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

Luis Gustavo de Jesus Araujo[1], **Andreia Pinheiro dos Santos Silva**[2]

[1]IFBA – Instituto Federal de Educação
Ciência e Tecnologia da Bahia
44700-000 – Jacobina – BA – Brasil

[2]Prefeitura Municipal de Juazeiro
48903-495 – Juazeiro – BA – Brasil

luis_araujo@ifba.edu.br, andreiapinheiro.ifba@gmail.com

Abstract. *Programming courses have historically shown high rates of retention and dropout .Researchers seek to address this problem in different ways. Among these approaches, the use of educational games in the classroom stands out.Although there are many games focused on logic, algorithms, and computational thinking, there are limited games that cover data structure topics.This paper aims to present the development process of the code puzzle game CodeBô to aid in teaching data structures.The game was developed based on Flow Theory and the Zone of Proximal Development. To evaluate its effectiveness, a case study was conducted with players of various profiles. Evaluation using E-GameFlow showed positive results in terms of improved knowledge, concentration, goal clarity, feedback, challenge, and autonomy. The study includes a triangulation of logs and qualitative data to gain a deeper understanding of E-GameFlow-related issues.*

Resumo. *Disciplinas de programação apresentam, historicamente, altas taxas de reprovação e evasão. Pesquisadores buscam solucionar esse problema de diferentes maneiras. Dentre essas abordagens, destaca-se o uso de Jogos Educacionais em sala de aula. Embora existam muitos jogos sobre lógica, algoritmos e Pensamento Computacional, são limitados os jogos que abordam tópicos de Estrutura de Dados. Este artigo tem como objetivo apresentar o processo de desenvolvimento do jogo CodeBô, um puzzle code game, para auxiliar no ensino de Estrutura de Dados. O jogo desenvolvido foi baseado na Teoria do Flow e na Zona de Desenvolvimento Proximal. Para a avaliação, foi realizado um estudo de caso com jogadores de perfis variados. A avaliação, feita com o E-GameFlow, demonstrou resultados positivos quanto à melhoria do conhecimento, concentração, clareza nos objetivos, feedback, desafio e autonomia. O estudo apresenta ainda uma triangulação com logs e dados qualitativos para uma compreensão mais profunda das questões indicadas pelo E-GameFlow*

1. Introdução

Disciplinas da área de programação como Algoritmos e Estrutura de Dados historicamente apresentam alto índice de evasão e reprovação no ensino superior [Bennedsen and Caspersen 2007, Watson and Li 2014]. Pela característica complexa

do problema e fatores envolvidos neste cenário, pesquisadores buscam soluções diversificadas como novas abordagens pedagógicas, uso de mídias, *feedbacks* melhorados, robótica e jogos. Dentre as abordagens com jogos, destacam-se duas principais: a criação de jogos em sala de aula [Araujo et al. 2018] e a utilização de jogos educacionais [Araujo and Junior 2015]. A aprendizagem baseada em jogos digitais é algo já destacado na literatura [Prensky 2021]. No entanto, em pesquisa realizada [Gomes and Araujo 2020], constatou-se que existem poucos jogos sobre programação no idioma português e este número reduz significantemente quando focamos em jogos sobre Estruturas de Dados, *e.g.,* Pilha, Fila, Lista e Árvore. Outro fator identificado nesta pesquisa é que esses poucos jogos exploram apenas habilidades de níveis mais baixo, como lembrar e compreender, e não incorporam os conteúdos em suas mecânicas, limitando-se a utilizar os conceitos como tema do jogo [Gomes and Araujo 2020].

Com o intuito de abordar conceitos sobre Estrutura de Dados e possibilitar o desenvolvimento de habilidades de níveis mais altos como Aplicar, Avaliar ou Analisar [Ferraz and Belhot 2010] idealizou-se o jogo CodeBô. Este artigo tem como objetivo apresentar o processo de *design* do jogo CodeBô tendo como premissa a teoria do Fluxo, como modelo que promove a imersão do jogador, [Csikszentmihalyi and Csikzentmihaly 1990] e o conceito de Zona de Desenvolvimento Proximal, como o processo que mantém o jogador nesse estado [VYGOTSKY 1988]. Pretende-se, ainda, destacar aspectos da avaliação preliminar do jogo por meio do *framework E-GameFlow* e análises de logs dos jogadores [Fu et al. 2009].

