

Desenvolvendo habilidades do Pensamento Computacional em Crianças: Uma Oficina de Criação de Jogos no Scratch

Carlos Augusto Ferreira Freire¹, Gilmar Rodrigues Campelo¹, Ulremberg Barbosa T. da Silva¹, Walquiria Pereira dos Santos¹, Marcos Paulo da Silva Santana³, Rodrigo Lins Rodrigues²

¹Departamento de Computação, ²Departamento de Educação, ³Programa de Pós-graduação em Ensino das Ciências
Universidade Federal Rural de Pernambuco (UFRPE)
CEP: 52171-900 – Recife – PE – Brasil

carlos.auugustof, gilmar.campelo, ulremberg.silva, walquiria.psantos,
rodrigo.linsrodrigues, marcos.pssantana@ufrpe.br

Abstract. *This paper describes an experience report of a game creation workshop using Scratch, aimed at developing computational thinking in elementary school children aged 10 to 12 years from both primary and middle school levels. The workshop used the Project-Based Learning (PBL) methodology and aimed to promote computational thinking, involving the pillars of decomposition, pattern recognition, abstraction, and algorithm creation. The intervention combined theoretical and practical activities, using Scratch and tools like Kahoot! to assess student progress. The results indicated a gradual improvement in the understanding of programming concepts, with notable student engagement and practical application of knowledge in game creation. The experience highlighted the importance of computational thinking in education, preparing students for the challenges of the digital world.*

Resumo. *Este artigo descreve um relato de experiência de uma oficina de criação de jogos no Scratch, voltada para o desenvolvimento do pensamento computacional em crianças do ensino fundamental I e II, com idades entre 10 e 12 anos. A oficina utilizou a metodologia de Aprendizagem Baseada em Projetos (ABP) e teve como objetivo promover o pensamento computacional, envolvendo os pilares de decomposição, reconhecimento de padrões, abstração e criação de algoritmos. Durante a intervenção, foram aplicadas atividades teóricas e práticas, com o uso do Scratch e ferramentas como o Kahoot! para avaliação do progresso dos alunos. Os resultados mostraram uma evolução gradual no entendimento dos conceitos de programação, com destaque para o engajamento dos alunos por meio da aplicação prática dos conhecimentos adquiridos na criação de jogos. A experiência reforçou a importância do pensamento computacional na educação, preparando os alunos para os desafios do mundo digital.*

1. Introdução

A informação e a comunicação desempenham papéis fundamentais no processo de transformação política, econômica e social da sociedade contemporânea. Nesse contexto, a educação formal precisa se adaptar para desenvolver habilidades cognitivas, sociais e profissionais essenciais para o século XXI (Gatti, 2021). A integração das Tecnologias de Informação e Comunicação (TIC) na educação é um passo crucial nessa direção, pois apresenta uma ampla gama de benefícios como facilitar o acesso à informação, melhorar a comunicação e a colaboração, aumentar a eficiência e

produtividade, possibilitar novas formas de aprendizagem e entretenimento, e promover a inovação e o desenvolvimento econômico (Dos Anjos et al, 2024).

Para o desenvolvimento de habilidades fundamentais para o século XXI, a integração das Tecnologias de Informação e Comunicação (TIC) na educação, não apenas facilita o acesso à informação e promove a inovação, mas também abre espaço para a formação de competências essenciais. Nesse contexto, o Pensamento Computacional (PC) (Wing, 2021) surge como uma abordagem crucial para a resolução de problemas de forma estruturada e lógica (Denning & Tedre, 2021). Envolvendo a decomposição de problemas, organização de dados, abstração, criação de algoritmos e generalização de soluções, o PC consolida-se como uma forma de pensar que transcende a ciência da computação, aplicando-se a diversas áreas do conhecimento e tornando-se uma competência transversal.

