ataques bem-sucedidos foram realizados em relação ao total;
- **Tempo:** bônus por completar a fase rapidamente;
- **Sem Dano:** bônus adicional se o jogador completar a fase sem perder vidas.

O progresso é salvo automaticamente em pontos de controle (checkpoints) estratégicamente posicionados em cada fase, evitando frustração causada por perda de progresso.

A3.4 - Implementação Técnica

Ferramentas e Tecnologias

- **Motor de Jogo:** Unity 2022 LTS, selecionada por sua robustez, suporte multiplataforma e capacidade de criar jogos de performance eficiente;
- **Linguagem de Programação:** C#, linguagem nativa da Unity, utilizada para implementar toda a lógica do jogo, incluindo estruturas de dados e sistemas de *gameplay*;
- **Criação de Assets Visuais:** GIMP (GNU Image Manipulation Program), software livre utilizado para criar e editar sprites em pixel art;
- **Controle de Versão:** Git com repositório remoto, garantindo rastreamento de alterações e possibilidade de recuperação de versões anteriores.

Arquitetura de Código

As estruturas de dados (pilha, fila e lista) foram implementadas como classes independentes em C#, permitindo:

- Reaproveitamento de código entre diferentes contextos do jogo;
- Clareza didática: o código das estruturas reflete diretamente os conceitos ensinados;
- Testes unitários simplificados para validar o comportamento das estruturas.

Cada classe implementa operações fundamentais:

- **Stack (Pilha):** Push (), Pop (), Peek (). Respeitando a ordem LIFO.
- **Queue (Fila):** Enqueue (), Dequeue (), Peek (). Respeitando a ordem FIFO.
- **List (Lista):** Add (), Insert (), RemoveAt (), Get (). Permitindo acesso livre em qualquer posição.

Sistemas de Jogo Implementados

- **Sistema de Movimento:** controle do personagem (esquerda, direita, pulo), colisão com plataformas e inimigos;
- **Sistema de Combate:** detecção de combinações de elementos, validação contra regras de estruturas, cálculo de dano;
- **Sistema de Inventário:** gerenciamento de elementos nos inventários, suporte a operações específicas de cada estrutura;
- **Sistema de Vida:** rastreamento da vida do jogador, aplicação de penalidades, condição de derrota;
- **Sistema de Progresso:** salvamento e carregamento de estado, gestão de fases desbloqueadas, registro de pontuação;
- **Sistema de Feedback:** implementação de mensagens visuais, sonoras e textuais que reforçam o aprendizado.

A3.5 - Testes do Jogo

Os testes foram conduzidos em múltiplas etapas:

Testes Funcionais

Testes para validar se todas as funcionalidades implementadas operam conforme especificado:

- Operações de estruturas de dados (inserção, remoção, ordenação);
- Lógica de combate e detecção de vulnerabilidades;
- Sistema de salvamento e carregamento;
- Interface e feedback visual.

Testes Educacionais

Testes com usuários reais para validar a eficácia pedagógica:

- **Participantes:** estudantes do curso de Análise e Desenvolvimento de Sistemas (conhecimento prévio de programação), entusiastas de jogos (sem conhecimento específico de estruturas de dados) e pessoas leigas (sem conhecimento técnico);
- **Instrumentos:** observação direta, questionários pós-jogo, análise de métricas de desempenho (tempo, tentativas, acertos);
- **Focos de Avaliação:** compreensão implícita dos conceitos, engajamento, diversão, clareza das mecânicas, dificuldade progressiva.

Resultados Preliminares

Os testes iniciais indicaram:

- Boa compreensão dos mecanismos de estruturas de dados mesmo entre participantes sem conhecimento prévio;
- Engajamento sustentado com a temática alquímica e narrativa do *plague doctor*;
- Feedback visual e mecânicas intuitivas facilitando a adoção;
- Necessidade de ajustes na dificuldade de certas fases e clareza de alguns consumíveis.

4.1.4 Fase 4 - Execução da Unidade Instrucional

Na quarta fase, o jogo será integrado em contextos educacionais reais, sendo utilizado em sala de aula como ferramenta complementar de ensino.

Implementação em Contexto Educacional

O jogo será disponibilizado como atividade complementar em disciplinas de Estruturas de Dados de cursos de Computação. Os docentes poderão:

- Integrar o jogo como revisão ou pré-aprendizado de conceitos;
- Observar o desempenho dos alunos através das métricas de progresso registradas pelo jogo;
- Utilizar os resultados como indicadores de compreensão e dificuldades específicas;
- Orientar discussões em sala sobre os conceitos descobertos implicitamente durante o jogo.

Coleta de Dados

Durante a execução, serão coletados:

- Métricas de jogo: tempo de conclusão, tentativas, acertos de combinação;
- Feedback dos participantes: satisfação, percepção de aprendizado, dificuldades encontradas;
- Observações qualitativas: comportamento durante o jogo, discussões com pares, reações emocionais.

4.1.5 Fase 5 - Avaliação da Unidade Instrucional

A avaliação final mede a efetividade global do jogo como ferramenta educacional.

Metodologia de Avaliação

Os dados coletados serão analisados de acordo com critérios baseados no MEEGA+:

- **Usabilidade:** clareza da interface, facilidade de aprender, ausência de bugs que interfiram na experiência;

- **Engajamento:** engajamento emocional, diversão, imersão, motivação intrínseca;
- **Aprendizagem Percebida:** percepção do jogador sobre aprendizado obtido, aplicabilidade dos conceitos, confiança em usar estruturas de dados;
- **Experiência Geral:** satisfação, disposição para recomendação, intenção de uso futuro.

Análise de Resultados

Os resultados serão analisados através de:

- **Análise Quantitativa:** estatísticas descritivas das métricas de desempenho, correlações entre tempo de jogo e compreensão de conceitos;
- **Análise Qualitativa:** análise temática de respostas abertas, observações de comportamento, entrevistas;
- **Triangulação:** cruzamento de dados quantitativos e qualitativos para validação de resultados.

Os resultados serão cotejados com os objetivos de aprendizagem definidos na Fase 1, verificando se o jogo contribuiu significativamente para o processo de ensino e aprendizagem de estruturas de dados.

Iterações e Melhorias

Com base nos resultados da avaliação, serão identificadas oportunidades de melhoria:

- Ajustes nas mecânicas e balanceamento de dificuldade;
- Refinamentos na narrativa e imersão;
- Otimizações de performance e acessibilidade;
- Expansões futuras (novos tipos de estruturas, fases adicionais, modos de jogo).