Este artigo está organizado da seguinte maneira: na Seção II apresenta-se os principais conceitos sobre jogos educacionais para ensino de programação bem como os trabalhos relacionados. Na Seção III é descrito o processo de desenvolvimento do jogo CodeBô. Os elementos do jogo são destacados na Seção IV. A Seção V apresenta os participantes e o processo de avaliação. Os resultados são apresentados na Seção VI. As considerações finais são apresentadas na Seção VII.

2. Fundamentação e Trabalhos Relacionados

Reduzir as taxas de evasão e reprovação em disciplinas de programação em cursos de Computação ou cursos *Non-Major* é uma tarefa difícil e complexa. Pesquisas indicam que esse problema não é recente, podendo ser considerado um dos principais desafios da Educação em Computação [Bennedsen and Caspersen 2007, Watson and Li 2014]. Diante disso, diversas abordagens são idealizadas por pesquisadores, entre elas, o uso de jogos educacionais ou a criação de jogos em disciplinas [Araujo et al. 2018, Araujo and Junior 2015].

A criação de jogos educacionais visa utilizar os conceitos de programação na prática do desenvolvimento de jogos. Já o uso de jogos em sala de aula possibilita aos estudantes a interação com jogos educacionais. O processo de aprendizagem, no entanto, não se resume apenas a jogar; é necessário realizar contextualizações e discussões sobre o tema.

O trabalho de [Araujo and Junior 2015] apresenta o jogo Ghostbuster, que tem como objetivo o ensino de Programação Orientada a Objetos (POO). Esse jogo permite o uso de conceitos de POO por meio de blocos de comando que controlam a dinâmica do jogo. AlgoBot é um jogo em 3D que utiliza blocos de comandos em sequência para

controlar um robô que deve sair de um local e chegar a outro, com o objetivo de desenvolver o Pensamento Computacional [Higuchi et al. 2021]. Outros jogos que incorporam o conteúdo em sua mecânica de maneira similar são LightBot e SpriteBox, ambos da LightBot Inc., além de Cargo-Bot e Fur-Bot [Higuchi et al. 2021].

Embora existam muitos jogos sobre Algoritmos, o tema de Estruturas de Dados (ED) é raramente abordado. Em uma pesquisa realizada, que mapeou artigos em congressos brasileiros, foram encontrados apenas seis trabalhos que apresentam jogos voltados para ED. Dentre os temas abordados estão algoritmos de ordenação, árvores, listas, pilhas e filas. A maioria dos jogos utiliza mecânicas como quizzes, perguntas interativas e simulação de operações, sem integrar os conceitos à mecânica do jogo [Gomes and Araujo 2020]. Percebe-se que os desenvolvedores focam em habilidades de níveis mais baixos, como compreender e lembrar. Embora estimular essas habilidades seja importante, os jogos podem se tornar enfadonhos. Outro ponto a ser destacado é que os jogos não incorporam os conceitos em suas mecânicas. Esses fatores reduzem o engajamento dos estudantes e impedem a exploração do potencial da aprendizagem baseada em jogos.

Um grande desafio dos jogos educacionais é proporcionar ao jogador uma experiência imersiva e envolvente. Sem isso, o jogo educacional pode não ser eficaz no processo de aprendizagem. Deseja-se, portanto, que o jogador entre em um estado de concentração total, que favoreça a absorção do conhecimento, ou seja, permaneça no que Csikszentmihalyi chama de Flow (Fluxo). O papel do game design é justamente garantir que o jogador esteja sempre entre o desafio e o tédio, ou seja, que os desafios acompanhem o desenvolvimento do jogador [Rabin et al. 2011].

Da mesma forma, os jogos podem ser compreendidos como um processo de aprendizagem de suas próprias mecânicas. Assim, busca-se que a aprendizagem impulsione o desenvolvimento, permitindo que os jogadores consolidem os novos conhecimentos em processo de elaboração, apoiados por signos, interlocutores e instrumentos. Dessa maneira, ao jogar, os estudantes permanecem na Zona de Desenvolvimento Proximal, que se caracteriza pela distância entre o nível de desenvolvimento real, no qual o jogador é capaz de operar sozinho, e o nível de desenvolvimento potencial, determinado pela solução de problemas sob orientação [VYGOTSKY 1988]. Nesse sentido, as teorias do Fluxo e da Zona de Desenvolvimento Proximal podem auxiliar o processo de design do jogo.