Considerando a importância do Pensamento Computacional como uma competência transversal, ferramentas como o Scratch, uma linguagem de programação visual e acessível, têm se destacado como recursos eficazes para introduzir conceitos de programação e desenvolver habilidades de forma lúdica e criativa. Este artigo relata uma experiência de ensino do pensamento computacional utilizando o Scratch, com ênfase no desenvolvimento de habilidades cognitivas e na resolução de problemas. A atividade foi realizada com alunos do ensino fundamental, empregando a metodologia de Aprendizagem Baseada em Projetos (ABP), e teve como objetivo principal promover o aprendizado significativo, alinhando-se às demandas educacionais da sociedade contemporânea.

2. Referências Teóricas

2.1 Pensamento Computacional

O pensamento computacional, conforme definido por Wing (2006), é um processo de raciocínio que envolve a formulação de problemas, a organização e análise de dados, a abstração, a criação de algoritmos e a generalização de soluções. Essa abordagem é fundamental para a resolução de problemas complexos e pode ser aplicada em diversas áreas do conhecimento, desde as ciências exatas até as humanidades. Wing (2021) destaca que o pensamento computacional não se restringe à programação, mas é uma habilidade transversal que pode ser desenvolvida em diferentes contextos educacionais.

Segundo Luo, Antonenko e Davis (2020), o pensamento computacional pode ser entendido como um conjunto de conceitos, práticas e perspectivas que permitem aos indivíduos resolver problemas de forma eficaz, utilizando métodos e ferramentas da ciência da computação. O pensamento computacional não se limita à programação, mas a programação é uma das formas mais comuns de aplicá-lo, especialmente em contextos educacionais. Os autores destacam a importância de atividades desplugadas (offline) e de feedback imediato para o desenvolvimento do pensamento computacional.

Na educação básica, o pensamento computacional promove o desenvolvimento de habilidades como raciocínio lógico, resolução de problemas e pensamento analítico, essenciais para a formação de cidadãos críticos e criativos (Batista, 2024). O autor ainda destaca que é uma habilidade transversal, aplicável em diversas áreas do conhecimento,

desde a matemática até as ciências humanas. A introdução desses conceitos desde cedo prepara os alunos para os desafios da sociedade digital, onde a capacidade de "aprender a aprender" é fundamental (WEF, 2020).

2.2 Aprendizagem Baseada em Projetos (ABP)

Aprendizagem Baseada em Projetos (ABP) é uma metodologia ativa que promove o aprendizado por meio da realização de projetos práticos e colaborativos. Segundo Bell (2010), a ABP estimula o desenvolvimento de habilidades do século XXI, como criatividade, trabalho em equipe, comunicação e resolução de problemas. Essa abordagem tem raízes no construtivismo, onde os alunos são protagonistas do seu próprio aprendizado, construindo conhecimento a partir de experiências significativas.

Moursund (1999) destaca que a ABP combina diferentes métodos de ensino, incluindo momentos de reflexão e instrução direta por parte do professor. Um estudo do American Institutes for Research (Zeiser et al., 2014) mostrou que alunos que participam de escolas com enfoques baseados em projetos alcançam melhores resultados em avaliações de conhecimento e habilidades interpessoais, além de terem maior probabilidade de ingressar no ensino superior.

No contexto do ensino de programação, a ABP permite que os alunos apliquem conceitos teóricos em projetos reais, como a criação de jogos e animações, promovendo um aprendizado mais profundo e significativo.

2.3 Desenvolvimento de Jogos

O desenvolvimento de jogos como estratégia educacional tem ganhado destaque por sua capacidade de engajar os alunos e promover o aprendizado de forma lúdica e interativa. Segundo Marques et al. (2011), a criação de jogos digitais estimula o raciocínio lógico, a criatividade e a colaboração, habilidades essenciais para o pensamento computacional.