4.2 SÍNTESE DA METODOLOGIA

A adoção da metodologia ENgAGED garante que o desenvolvimento do jogo mantenha coerência sistemática, integrando continuamente princípios de design educacional com design de jogos. Ao estruturar o processo em fases bem definidas, com artefatos claros e objetivos específicos em cada etapa, a metodologia assegura que:

- O conteúdo educacional está solidamente fundamentado em objetivos de aprendizagem bem definidos;
- As mecânicas do jogo refletem os conceitos de estruturas de dados de forma implícita, promovendo aprendizado significativo;
- A experiência lúdica é priorizada, mantendo o jogo divertido e engajador;
- A avaliação é contínua e sistemática, fornecendo retroalimentação para iterações futuras;
- O resultado final é uma ferramenta educacional robusta, testada e validada para contribuir efetivamente ao processo de ensino e aprendizagem de estruturas de dados.

Dessa forma, o presente trabalho posiciona-se não apenas como um exercício acadêmico de desenvolvimento de software, mas como uma contribuição metodologicamente estruturada ao campo de educação em computação, especificamente na inovação de estratégias para tornar o ensino de estruturas de dados mais motivador, prático e efetivo.

5 TRABALHOS RELACIONADOS

Este capítulo apresenta uma revisão dos trabalhos existentes que combinam jogos sérios com o ensino de conceitos de programação. O conteúdo está organizado em duas partes: a primeira aborda pesquisas acadêmicas relacionadas ao tema, enquanto a segunda explora jogos já existentes com propostas semelhantes. Ao final, é apresentada uma síntese geral dos principais aspectos identificados nos trabalhos analisados.

5.1 ARTIGOS

Foi realizada uma pesquisa na plataforma *Google Scholar* com o objetivo de identificar trabalhos correlatos desenvolvidos nos últimos cinco anos (2021 a 2025). Utilizaram-se as palavras-chave "Estrutura de Dados", "Jogo Sério" e "Desenvolvimento", o que resultou inicialmente em um total de 45 artigos.

Após uma análise preliminar, 40 artigos foram descartados por não envolverem o desenvolvimento de um jogo ou por tratarem de jogos cuja temática não estava relacionada à área de programação. Com isso, restaram 5 artigos para uma avaliação mais aprofundada.

Os artigos selecionados para essa etapa de análise detalhada foram aqueles que atenderam simultaneamente aos seguintes critérios: aplicação de jogo sério no ensino de programação e apresentação de um protótipo funcional ou em desenvolvimento. Sendo estes:

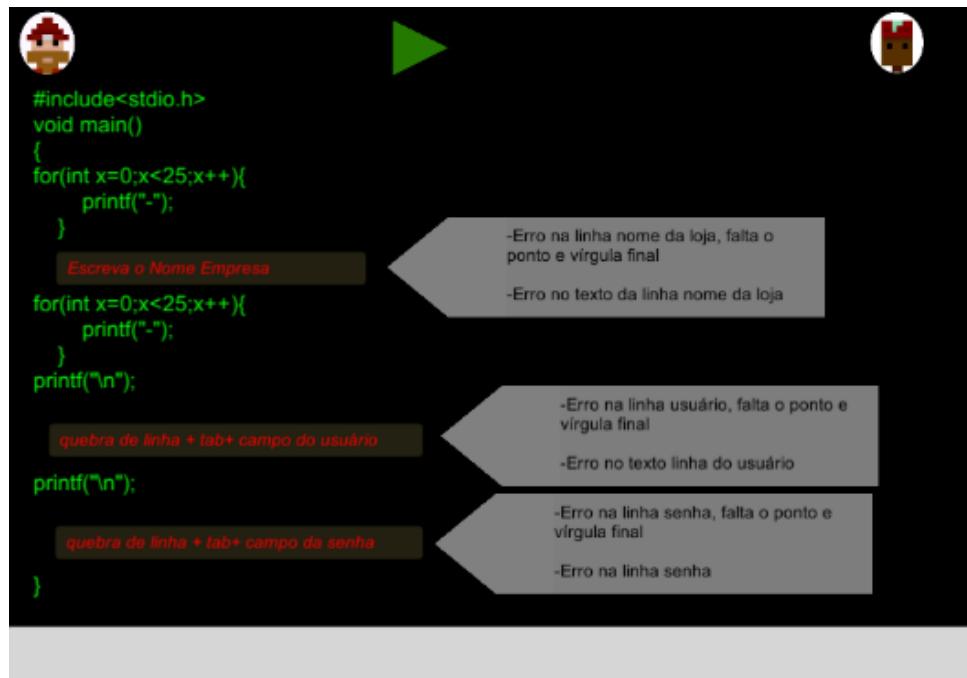
1. CodingJob: Um jogo sério para auxiliar nas disciplinas de Linguagem de Programação C

A dissertação CodingJob ([COSTA et al., 2023](#)) apresenta um jogo sério que fornece um ambiente para prática dos conhecimentos da disciplina de Linguagem de Programação I utilizando a linguagem C, não abrangendo nenhum conceito de Estruturas de Dados.

Este é um jogo *puzzle* que apresenta desafios de programação, simulando um ambiente de trabalho, o objetivo do jogador é arrumar as seções de códigos necessárias. Por se tratar de um simulador, o conteúdo didático deste é ensinado de forma explícita.

Por fim, identificou-se que o projeto analisado apresentou boa aceitação por parte dos alunos participantes. O estudo também revelou que a narração exerce um impacto mais significativo do que o inicialmente previsto na motivação dos jogadores. Dessa forma, a construção de um enredo mais envolvente se mostra uma característica fundamental a ser considerada em futuras melhorias, com o objetivo de aumentar o engajamento dos usuários ([COSTA et al., 2023](#)).