3. Desenvolvimento do CodeBô

O jogo CodeBô, um *puzzle code game*, é baseado na mecânica do LightBot, que consiste em selecionar comandos de movimento para levar o robô do local inicial até um local específico, em um cenário isométrico. O CodeBô possui seu próprio personagem, estética, narrativa e *levels*. No entanto, seu maior diferencial é a adição de conceitos de Estruturas de Dados. Ou seja, o CodeBô não requer apenas conhecimentos de algoritmos e Pensamento Computacional, mas também de pilhas, filas e listas.

Diferentemente dos jogos apresentados em [Gomes and Araujo 2020], o CodeBô utiliza as operações de inserção e remoção nas estruturas de Fila, Lista e Pilha como mecânicas do jogo. Esse fator possibilita que o jogador empilhe blocos para acessar um nível mais alto dentro do mapa ou crie novos caminhos utilizando filas e listas. Isso torna o CodeBô mais lúdico, um aspecto importante em jogos educacionais.

O processo de desenvolvimento do CodeBô pode ser dividido em pré-produção, produção e pós-produção, seguindo um modelo em espiral. Foram utilizadas ferramentas proprietárias, como o Illustrator e o Photoshop, para a criação das imagens. O jogo foi desenvolvido com tecnologia web, por meio do *framework* StarterJs.

Do ponto de vista pedagógico, é necessário adotar uma corrente teórica que guie as decisões do game design. Dentre essas teorias, destacam-se as teorias psicogenéticas. Nossa abordagem utiliza a teoria sociointeracionista, baseada no trabalho de [Araujo and Junior 2015].

O principal objetivo do nosso *design* é possibilitar a imersão e evitar frustrações, fundamentando-se nas teorias do Fluxo e da Zona de Desenvolvimento Proximal. Para auxiliar o processo de game design, algumas perguntas são respondidas, baseadas no trabalho de [Araujo and Junior 2015]:

- **Desenvolvimento Real:**
  DR1 - Como iniciar?
  DR2 - O que o jogador aprendeu até o momento?
  DR3 - Esse elemento apresenta de forma convincente a nova mecânica?
  DR4 - Esse desafio está aquém das habilidades do jogador?
- **Desenvolvimento Proximal:**
  DP1 - O jogador tem habilidade para vencer o desafio?
  DP2 - Como posso utilizar essa aprendizagem para gerar novas mecânicas/desafios?
  DP3 - Como posso apresentar uma nova mecânica/regra sem gerar frustração no jogador?
  DP4 - Tenho desafios que possibilitam ao jogador o uso das mecânicas já aprendidas?

Atualmente, o jogo conta com um mundo implementado, Pilha, disponível online[1]. O mundo Pilha possui 10 *levels*, que vão desde a introdução das mecânicas até situações mais complexas, exigindo lógica avançada, como capturar blocos do *level* em vez de apenas utilizar os blocos fornecidos.

4. O CodeBô

O design do CodeBô prevê a criação de três mundos: Pilha, Fila e Lista. Em cada mundo, é necessário utilizar mecânicas baseadas nessas estruturas para solucionar os *puzzles*. No mundo Fila, o jogador precisa enfileirar e desenfileirar blocos. No mundo Lista, o jogador deve controlar a inserção de blocos em diferentes locais. No mundo Pilha, o jogador precisa empilhar e desempilhar blocos. Além disso, nesse mundo, é possível empilhar e desempilhar o próprio robô. Essa mecânica permite que o robô suba em locais mais elevados ou desça dentro do *level*.

A construção do CodeBô privilegia o *feedback*. Por isso, foi inserido um console (Figura 1, a), semelhante ao das IDEs. As mensagens emitidas podem ser classificadas como erro (vermelho) ou alerta (amarelo).