A ABP é particularmente eficaz no desenvolvimento de jogos, pois permite que os alunos trabalhem em equipe, planejem, implementem e testem seus projetos, aplicando conceitos de programação de forma prática. Oliveira (2009) e Souza e Lencastre (2013) destacam que a criação de jogos no Scratch, por exemplo, facilita a compreensão de conceitos abstratos, como variáveis e condicionais, de maneira contextualizada e significativa.

2.4 Scratch

Scratch é uma linguagem de programação visual desenvolvida pelo MIT, projetada para ser acessível a iniciantes, especialmente crianças e jovens. Sua interface baseada em blocos permite que os usuários criem projetos interativos, como jogos, animações e histórias, sem a necessidade de dominar sintaxes complexas (Maloney et al., 2010). Essa característica torna o Scratch uma ferramenta ideal para o ensino de conceitos de programação e pensamento computacional de forma lúdica e criativa.

De acordo com Zaharija et al. (2013), o Scratch promove o desenvolvimento de habilidades como criatividade, colaboração e pensamento sistemático. A interface gráfica e os blocos de comandos organizados em categorias (como "movimento" e

"loops") facilitam a criação de programas sem a necessidade de memorização de códigos, tornando a programação acessível a pessoas sem experiência prévia.

O Scratch permite que os alunos desenvolvam projetos práticos, como jogos e animações, aplicando conceitos de programação de forma contextualizada. A criação de projetos no Scratch estimula a resolução de problemas, a colaboração e a criatividade, (Wang et al., 2014).

3. Metodologia

3.1 Campo e Participantes da oficina

A intervenção proposta teve como objetivo ensinar conceitos básicos de pensamento computacional e introduzir os alunos ao desenvolvimento de jogos utilizando a plataforma Scratch. A oficina foi realizada na Unidade de Tecnologia e Educação- Utec Nova Descoberta, localizada no Bairro de Vasco da Gama, Recife-PE, o público-alvo da oficina foi composto por crianças de 10 a 12 anos do ensino fundamental I e II, nove alunos ao todo sendo quatro meninas e cinco meninos, sem familiaridade com conceitos formais de computação e programação. Algumas dessas crianças apresentavam dificuldades no uso de periféricos de computador, como mouse e teclado, além de noções limitadas sobre o sistema operacional. Um dos alunos tinha Transtorno do Déficit de Atenção e Hiperatividade (TDAH), que exigiu uma atenção maior para mantê-lo engajado.

3.2 Planejamento do Curso

A oficina foi estruturada em 5 encontros, realizados às terças e quintas-feiras, das 08h às 11h da manhã, com uma duração de 3 horas por aula. Cada sessão combinou atividades teóricas e práticas, com foco na utilização do Scratch offline para o desenvolvimento de jogos simples. As aulas práticas permitiram que os alunos aplicassem os conceitos de programação de maneira lúdica e interativa no scratch, enquanto as aulas teóricas abordaram tópicos como sequência de comandos, loops, condições, variáveis, eventos e pilares do pensamento computacional.

O planejamento do curso foi estruturado em etapas, conforme ilustrado na Figura 1. A primeira etapa consistiu na ampliação dos conceitos de programação e atividades desplugadas, onde os alunos foram introduzidos a ideias fundamentais de pensamento computacional por meio de atividades que não requerem o uso de computadores. Em seguida, houve a apresentação dos blocos de programação no Scratch, onde os alunos tiveram a oportunidade de experimentar diretamente na ferramenta, familiarizando-se com a interface e os comandos básicos. Após essa introdução prática, os alunos receberam conceitos teóricos complementares, que foram consolidados por meio de questionários no Kahoot, uma plataforma que permitiu revisar e testar o entendimento dos contextos abordados em aula. O curso também incluiu a apresentação dos projetos autorais, onde os alunos foram incentivados a pensar

criativamente e a aplicar os conceitos aprendidos. Essa etapa foi seguida pela criação dos Projetos Autorais, onde os alunos desenvolveram seus próprios projetos no Scratch, com acompanhamento contínuo dos instrutores para garantir o entendimento e a aplicação correta dos conceitos. Por fim, os alunos apresentaram seus projetos, compartilhando suas criações com a turma e recebendo feedback.