Figura 1 – Captura de tela do jogo CodingJob



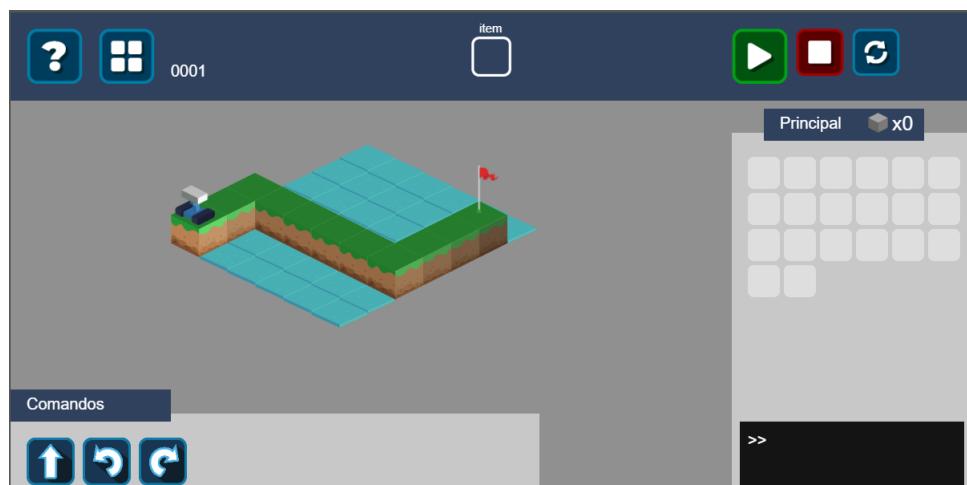
Fonte: (COSTA et al., 2023)

2. CodeBô: Design e avaliação de um puzzle game para o ensino de Estrutura de Dados

O intuito do projeto CodeBô (ARAUJO; SILVA, 2025) foi desenvolver um jogo digital isométrico de *puzzles* que se baseia na mecânica do *lightBot*, um jogo mobile educacional conceituado, mecânica esta que consiste em selecionar uma ordem de movimentos que o personagem deve executar para se mover até um local específico.

O jogo CodeBô utiliza essas mecânicas para ensinar conceitos como Pensamento Computacional, pilhas, filas e listas. (ARAUJO; SILVA, 2025)

Figura 2 – Captura de tela do jogo CodeBô

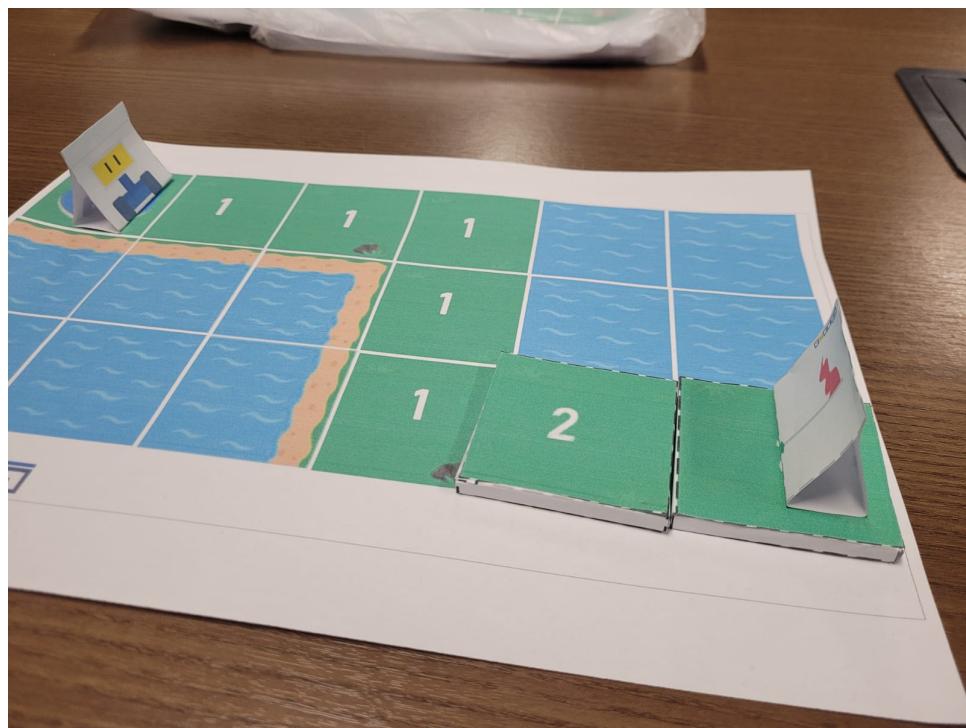


Fonte: (ARAUJO; SILVA, 2025)

3. CodeBo Unplugged: Um jogo desplugado para o ensino de Pilha

O jogo de tabuleiro CodeBo Unplugged ([CERQUEIRA; SILVA; ARAUJO, 2023](#)) foi desenvolvido com o intuito de ensinar a estrutura de dados Pilha a alunos do ensino fundamental de forma lúdica, utilizando elementos como robôs e mapas que aumentam em dificuldade de forma progressiva. ([CERQUEIRA; SILVA; ARAUJO, 2023](#))

Figura 3 – Foto do tabuleiro de CodeBô Unplugged



Fonte: ([CERQUEIRA; SILVA; ARAUJO, 2023](#))

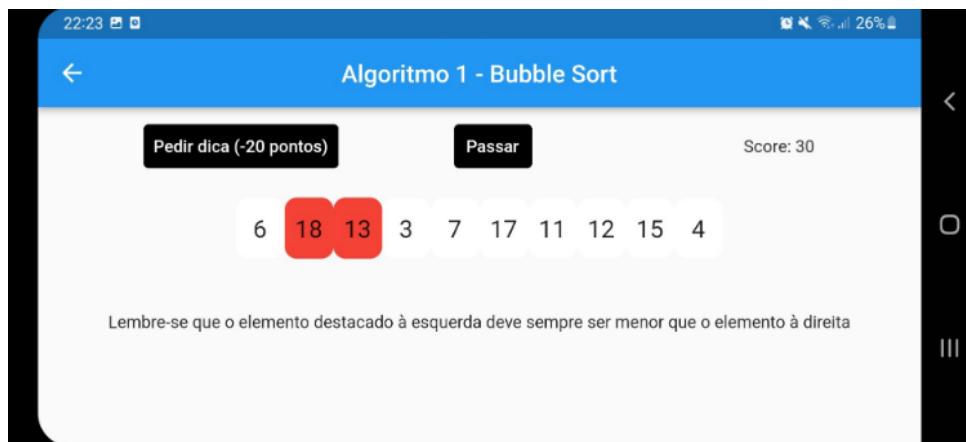
4. Desenvolvimento de um Jogo para Auxílio no Ensino de Estruturas de Dados

Durante o trabalho de conclusão de curso intitulado "Desenvolvimento de um Jogo para Auxílio no Ensino de Estruturas de Dados", foi desenvolvido um jogo digital mobile com o intuito de facilitar e auxiliar o ensino, a aprendizagem e a visualização dos conceitos de algoritmos de busca da disciplina de Estrutura de Dados.

A tecnologia utilizada para desenvolver este jogo foi a linguagem de programação Dart, em conjunto com o framework Flutter.