Os botões de *play*, pause e reinício coordenam a execução dos comandos (Figura 1, b). O jogador possui um número limitado de blocos para a criação, que pode ser

[1]https://sites.google.com/view/codebo-game/codebo

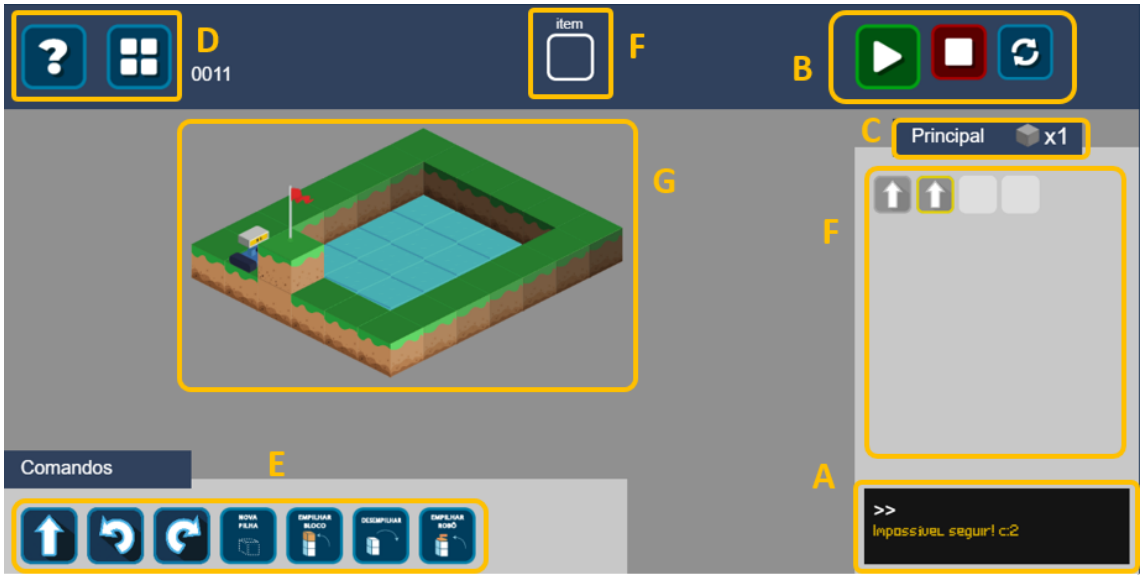

Figura 1. Level 3 - Mundo Pilha

visualizado na Figura 1, (c). O botão de ajuda exibe o *pop-up* inicial da fase, e o menu leva à seleção das fases (d).

Os comandos disponíveis para uso são apresentados na aba de comandos (e). Os comandos selecionados são visualizados e destacados quando executados (Figura 1, f). A execução dos comandos é simulada pelo robô no centro da tela (g).

No início de algumas fases, são exibidos *pop-ups* (Figura 2) com instruções e desafios que incentivam o uso das mecânicas do jogo. Por exemplo, na Fase 3, que é similar à Fase 2, o número de comandos disponíveis é reduzido, forçando o uso do comando de empilhar (Figura 1).

Além desses elementos, três mecânicas foram desenvolvidas exclusivamente para aprimorar a jogabilidade e tornar o jogo mais lúdico:

Ponte: constrói um caminho entre dois blocos separados por água. Espelho: reflete os blocos criados em outro local do mapa. Buraco Negro: transporta o robô para outro local do mapa. Cada um desses itens é apresentado em mundos diferentes.

O Mundo Pilha

Como já mencionado, o mundo Pilha possui 10 *levels*. Apresentamos a seguir cada *level*, destacando as questões norteadoras anteriormente discutidas para seu desenvolvimento.Level 1: São introduzidas, por meio de *pop-ups* (DR1, DR3), as mecânicas básicas de movimento (frente, esquerda e direita), bem como os botões da interface. O mapa contém curvas que possibilitam o uso de todos os comandos iniciais (DP1). Level 2: Não apresenta novas mecânicas (DR2), mas introduz uma regra: o robô só pode andar por blocos do seu nível de altura (DP2, DP3). Level 3: Utiliza o mesmo mapa do Level 2 (DP2), porém com uma quantidade reduzida de comandos disponíveis (DP3). Além disso, apresenta novos comandos por meio de *pop-ups*: criar pilha, empilhar blocos, empilhar o robô e desempilhar (DR3). Com isso, o jogador aprende a utilizar pilhas para subir níveis no mapa, gerando um novo desafio (DR4, DP1). Level 4: Não apresenta novas mecânicas

Figura 2. Level 6 - Pop-up inicial

(DP4).