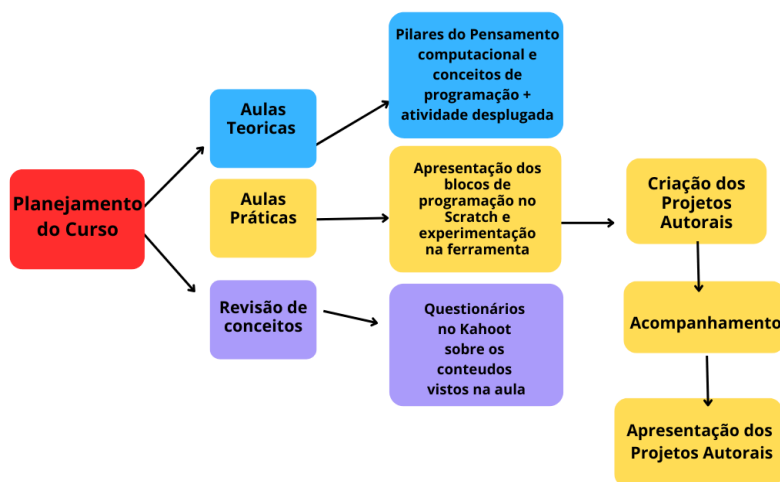


Figura 1- Estrutura do planejamento do curso

Além das atividades práticas, a oficina utilizou o Kahoot ao final de cada aula para reforçar o conteúdo aprendido, promovendo uma revisão dinâmica e engajante do que foi trabalhado. A estrutura do curso foi planejada de forma progressiva, partindo de conceitos fundamentais para aplicações mais complexas, seguindo os princípios da aprendizagem significativa (Ausubel, 1968). Esta abordagem gradual foi escolhida considerando o perfil dos participantes, crianças sem experiência prévia em programação, permitindo que construíssem conhecimento novo a partir de bases sólidas e experiências concretas antes de avançar para conceitos mais abstratos. A abordagem pedagógica combinou atividades desplugadas e práticas no Scratch. Considerando a pouca familiaridade dos alunos com pensamento computacional, foram utilizadas dinâmicas que relacionam conceitos abstratos a situações cotidianas. Um exemplo foi a elaboração de um algoritmo para a preparação de um alimento pré-preparado, permitindo a identificação de conceitos como decomposição e abstração.

3.3 Material Desenvolvido

Durante a oficina as crianças participaram de atividades desplugadas e dialogadas sobre pilares do pensamento computacional e alguns conceitos de programação. Essas atividades foram projetadas para tornar os conceitos abstratos mais tangíveis e relacionados ao cotidiano dos alunos. Por exemplo, na decomposição, os alunos discutiram o passo a passo de uma receita pré-pronta, aprendendo a dividir tarefas complexas em partes menores. Para reconhecimento de padrões, identificaram loops e

sequências em atividades em papel, enquanto na abstração, refletiram sobre o desenvolvimento de personagens e tarefas cotidianas, além de explorarem o conceito de variáveis e condicionais com exemplos práticos. Ao final da oficina, os alunos aplicaram esses conceitos no desenvolvimento de um jogo no Scratch, conforme ilustrado na Figura 2.