Por se tratar de um *quiz*, a abordagem de ensino é explícita. ([GLATZ et al., 2023](#))

Figura 4 – Captura de tela do jogo para auxiliar em estrutura de dados



Fonte: ([GLATZ et al., 2023](#))

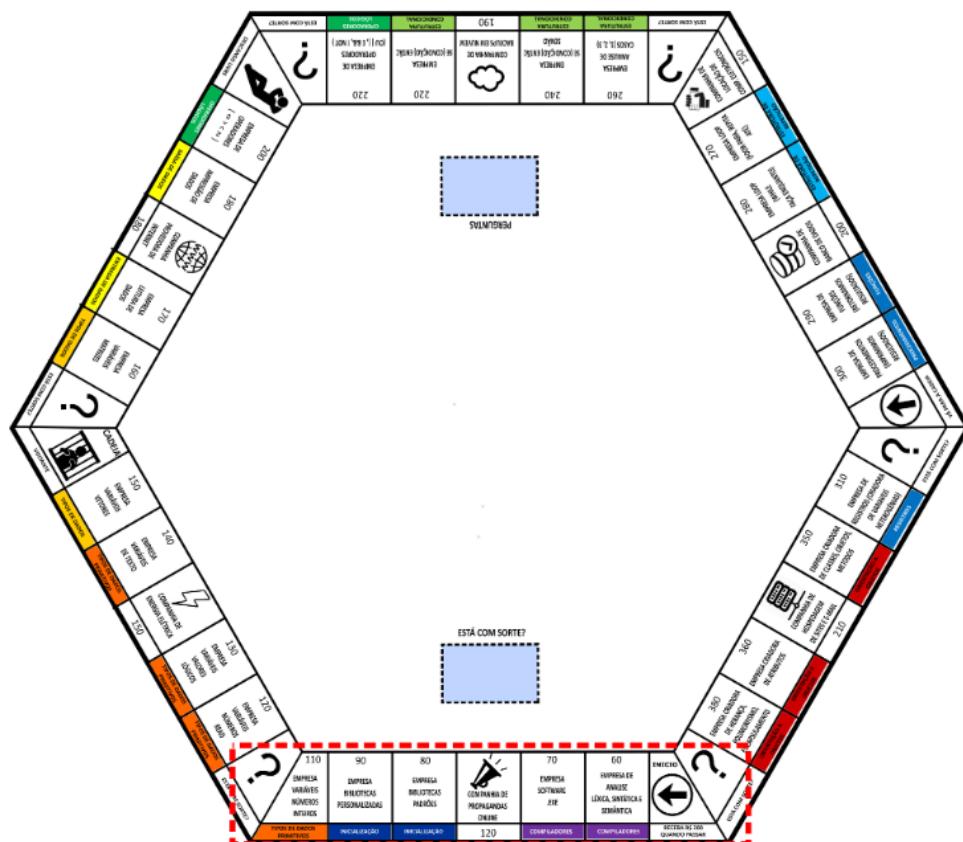
5. Prog-poly: Jogo de tabuleiro baseado no monopoly para ajudar nos estudos de linguagem de programação e engenharia de software

Durante o trabalho de pesquisa de mestrado Prog-poly ([NASCIMENTO et al., 2022](#)), foi desenvolvido um jogo de tabuleiro com o intuito de facilitar e auxiliar a aprendizagem de temas como linguagem de programação e engenharia de software.

Este jogo foi baseado na mecânica do clássico jogo de tabuleiro Monopoly, onde cada jogador deve comprar propriedades no tabuleiro. Entretanto, este jogo se diferencia pois para ter a oportunidade de comprar a propriedade, o jogador deve responder de forma correta perguntas a respeito de ILPC (Introdução Linguagem de Programação C) e, somente se acertar, poderá adquirir a propriedade, caso possua dinheiro suficiente. Ganhador o jogador que possuir a maior quantidade de dinheiro e propriedades.

Este Jogo se caracteriza por ser um jogo com uma abordagem de ensino explícita, pois se trata de um *quiz*. ([NASCIMENTO et al., 2022](#))

Figura 5 – Captura de tela do tabuleiro de Prog-poly



Fonte: (NASCIMENTO et al., 2022)

Por fim, foi efetuada uma comparação entre os trabalhos correlatos e este trabalho, demonstrada na [Tabela 2](#).

5.1.1 Comparação dos artigos

Com base na análise dos cinco trabalhos selecionados, observa-se uma variedade de abordagens no uso de jogos sérios para o ensino de conceitos de programação. Cada proposta apresenta escolhas distintas quanto aos conceitos abordados, estilo visual, mecânicas de interação e a forma como os conteúdos educacionais são apresentados ao jogador.

A [Tabela 2](#) a seguir apresenta uma síntese comparativa dos trabalhos analisados, destacando os aspectos centrais de cada proposta e os elementos recorrentes identificados entre eles.

Os critérios utilizados para a comparação incluem:

- **Jogo Digital (JD)** - Indica se o jogo é digital ou físico.
- **Conceitos de Estrutura de Dados (CED)** - Quais conceitos foram abordados:

- Pilha (P)
 - Fila (F)
 - Lista (L)
 - Lista Encadeada (LE)
 - Lista Duplamente Encadeada (LDE)
 - Árvore Binária (AB)
 - Algoritmo de Busca (B)
 - Algoritmo de Ordenação (O)
- **Forma de Abordagem Educacional (Ensino)** - Define se o ensino é tratado de forma explícita ou implícita.
 - **Estilo** - Representação visual do jogo.
 - **Gênero** - Categoria do jogo.

Tabela 1 – Comparação entre os trabalhos relacionados

Trabalho	JD	CED	Ensino	Estilo	Gênero
CodeBo Unplugged	Sim	-	Explícito	Simulador	Puzzle
	Sim	F,L,P,B	Implícito	Isométrico	Puzzle
	Não	P	Implícito	Tabuleiro	Puzzle
	Sim	B	Explícito	P&C	Puzzle
	Não	-	Explícito	Tabuleiro	Quiz

Fonte: Autor

5.2 APLICATIVOS

Em uma busca realizada nas plataformas *Play Store* e *App Store*, para aplicativos móveis, e nas plataformas *Steam* e *Itch.io*, para jogos digitais, foram identificados jogos correlatos. Utilizaram-se as palavras-chave "Programação", "Estrutura de Dados" e "Jogo Educacional", resultando inicialmente em um total de 262 jogos encontrados.