Level 5: Introduz uma nova regra: a direção do CodeBô ao desempilhá-lo (DR2, DP1, DP2, DP3). Level 6: Apresenta, via *pop-up* (DR3), uma mecânica que estava subentendida até então: ao desempilhar blocos, o jogador ganha um bloco extra para usar posteriormente (DR2, DP1, DP2, DP3). Level 7: Reaproveita a mecânica do Level 6, utilizando o mesmo mapa do Level 4 (DP4), mas com uma quantidade inicial reduzida de blocos. Isso força o jogador a recuperar os blocos que utilizou (DR2, DR4, DP1, DP4). Level 8: Não apresenta novas mecânicas, apenas um mapa mais complexo (DR4). Level 9: Introduz a captura e uso do item Ponte, por meio de *pop-up* (DR3, DP2). Level 10: Não adiciona novas mecânicas, apenas apresenta um mapa mais desafiador.

5. Coleta de Dados e Análise

Com o objetivo de avaliar a adequação do jogo, testamos o primeiro mundo com jogadores reais. Para a avaliação, os participantes foram convidados, por meio das redes sociais dos autores, a participar do estudo. A amostra foi selecionada por conveniência, ou seja, os participantes foram escolhidos com base na acessibilidade e disponibilidade.

Para cada jogador, foi apresentado o Termo de Consentimento Livre e Esclarecido. Aos que aceitaram o convite, enviamos o link do jogo e um link para um formulário de avaliação. O formulário continha perguntas para coletar dados dos usuários (anonimizados), questões baseadas no *E-GameFlow* e perguntas abertas sobre conceitos de Estruturas de Dados. O *E-GameFlow* foi escolhido como *framework* de avaliação por utilizarmos a teoria do *Flow* como referência. A escala utilizada no *E-GameFlow* varia de 0 a 7.

Não informamos aos jogadores que se tratava de um jogo educacional sobre Estruturas de Dados. Além disso, solicitamos o envio de metadados coletados pelo navegador. Esses metadados registravam as ações dentro do jogo. O objetivo da coleta de metadados foi fornecer informações mais precisas sobre a experiência dos jogadores e complementar os resultados obtidos por meio do *E-GameFlow*.

Nossos resultados são divididos entre dados qualitativos e quantitativos. Por esse motivo, utilizamos o método de pesquisa misto [Creswell and Creswell 2021]. Nosso principal objetivo é aproveitar o potencial de cada um desses tipos de dados. Nossa metodologia permite triangular os dados e identificar aspectos específicos da avaliação do *E-GameFlow* por meio dos logs. A coleta e visualização dos logs foram baseadas no trabalho de [Horn et al. 2016].

5.1. Participantes

Ao todo, 14 avaliadores jogaram o CodeBô. Dentre eles, 11 eram homens e 5, mulheres. A média de idade dos participantes foi de 21 anos. Em relação à formação, 14% dos jogadores eram professores de Computação, 14% eram formados em cursos da área de Computação e 50% eram estudantes de Computação. Além disso, 21% dos jogadores eram estudantes do Ensino Fundamental. Quanto ao conhecimento prévio sobre Estruturas de Dados, 50% dos jogadores já haviam cursado ou estavam cursando disciplinas relacionadas ao tema, enquanto 28% já haviam ouvido falar sobre o assunto. Sobre o progresso no jogo, 57% dos jogadores chegaram até a última fase, enquanto 36% passaram da metade do jogo. Os resultados desta avaliação serão apresentados na próxima seção. Neste trabalho, mencionaremos os participantes pela letra E, seguida de um número identificador que varia de 1 a 14.

5.2. Resultados

A avaliação do *E-GameFlow* pode ser vista de modo geral com a média das categorias. A média obtida para a categoria Concentração foi 5,2, para Clareza foi 6,0, para *Feedback* 5,5, Desafio foi 5,5, a categoria Autonomia obteve média de 5,9, Imersão obteve média 4,7, Interação 4,2 e Conhecimento 6. Como é possível perceber, todas as médias foram maiores que 4, logo positivas. A média de Interação obteve o menor valor, o que já era esperado, pois o jogo não utilizou recursos para interação social entre os jogadores.