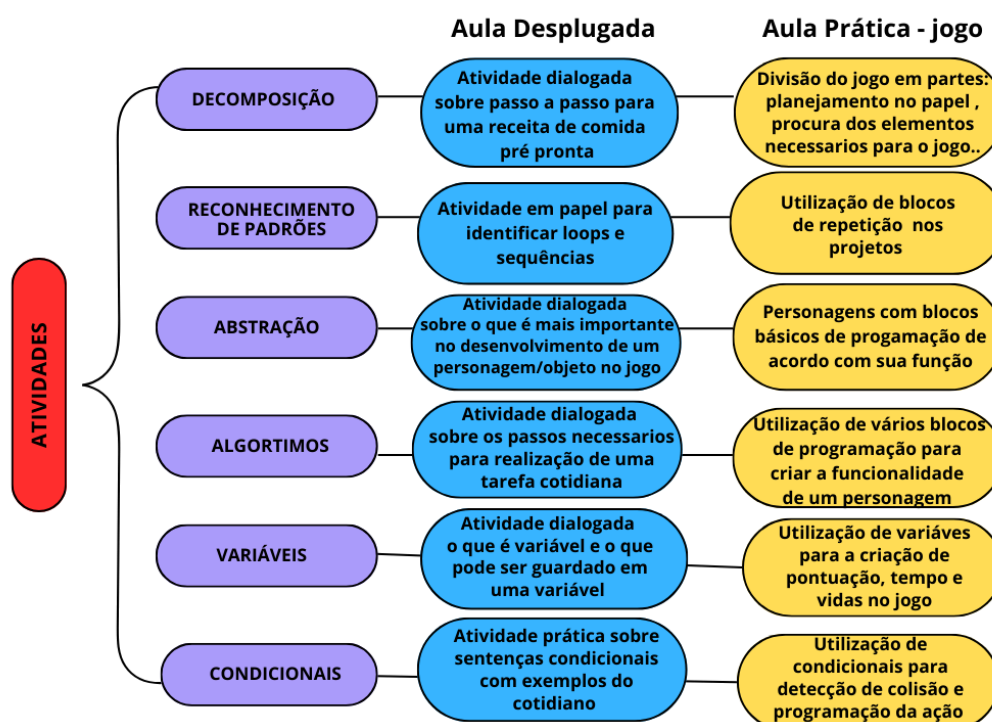


Figura 2 - Conceitos trabalhados nas aulas.

Eles começaram com o planejamento no papel, estruturando o jogo e identificando os elementos necessários. Em seguida, utilizaram blocos de repetição para criar loops e otimizar o código. Cada personagem foi programado com blocos básicos de acordo com sua função, e vários blocos foram combinados para criar as funcionalidades desejadas. Variáveis foram usadas para controlar elementos como pontuação, tempo e vida, enquanto condicionais foram aplicadas para detectar colisões e programar as ações correspondentes.

Trabalhamos os pilares do pensamento computacional de forma desplugada para aumentar a assimilação do conteúdo e prática para o desenvolvimento dos jogos, pois quando os conceitos já estão bem compreendidos, a transição para a programação digital se torna mais fluida, tornando a criação dos jogos e projetos computacionais mais intuitiva e eficiente. Esta abordagem é respaldada por estudos que demonstram que atividades desplugadas facilitam a compreensão de conceitos abstratos de computação (Luo, Antonenko e Davis, 2020), especialmente para iniciantes, permitindo que os

alunos desenvolvam o pensamento computacional sem as barreiras iniciais impostas pela sintaxe de programação.

4. Resultados e Discussão

A evolução do aprendizado dos alunos foi avaliada ao longo da oficina por meio de uma avaliação de conhecimentos feitos na primeira e última aula (resultados na Figura 3) e atividades no Kahoot. Foram aplicadas cinco avaliações, cada uma abordando os conceitos ensinados nas aulas.