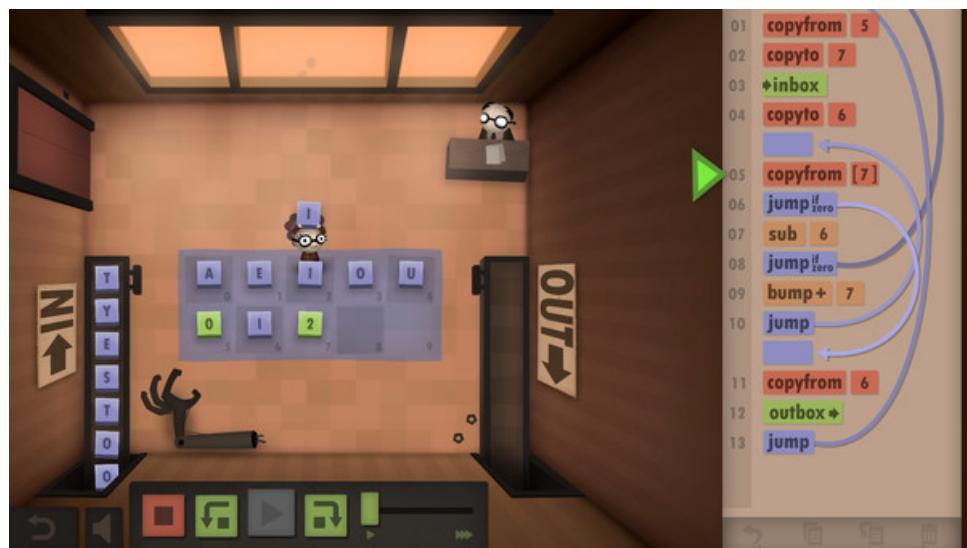
Após uma análise preliminar, 258 jogos foram descartados por não se enquadarem nos critérios da pesquisa. A maioria consistia em questionários ou aplicativos de ensino que não se configuravam como jogos, enquanto outros apresentavam temáticas não relacionadas à programação ou não atendiam aos requisitos mínimos de qualidade. Com isso, restaram 4 jogos para a etapa de avaliação mais detalhada.

1. Human Resource Machine

Jogo de *puzzle* desenvolvido pela *Tomorrow Corporation*, tem como intuito ensinar lógica de programação de forma lúdica utilizando uma ambientação no meio corporativo, onde em cada fase o seu chefe lhe dá um trabalho e o seu objetivo é automatizar este trabalho programando pequenos trabalhadores para tal tarefa, e se você tiver sucesso nesta tarefa, será "promovido" para a próxima fase.

Com uma narrativa motivadora e envolvente, o jogo *Human Resource Machine*, se caracteriza como um jogo com ensino explícito. ([Tomorrow Corporation, 2015](#))

Figura 6 – Captura de tela do jogo Human Resource Machine



Fonte: ([Tomorrow Corporation, 2015](#))

2. AlgoBot

Jogo de *puzzle* desenvolvido pela *Fishing Cactus*, tem como intuito ensinar algoritmos de forma lúdica utilizando uma ambientação futurística, onde o jogador terá o papel de operador que deverá criar uma sequência de comandos para o Algo Bot executar, seu objetivo é resolver puzzles para terminar a sua missão de reciclagem. ([Fishing Cactus, 2018](#))

Figura 7 – Captura de tela do jogo AlgoBot

Fonte: ([Fishing Cactus, 2018](#))

3. MOP’N SPARK

Jogo de *puzzle* desenvolvido pela *Omoplata Games*, tem como intuito ensinar algoritmos de forma lúdica utilizando uma ambientação fantasiosa, onde o jogador deverá criar uma sequencia de commandos para que os personagens Bepp e Gola alcancem os seus objetivos. ([Omoplata Games, 2025](#))

Figura 8 – Captura de tela do jogo MOP’N SPARK

Fonte: ([Omoplata Games, 2025](#))

4. Iron Ears: Data Structure

Iron Ears é um jogo do gênero *puzzle*, desenvolvido pela *NPC42 Games*, com o objetivo de ensinar conceitos de estruturas de dados de forma interativa. Ambientado em um

universo fictício habitado por animais antropomorfizados, com inteligência e racionalidade comparáveis às humanas.

O jogo coloca o jogador no papel de um coelho humanóide encarregado de projetar e operar linhas de montagem de *Mechs*. Esses robôs são essenciais na resistência contra uma facção hostil que busca dominar o mundo.

A mecânica do jogo integra diretamente estruturas de dados clássicas, como listas, filas e pilhas, aos sistemas de produção, exigindo que o jogador aplique esses conceitos para resolver desafios e otimizar os processos de fabricação. ([NPC42 Games, 2020](#))

Figura 9 – Captura de tela do jogo Iron Ears



Fonte: ([NPC42 Games, 2020](#))

No tópico a seguir, é apresentada uma comparação entre os aplicativos selecionados.

5.2.1 Comparação dos aplicativos

A análise dos jogos selecionados revela uma variedade de abordagens no uso de jogos sérios voltados ao ensino de lógica de programação. No entanto, observa-se uma presença ainda limitada de propostas que exploram conceitos de estruturas de dados de maneira mais aprofundada.

A [Tabela 3](#) apresenta uma síntese comparativa dessas iniciativas, destacando os principais elementos de cada proposta. Os critérios utilizados para esta comparação foram:

- **Conceitos de Estrutura de Dados (CED)** - Quais conceitos foram abordados:
 - Pilha (P)

- Fila (F)
- Lista (L)
- **Forma de Abordagem Educacional (Ensino)** - Define se o ensino é tratado de forma explícita ou implícita.
- **Estilo** - Representação visual do jogo.
- **Gênero** - Categoria do jogo
- **Gratuito** - Indica se o jogo está disponível gratuitamente.
- **Avaliação** - Nota atribuída com base no sistema de avaliação da plataforma onde o jogo é disponibilizado.

Nos casos em que o jogo está disponível em plataformas que não possuem sistema de avaliação integrado (como o *itch.io*), ou em que não há avaliações registradas, a avaliação é considerada *indefinida*.

Para os jogos disponíveis na Steam, foi utilizada uma conversão aproximada baseada na porcentagem de avaliações positivas, conforme mostrado na [Tabela 4](#).

Tabela 2 – Comparaçao entre os jogos relacionados

Trabalho	CED	Ensino	Estilo	Gênero	Gratuito	Avaliação
Human R.M.	-	Explícito	<i>Top Down</i>	Puzzle	Não	4.5
AlgoBot	-	Implícito	<i>Top Down</i>	Puzzle	Não	4.2
MOP’N SPARK	-	Implícito	Plataformer	Puzzle	Indefinido	Indefinido
Iron Ears	P,F,L	Implícito	<i>Drag & Drop</i>	Puzzle	Sim	Indefinido

Fonte: Autor

Tabela 3 – Conversão do Sistema de Avaliação da Steam para um sistema numeral de 1 a 5

Porcentagem de Avaliações Positivas	Avaliação Steam	Conversão Aproximada
90% - 100%	Extremamente positivas	4.5 - 5.0
70% - 89%	Muito positivas	3.5 - 4.4
40% - 69%	Neutras	2.0 - 3.4
10% - 39%	Muito Negativas	0.5 - 1.9
0% - 9%	Extremamente negativas	0.0 - 0.4

Fonte: Autor

O cálculo utilizado para a conversão é dado pela fórmula:

$$\text{Porcentagem de Avaliações Positivas} \times \frac{5}{100}$$

5.3 SÍNTES

Com base na análise dos trabalhos e jogos relacionados, observa-se que, embora existam diversas iniciativas que utilizam jogos para o ensino de conceitos de programação, a maioria adota abordagens mais explícitas e centradas em gêneros como *puzzle* e *quizzes*.