Com o intuito de entender melhor os aspectos da avaliação, analisamos as avaliações individuais e comparamos com a média. A Figura 3 apresenta a avaliação de alguns jogadores que representam de modo significativo a avaliação. A linha laranja representa a média geral, enquanto a linha azul indica a nota atribuída pelo jogador. Como é possível observar, o jogador E4 avaliou os itens Melhora do Conhecimento e Interação abaixo da média. O jogador E8 avaliou os itens Imersão e Melhora do Conhecimento abaixo da média. O jogador E12 avaliou a Clareza de Objetivos abaixo da média. Por fim, o jogador E13 avaliou o *Feedback* abaixo da média. O estudante E4 jogou apenas até a fase 6, não deixando comentários adicionais e nem percebendo o conteúdo envolvido no jogo. O jogador E8 não conhecia o conceito de Pilha, assim como o jogador E12. Este jogador ainda pontua um fato que pode ter influenciado sua avaliação sobre os objetivos: "*na fase 7, o personagem não conseguiu empilhar 2 blocos para subir o degrau e avançar de fase. Não sei se tem outra forma de passar, mas eu não encontrei rs*"(E12). O jogador E13 avaliou o *Feedback* abaixo da média. Este jogador afirma conhecer o conceito de Pilha, mas não profundamente. Este fator pode justificar a baixa avaliação em Melhoria do Conhecimento.

Todos os jogadores que já cursaram a disciplina de Estrutura de Dados mencionaram a presença do conceito de Pilhas no jogo. O estudante E5 pontuou: "*Sim, as pilhas. Funcionam basicamente como na programação, o último processo a entrar é o primeiro*