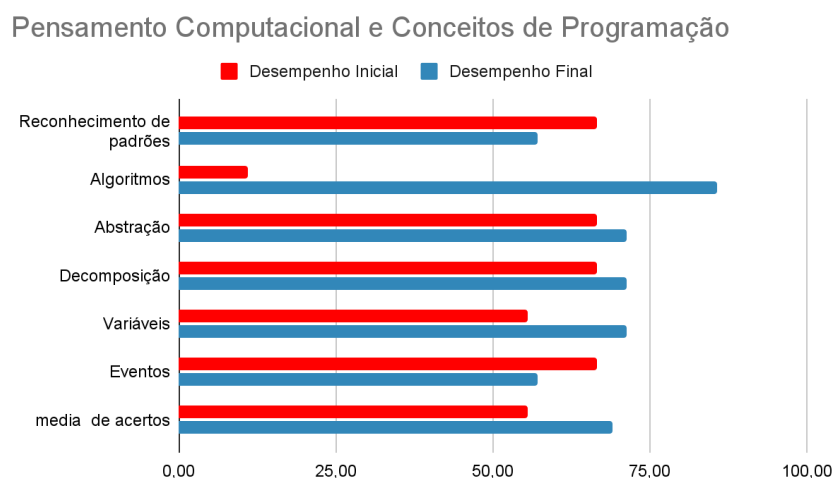


Figura 3. Desempenho inicial e final da Verificação de Conhecimentos.

Na Figura 3 temos um comparativo de desempenho dos alunos da primeira e última verificação de conhecimentos, na qual fizemos em papel para medir o grau de conhecimento sobre pensamento computacional e alguns conceitos de programação, a primeira verificação foi feita no primeiro dia da oficina e os alunos não tinham nenhum conhecimento prévio, a segunda verificação foi feita no último dia. O objetivo das verificações era avaliar o grau de compreensão dos conceitos apresentados na oficina, embora entre uma verificação e outra houve uma diferença de dois alunos que faltaram, é possível observar uma progressão na média de acertos, principalmente no que se refere a compreensão de algoritmo, isso denota que a oficina mesmo com a duração reduzida contribuiu para um primeiro contato com conceitos de pensamento computacional “o qual permite que as crianças se tornem criadoras de tecnologia, não apenas consumidoras.” (Seymour Papert, 1980).

4.1 Desempenho nas Avaliações

Ao longo das avaliações com Kahoot, observamos diferentes níveis de compreensão dos conceitos do Scratch. No primeiro teste, que abordou movimentação de personagens e sequência de comandos, os alunos tiveram um bom desempenho, demonstrando

facilidade com a interface e a lógica básica de blocos. Esse resultado reforça a acessibilidade do Scratch na introdução à programação (Ito et al., 2019).

Avançando para condicionais e interatividade, percebemos uma leve queda na média de acertos. Muitos alunos encontraram dificuldades com os blocos de controle, especialmente aqueles que envolviam "IF" e "ELSE". Essa complexidade é esperada, pois a lógica condicional exige um pensamento mais abstrato, o que representa um desafio comum no aprendizado de programação (Tang et al., 2020).

Durante o terceiro teste, focado no uso de variáveis para armazenar pontuação e vidas, foi revelado uma variação no desempenho. Enquanto alguns alunos assimilaram rapidamente o conceito, outros tiveram dificuldades em modificar valores dinamicamente. Para sanar essa questão, reforçamos o conteúdo com exemplos práticos relacionados ao cotidiano dos alunos.

Já na avaliação sobre estruturas do pensamento computacional, eventos e interações do Scratch, a média de acertos foi moderada. Os resultados mostraram que, embora os alunos conseguissem lidar com decomposição e eventos, o conceito de abstração ainda precisava ser mais trabalhado, pois nem sempre era aplicado corretamente na criação dos jogos. Esse aspecto segue sendo um ponto de atenção, conforme estudos recentes discutem formas de aprimorar a avaliação do pensamento computacional (Brennan & Resnick, 2020).

Por fim, a última avaliação integrou todos os conceitos em um jogo completo. O desempenho geral foi satisfatório, evidenciando progresso no uso de comandos e na estruturação lógica dos projetos. No entanto, alguns alunos ainda enfrentaram dificuldades ao adaptar os conceitos a algoritmos mais complexos, o que reforça a importância de práticas contínuas para o domínio da programação. A escolha do Kahoot como ferramenta de avaliação foi estratégica, pois sua natureza gamificada mantém o engajamento dos alunos enquanto fornece feedback imediato sobre o aprendizado. Essa abordagem alinha-se às pesquisas de Román-González et al. (2021), que destacam a importância de métodos avaliativos dinâmicos no ensino de programação, permitindo ajustes rápidos nas estratégias pedagógicas conforme as necessidades identificadas.