Além disso, os conceitos de estruturas de dados são raramente abordados e quando são, costumam ser apresentados de forma segmentada, focando em apenas um ou dois tipos, como pilhas ou filas.

6 DESENVOLVIMENTO

Este capítulo descreve o processo de desenvolvimento do jogo educacional seguindo a metodologia ENgAGED. O desenvolvimento foi estruturado em fases práticas, com foco no que foi efetivamente implementado durante este trabalho.

6.1 ARQUITETURA E ESTRUTURA DO PROJETO

O projeto foi desenvolvido utilizando o motor de jogo *Unity 2022 LTS* com linguagem de programação *C#*, garantindo compatibilidade e suporte a longo prazo. A estrutura do projeto segue padrões de organização profissionais, separando assets visuais, scripts, cenas, prefabs e recursos.

6.1.1 Estruturas de Dados Implementadas

As três estruturas de dados fundamentais foram implementadas como classes independentes em *C#*, refletindo com precisão os conceitos teóricos:

Pilha (Stack)

A implementação da pilha segue o padrão LIFO (Last In, First Out). As operações principais implementadas são:

- `Push(element)`: insere um elemento no topo da pilha;
- `Pop()`: remove e retorna o elemento do topo;
- `Peek()`: retorna o elemento do topo sem remover;
- `IsEmpty()`: verifica se a pilha está vazia.

Esta estrutura representa o inventário da pilha no jogo, onde elementos só podem ser removidos do topo, refletindo a ordem LIFO durante o combate.

Fila (Queue)

A implementação da fila segue o padrão FIFO (First In, First Out). As operações principais são:

- `Enqueue (element)` : insere um elemento no final da fila;
- `Dequeue ()` : remove e retorna o primeiro elemento;
- `Peek ()` : retorna o primeiro elemento sem remover;
- `IsEmpty ()` : verifica se a fila está vazia.

Este inventário representa a fila no jogo, onde elementos devem ser removidos na ordem em que foram adicionados, essencial para desafios que exigem sequência ordenada.

List (Linked List)

A implementação da lista encadeada permite acesso livre em qualquer posição. As operações principais são:

- `Add (element)` : adiciona um elemento ao final da lista;
- `Insert (index, element)` : insere um elemento em posição específica;
- `RemoveAt (index)` : remove o elemento na posição especificada;
- `Get (index)` : retorna o elemento na posição especificada;
- `Size ()` : retorna o tamanho da lista.

Este inventário oferece máxima flexibilidade, permitindo ao jogador acessar qualquer elemento independentemente de sua posição.

6.2 SISTEMAS PRINCIPAIS DO JOGO

6.2.1 Sistema de Movimento

O sistema de movimento implementa controles responsivos para o protagonista *plague doctor*. Através de entrada de teclado, o jogador pode:

- Movimentar-se horizontalmente (esquerda e direita);
- Pular com física realística de plataforma;
- Interagir com elementos do ambiente.

O sistema inclui detecção de colisão com plataformas, paredes e inimigos, garantindo movimento suave e previsível.

6.2.2 Sistema de Combate

O sistema de combate é central à mecânica educacional do jogo. Funciona através de:

1. **Seleção de Elementos:** o jogador remove elementos de seus inventários respeitando as regras de cada estrutura;
2. **Combinação:** elementos iguais são combinados para formar um ataque (ex.: dois elementos fogo geram ataque de fogo);
3. **Validação:** o sistema verifica se o elemento do ataque corresponde à fraqueza do inimigo;
4. **Aplicação de Dano:** ataques corretos causam dano ao inimigo; ataques incorretos causam dano ao jogador.

Este sistema reforça implicitamente o aprendizado das estruturas de dados, pois o sucesso depende diretamente de operações corretas.

6.2.3 Sistema de Inventário

Cada um dos três inventários é gerenciado independentemente, refletindo as características de sua estrutura:

- **Inventário de Pilha:** display visual mostra elementos em ordem LIFO, com remoção apenas do topo;
- **Inventário de Fila:** display visual mostra elementos em ordem FIFO, com remoção apenas do primeiro;
- **Inventário de Lista:** display visual permite visualizar todos os elementos com acesso livre a qualquer posição.

O sistema inclui geração contínua de elementos alquímicos (fogo, água, ar, terra) durante o jogo, preenchendo os inventários.

6.2.4 Sistema de Vida e Penalidades

O jogador inicia cada fase com quantidade determinada de vidas. Penalidades são aplicadas em casos de:

- Combinação incorreta de elementos (culpa didática);

- Contato com inimigos ou obstáculos;
- Queda fora dos limites do mapa.

Quando a vida atinge zero, o jogador retorna ao último checkpoint (ponto de controle), sem necessidade de reiniciar toda a fase.

6.2.5 Sistema de Progresso e Salvamento

O sistema gerencia:

- **Checkpoints:** posições de salvamento automático distribuídas estrategicamente;
- **Salvamento de Estado:** persistência de inventários, vida, fases desbloqueadas;
- **Métricas de Desempenho:** registro de tempo de conclusão, tentativas, acertos e erros de combinação.

Dados são salvos em arquivo local, permitindo continuação de partidas.

6.2.6 Sistema de Feedback

O jogo fornece feedback contínuo através de múltiplos canais:

Feedback Visual

- Animações do personagem indicando sucesso ou erro;
- Mudanças de cor (avermelhamento ao tomar dano);
- Efeitos particulares em ataques bem-sucedidos;
- Display visual dos elementos nos inventários.

Feedback Sonoro

- Sons de sucesso em combinações corretas;
- Sons de erro em combinações incorretas;
- Trilha sonora adaptativa ao contexto.

Feedback Textual

- Mensagens contextualizadas explicando erros;
- Indicadores de status (falta de mana, vida baixa);
- Descrições de inimigos e suas fraquezas.