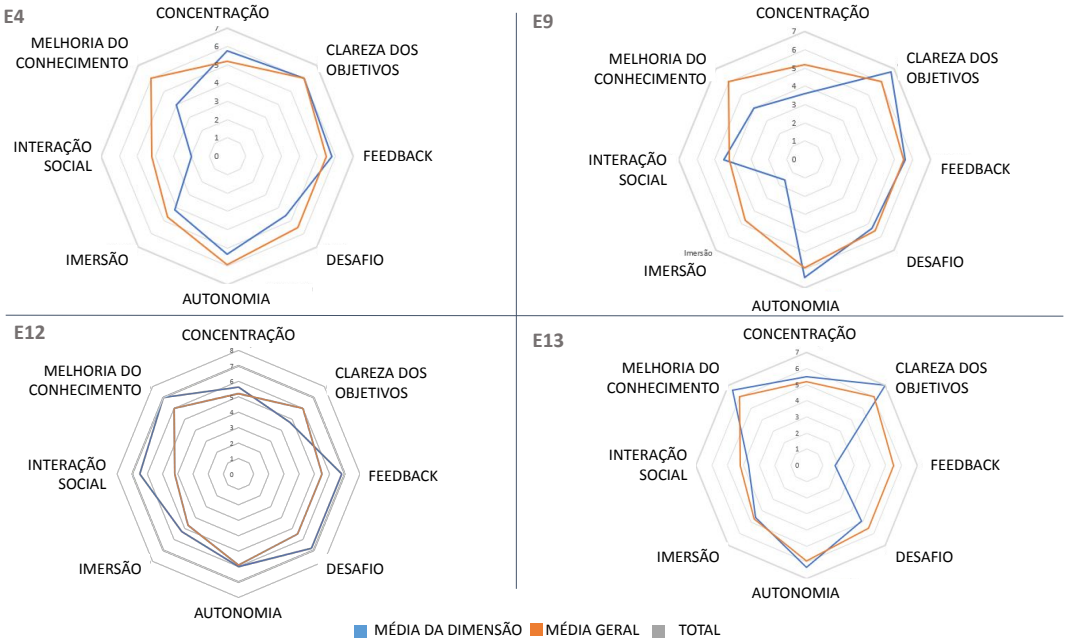

Figura 3. Avaliação E-GameFlow de alguns jogadores

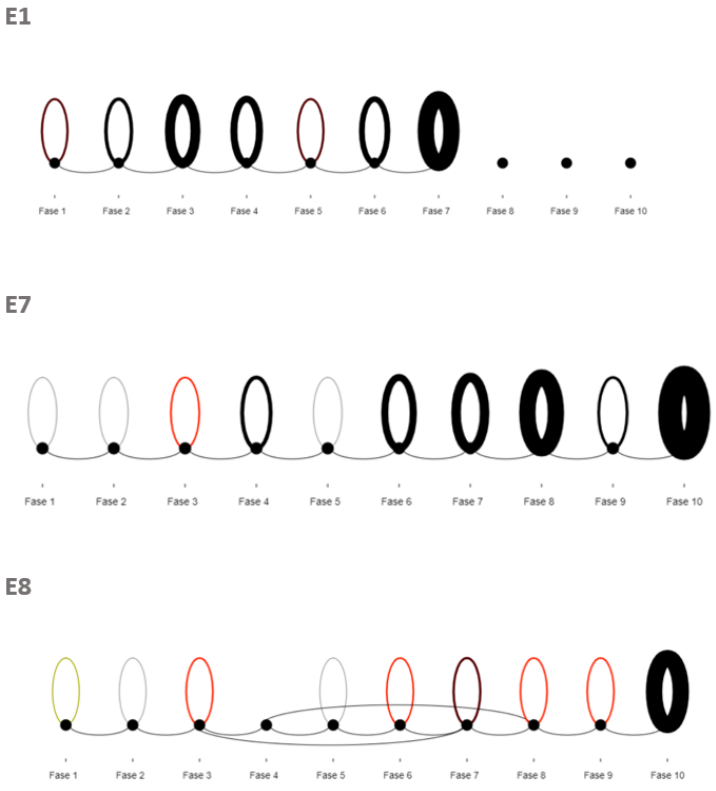

Figura 4. Visualização dos dados log de alguns jogadores

a sair"(E5). Já o estudante E4 afirmou: "*Sim, pilhas. As fases do jogo utilizam o conceito de pilhas: empilhar, desempilhar, etc.*"(E4).

O jogador E11 sinaliza sobre a ideia de usar a mecânica de Pilha, mas aponta um problema: "*O sistema de pilha no jogo é uma ideia interessante, porém confusa para o primeiro contato. Passei alguns minutos para entender como funciona a ideia de empilhar o robô e fazê-lo avançar. Somente os blocos de tutorial não foram suficientes. Talvez uma animação demonstrando como fazer tal procedimento resolveria esse problema. No mais, o jogo é muito bom, tem uma ideia legal, animação e design bem satisfatórios.*"(E11).

A Figura 4 apresenta os logs em formato visual. Linhas que conectam dois pontos indicam que o jogador passou de fase. Linhas que se conectam ao mesmo ponto sinalizam a repetição das fases. Quanto mais espessa e escura a linha, mais vezes o jogador repetiu o processo. O jogador E8 pontua: "*O cenário da última fase poderia ficar mais claro para o jogador.*"(E8). É possível perceber pelo seu log (Figura 4) que a última fase foi a mais repetida, embora tenha ocorrido repetições nas fases 1, 3, 6, 7, 8 e 9 também. Essa é uma fase com maior complexidade, por conter diversos caminhos e utilizar sobreposições nos blocos. Um fato interessante no comportamento deste jogador é o seu retorno para fases já superadas. Supõe-se que a volta tenha ocorrido devido à identificação de fases semelhantes, como já mencionado na Seção 4.

Contudo, é possível perceber que os jogadores não se afastam de modo considerável da média geral, sinalizando uma adequação do jogo. Quanto aos logs, a maioria dos jogadores precisou de mais de uma tentativa em cada fase. Em sua maioria, os jogadores jogaram fases subsequentes, sem retornar. Alguns jogadores, como o E1, interromperam o jogo após uma sequência significativa de repetições, enquanto outros, como o E7, continuaram até o final, mesmo com um número significativo de repetições.