6.3 INTERFACE DE USUÁRIO

6.3.1 Layout Principal do Jogo

A interface foi projetada com princípios de minimalismo, evitando sobrecarga visual. O layout principal inclui:

- **Área de Jogo:** centro da tela, ocupando a maior parte do espaço;
- **Inventários:** localizado na parte inferior direita, mostrando os três inventários lado a lado;
- **Barra de Vida:** indica a vida atual do jogador de forma clara;
- **Indicador de Inimigos:** mostra quantos inimigos restam na arena.

6.3.2 Menus Implementados

Menu Principal

Permite ao jogador:

- Iniciar novo jogo;
- Carregar jogo salvo;
- Acessar configurações;
- Sair do jogo.

Menu de Pausa

Disponível durante o jogo, permitindo:

- Retomar jogo;
- Acessar configurações;
- Voltar ao menu principal.

Telas de Livros

Livros informativos integrados, incluindo:

- **Livro de Tutorial:** explica mecânicas do jogo;
- **Livro de Inventários:** detalha cada estrutura de dados;
- **Livro de Combate:** documenta tipos de ataques e estratégias;
- **Livro de Consumíveis:** lista itens especiais disponíveis;
- **Bestiário:** informações sobre inimigos e fraquezas.

6.4 IMPLEMENTAÇÃO DE FASES

Cada fase foi implementada para enfatizar uma estrutura de dados específica:

6.4.1 Fase 1 - Pilha

Inimigos simples com uma única vulnerabilidade. Mecânica focada em:

- Compreensão do conceito LIFO;
- Remoção de elementos apenas do topo;
- Construção de confiança básica.

6.4.2 Fase 2 - Fila

Inimigos com padrões de movimento previsíveis. Mecânica focada em:

- Compreensão do conceito FIFO;
- Sequência ordenada obrigatória;
- Planejamento de ataques.

6.4.3 Fase 3 - Lista

Inimigos mais desafiadores com múltiplas fraquezas. Mecânica focada em:

- Flexibilidade de acesso à estrutura;

- Seleção estratégica de elementos;
- Adaptabilidade em combate.

6.4.4 Fase 4 - Consumíveis e Ordenação

Introduz itens especiais que modificam estruturas. Inclui:

- Item de Ordenação: reorganiza a estrutura de dados;
- Item de Inserção: adiciona elementos específicos;
- Item de Remoção: remove elementos específicos;
- Conceitos de algoritmos de ordenação aplicados implicitamente.

6.4.5 Fase 5 - Integração

Desafios finais combinando todas as estruturas. Requer:

- Domínio de todas as estruturas;
- Decisão de qual estrutura usar para cada ataque;
- Coordenação complexa de estratégias.

6.5 IMPLEMENTAÇÃO DE INIMIGOS

6.5.1 Tipos de Inimigos

Diversos tipos foram implementados, cada um ensinando conceitos específicos:

- **Inimigo Melee:** aproxima-se do jogador para atacar;
- **Inimigo Ranged:** atira projéteis à distância;
- **Inimigo com Múltiplas Fraquezas:** requer combinações mais complexas;
- **Inimigo Blindado:** requer ataques mais poderosos (combinações longas).

6.5.2 Sistema de Vulnerabilidades

Cada inimigo possui vulnerabilidades definidas (fogo, água, ar, ou terra). Ataques com elemento correto causam dano multiplicado; ataques com elemento incorreto causam dano ao jogador, reforçando a necessidade de planejamento.

6.5.3 Estatísticas Randomizadas

Para aumentar rejogabilidade, inimigos podem ter variações aleatórias em:

- Vida total;
- Velocidade de movimento;
- Cadênciа de ataque;
- Dano causado.

6.6 IMPLEMENTAÇÃO DE CONSUMÍVEIS

Seis tipos de consumíveis foram implementados:

6.6.1 Ordenação

Reorganiza a estrutura de dados ativa, servindo como auxílio estratégico quando elementos desejáveis estão em posições inacessíveis.

6.6.2 Insert

Adiciona dois elementos iguais ao inventário especificado, útil quando o jogador precisa de elementos específicos rapidamente.

6.6.3 Remove

Remove um elemento específico de um inventário, liberando espaço ou removendo obstáculos.

6.6.4 Mana

Concede mana infinita por tempo determinado, permitindo uso ilimitado de ataques durante o período.

6.6.5 Cura

Restaura 1 ponto de vida, fornecendo segunda chance quando vida está crítica.

6.6.6 Vida Extra

Aumenta a quantidade de tentativas em 1, estendendo a capacidade do jogador em uma fase.

6.7 DESENVOLVIMENTOS TÉCNICOS REALIZADOS

6.7.1 Persistência de Dados

Sistema de salvamento implementado utilizando serialização JSON, permitindo:

- Salvamento automático em checkpoints;
- Carregamento de partidas anterior;
- Registro de métricas de desempenho.

6.7.2 Gerenciamento de Câmera

Camera com pixel-perfect para manter qualidade artística do pixel art:

- Seguimento suave do personagem;
- Parallax visual com backgrounds;
- Limites de câmera para não revelar áreas não prontas.

6.7.3 Sistema de Diálogos

Sistema robusto de diálogos implementado, permitindo:

- Narrativa contextualizada;
- Interações com NPCs;
- Feedback narrativo sobre progresso.

6.7.4 Animações e Efeitos Visuais

Implementação completa de:

- Animações de movimento do personagem;

- Animações de ataque e dano;
- Efeitos particulares de elementos alquímicos;
- Transições e fade-outs.

6.8 DESAFIOS ENFRENTADOS E SOLUÇÕES

6.8.1 Balanceamento de Dificuldade

Desafio: Manter dificuldade progressiva sem frustrar o jogador.

Solução: Testes iterativos com diferentes públicos, ajustando vida de inimigos, velocidade e frequência de ataques baseado em feedback.

6.8.2 Clareza das Mecânicas

Desafio: Comunicar regras de estruturas de dados implicitamente.

Solução: Feedback visual claro e progressão cuidadosa, introduzindo um conceito por fase.

6.8.3 Performance

Desafio: Manter 60 FPS em máquinas variadas.

Solução: Otimização de sprites, redução de efeitos particulares em plataformas menos potentes, uso de object pooling para inimigos e projéteis.

6.8.4 Tratamento de Estados

Desafio: Gerenciar múltiplos estados do jogador (movimento, combate, diálogo, pausa).

Solução: Implementação de state machine para transições limpas entre estados.

6.9 TESTES REALIZADOS

6.9.1 Testes Funcionais

Validaram a corretude de todas as funcionalidades:

- Operações de estruturas funcionam conforme especificado;

- Lógica de combate resolve corretamente;
- Salvamento e carregamento preservam estado;
- Interface responde adequadamente.