6. Conclusão

Este artigo abordou o problema de evasão e reprovação em disciplinas de programação, dando foco em abordagens com jogos educacionais. Para tal, foi apresentado o processo de desenvolvimento do jogo CodeBô, seus elementos, mecânicas e fundamentos. O processo de avaliação do mundo Pilha foi descrito. Os resultados demonstram que o jogo possui uma boa avaliação, com necessidade de aprimoramento na interação social. Nota-se ainda que a maioria dos estudantes e profissionais consegue identificar o conceito de Pilha inserido na mecânica do jogo, demonstrando adequação do conceito e possibilidade de exploração pedagógica. Com isso, percebe-se que a utilização da teoria do Fluxo associada à Zona de Desenvolvimento Proximal é uma abordagem adequada ao desenvolvimento de um jogo educacional. O jogo se demonstrou mais adequado para estudantes que já conhecem o conceito de estrutura de dados. Por fim, o uso de logs forneceu informações relevantes quanto à complexidade de partes do jogo, que serão revistas no futuro.

6.1. Trabalhos Futuros

Como trabalhos futuros, destacamos os ajustes no jogo baseados na avaliação inicial, assim como o desenvolvimento dos mundos restantes. Será realizado uma nova avaliação visando analisar aspectos sobre aprendizagem de Estrutura de Dados mediante aplicação do jogo em sala. Por fim, serão disponibilizados, na página do projeto, planos de aula, materiais e atividades relacionadas à aplicação do jogo.

Referências

Araujo, L. G. and Junior, J. C. L. (2015). Estado de fluxo e zona de desenvolvimento proximal: A aprendizagem do jogador como elemento norteador do game designer. *Proceedings of Simpósio Brasileiro de Games e Entretenimento Digital (SBGAMES'15). SBC, Teresina, PI, BR. ISSN*, pages 2179–2259.

Araujo, L. G. J., Bittencourt, R. A., and Santos, D. M. B. (2018). Contextualized spiral learning of computer programming in brazilian vocational secondary education. In *2018 IEEE Frontiers in Education Conference (FIE)*, pages 1–9.

Bennedsen, J. and Caspersen, M. E. (2007). Failure rates in introductory programming. *AcM SIGcSE Bulletin*, 39(2):32–36.

Creswell, J. W. and Creswell, J. D. (2021). *Projeto de pesquisa-: Métodos qualitativo, quantitativo e misto*. Penso Editora.

Csikszentmihalyi, M. and Csikzentmihaly, M. (1990). *Flow: The psychology of optimal experience*, volume 1990. Harper & Row New York.

Ferraz, A. P. d. C. M. and Belhot, R. V. (2010). Taxonomia de bloom: revisão teórica e apresentação das adequações do instrumento para definição de objetivos instrucionais. *Gestão & produção*, 17:421–431.

Fu, F.-L., Su, R.-C., and Yu, S.-C. (2009). Egameflow: A scale to measure learners' enjoyment of e-learning games. *Computers & Education*, 52(1):101–112.

Gomes, L. S. G. and Araujo, L. G. d. J. (2020). AvaliaÇÃo e construÇÃo de jogos educacionais como suporte do processo de ensino-aprendizagem de estrutura de dados. *Anais da 17ª Jornada UNIFACS de Iniciação Científica*.

Higuchi, V., da Rocha, R. V., dos Santos Bezerra, D., and Goya, D. H. (2021). Algobot: jogo sério para o desenvolvimento do pensamento computacional. *Proceedings of Simpósio Brasileiro de Games e Entretenimento Digital (SBGAMES'20). SBC, Gramado, RS, BR. ISSN 2179-2259*.

Horn, B., Hoover, A. K., Barnes, J., Folajimi, Y., Smith, G., and Harteveld, C. (2016). Opening the black box of play: Strategy analysis of an educational game. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, pages 142–153.

Prensky, M. (2021). *Aprendizagem baseada em jogos digitais*. Editora Senac São Paulo.

Rabin, S. et al. (2011). Introdução ao desenvolvimento de games. *Cengage Learning, Sao Paulo, SP*.

VYGOTSKY, L. S. (1988). A formação social da mente. brasileira. *São Paulo, Martins*.

Watson, C. and Li, F. W. (2014). Failure rates in introductory programming revisited. In *Proceedings of the 2014 conference on Innovation & technology in computer science education*, pages 39–44.