6.9.2 Testes de Usabilidade

Realizados com público diverso:

- Estudantes do curso de ADS;
- Entusiastas de jogos;
- Pessoas sem experiência em programação.

Feedback indicou:

- Interface é intuitiva;
- Mecânicas são compreensíveis;
- Narrativa engaja os jogadores;
- Dificuldade é bem equilibrada.

6.9.3 Testes Educacionais

Avaliaram eficácia pedagógica:

- Compreensão implícita dos conceitos foi validada;
- Participantes conseguiram explicar comportamentos das estruturas;
- Interesse em estruturas de dados aumentou;
- Reconhecem aplicações práticas dos conceitos.

6.10 CONFORMIDADE COM REQUISITOS

O desenvolvimento seguiu rigorosamente os requisitos funcionais e não-funcionais definidos na metodologia, detalhados nos ?? e ?? (Apêndices A e B).

Todos os requisitos de alta prioridade foram implementados, grande maioria dos requisitos de média prioridade foi atendida, e alguns requisitos de baixa prioridade ficaram para futuras versões.

6.11 RESULTADOS DO DESENVOLVIMENTO

O desenvolvimento resultou em um jogo funcional, educacionalmente efetivo e visualmente coerente que:

- Ensina implicitamente conceitos de estruturas de dados através de mecânicas engajantes;
- Mantém coerência narrativa com temática de alquimia e plague doctor;
- Oferece experiência visual coerente em pixel art;
- Fornece feedback imediato e contextualizado para aprendizado;
- Permite progressão clara em dificuldade através das fases;
- Suporta reusabilidade de código em futuras extensões.

O jogo está pronto para integração em ambientes educacionais e validação de eficácia pedagógica junto a maior população de alunos.

7 CONSIDERAÇÕES FINAIS

REFERÊNCIAS

- ARAUJO, L. G. de J.; SILVA, A. P. dos S. Codebô: Design e avaliação de um puzzle game para o ensino de estrutura de dados. In: SBC. *Simpósio Brasileiro de Educação em Computação (EDUCOMP)*. [S.I.], 2025. p. 27–36. Citado 3 vezes nas páginas 11, 14 e 32.
- BATTISTELLA, P. E.; WANGENHEIM, C. G. von. *ENgAGED: Um Processo de Desenvolvimento de Jogos para Ensinar Computação*. Tese (Doutorado) — Universidade Federal de Santa Catarina, Florianópolis, SC, 2016. Disponível em: <https://www.researchgate.net/publication/309895607_ENgAGED_Um_Processo_de_Desenvolvimento_de_Jogos_para_Ensinar_Computacao>. Citado 3 vezes nas páginas 13, 18 e 19.
- CERQUEIRA, T. de O.; SILVA, A. P. S.; ARAUJO, L. G. de J. Codebo unplugged: Um jogo desplugado para o ensino de pilha. In: SBC. *Simpósio Brasileiro de Educação em Computação (EDUCOMP)*. [S.I.], 2023. p. 04–05. Citado na página 33.
- CORMEN, T. H. et al. *Introduction to algorithms*. MIT press, 2022. Accessed: 2025-07-04. Disponível em: <https://books.google.com.br/books?hl=en&lr=&id=RSMuEAAAQBAJ&oi=fnd&pg=PR13&ots=a311_W6FVM&sig=ws-06DHa7Xc06QXZKzh7Pb2fvsA&redir_esc=y#v=onepage&q&f=false>. Citado na página 14.
- COSTA, G. S. d. et al. Condigjob: um jogo sério para auxiliar nas disciplinas de linguagem de programação c. 2023. Citado 2 vezes nas páginas 31 e 32.
- CRESWELL, J. W.; CRESWELL, J. D. *Projeto de pesquisa:- Métodos qualitativo, quantitativo e misto*. Penso Editora, 2021. Disponível em: <https://books.google.com.br/books?hl=en&lr=&id=URclEAAAQBAJ&oi=fnd&pg=PT5&ots=9g5IISG0Hy&sig=xJnONf44ZCNqOeEsuCHHZ5A0lq8&redir_esc=y#v=onepage&q&f=false>. Citado na página 17.
- Fishing Cactus. *Algo Bot on Steam*. 2018. Accessed: 2025-04-14. Disponível em: <https://store.steampowered.com/app/286300/Algo_Bot/>. Citado 2 vezes nas páginas 37 e 38.
- GLATZ, I. et al. Desenvolvimento de um jogo para auxílio no ensino de estruturas de dados. Florianópolis, SC., 2023. Citado 2 vezes nas páginas 33 e 34.
- MALONE, T. W.; LEPPER, M. R. Making learning fun: A taxonomy of intrinsic motivations for learning. In: *Aptitude, learning, and instruction*. [S.I.]: Routledge, 2021. p. 223–254. Citado na página 15.
- MOUAHEB, H. et al. The serious game: what educational benefits? *Procedia-Social and Behavioral Sciences*, Elsevier, v. 46, p. 5502–5508, 2012. Citado 4 vezes nas páginas 11, 13, 14 e 17.
- MTAHO, A. B.; MSELLE, L. J. Difficulties in learning the data structures course: Literature review. *The Journal of Informatics*, v. 4, n. 1, p. 26–55, 2024. Accessed: 2025-07-04. Disponível em: <<https://www.ajol.info/index.php/tji/article/view/276299>>. Citado 4 vezes nas páginas 11, 13, 14 e 15.

NASCIMENTO, L. R. d. et al. *Prog-Poly: jogo de tabuleiro baseado no monopoly para ajudar nos estudos de linguagem de programação e engenharia de software*. Dissertação (Mestrado) — Universidade Tecnológica Federal do Paraná, 2022. Citado 2 vezes nas páginas 34 e 35.

NPC42 Games. *Iron Ears: Data Structure*. 2020. Accessed: 2025-04-15. Disponível em: <<https://npc42-games.itch.io/ironears>>. Citado na página 39.

Omoplata Games. *MOP'N SPARK on Steam*. 2025. Accessed: 2025-04-14. Disponível em: <https://store.steampowered.com/app/3491720/MOPN_SPARK/>. Citado na página 38.

PAPERT, S. The children's machine. *Technology Review-Manchester NH-, TECHNOLOGY REVIEW*, v. 96, p. 28–28, 1993. Citado 2 vezes nas páginas 11 e 15.

Tomorrow Corporation. *Human Resource Machine on Steam*. 2015. Accessed: 2025-04-14. Disponível em: <https://store.steampowered.com/app/375820/Human_Resource_Machine/>. Citado na página 37.