4.2 Observações Sobre o Aprendizado

Durante a oficina, foi possível perceber que a proposta de criação de jogos no Scratch aumentou significativamente o engajamento dos alunos e beneficiou seu desenvolvimento em várias dimensões. A natureza lúdica e interativa da atividade despertou o interesse dos estudantes, que se sentiram motivados a criar algo possível e divertido. Esse senso de propósito e realização foi importante para mantê-los envolvidos, especialmente em uma faixa etária em que o aprendizado por meio de jogos é naturalmente atraente (Madureira & Schneider, 2024). A gamificação das avaliações,

com o uso do Kahoot, reforçou ainda mais esse engajamento, transformando a revisão dos conceitos em uma experiência dinâmica e competitiva.

A criação de jogos permitiu que os alunos aplicassem conceitos de programação, como estruturas de controle, variáveis e condicionais, de forma prática e contextualizada. Essa aplicação direta ajudou a consolidar o aprendizado teórico, tornando o pensamento computacional mais tangível. Além disso, a necessidade de decompor problemas, reconhecer padrões e criar algoritmos para desenvolver os jogos estimulou habilidades essenciais de resolução de problemas. A atividade também incentivou a criatividade, pois os alunos projetaram personagens, cenários e mecânicas de jogo, e promoveu a colaboração, já que trabalharam em equipe, trocando ideias e soluções.

Ao longo das aulas, os alunos ganharam autonomia, passando a testar soluções por conta própria e a depender menos dos instrutores. Essa evolução foi um dos resultados mais significativos, demonstrando como a proposta de criação de jogos pode fortalecer a confiança e a independência dos estudantes. A conexão dos conceitos de programação com situações do cotidiano, como a preparação de uma receita, também facilitou a compreensão de ideias abstratas, como decomposição e abstração.

As avaliações progressivas com o Kahoot permitiram identificar dificuldades e ajustar as estratégias de ensino, reforçando tópicos como condicionais e variáveis com exemplos práticos. O uso de ferramentas de avaliação interativas, como o Kahoot, continua a ser relevante em ambientes de aprendizagem de programação, pois promove um feedback imediato e permite ajustes rápidos nas estratégias de ensino (Román-González et al., 2021). Essa abordagem gamificada não apenas aumentou o engajamento dos alunos, mas também contribuiu para uma avaliação mais dinâmica e eficaz do progresso no aprendizado.

Os resultados gerais foram positivos, mostrando uma evolução gradual no entendimento dos conceitos de programação e pensamento computacional. A abordagem gamificada e a criação de jogos se mostraram eficazes não apenas para aumentar o engajamento, mas também para promover um aprendizado significativo, preparando os alunos para os desafios do mundo digital de forma criativa e colaborativa.

4.3 Desenvolvimento de Projetos

Durante o desenvolvimento dos projetos tivemos desafios e aprendizados significativos. A primeira etapa foi a formação das equipes, que não ocorreu de maneira simples. Alguns alunos resistiram a mudar de grupo, outras não quiseram formar duplas com meninos e a frequência irregular de alguns alunos dificultou a organização. As dificuldades na formação de equipes, especialmente as questões de gênero, ainda são um desafio na área da computação (Grover et al., 2019). O suporte foi oferecido

conforme a necessidade de cada aluno, sem uma mentoria fixa. Esse modelo possibilitou um acompanhamento próximo e incentivou a colaboração entre os alunos. Durante o desenvolvimento, foi possível perceber que os códigos criados tinham pouca organização, o que indica a necessidade de reforçar boas práticas de programação em atividades futuras. Na apresentação final, no entanto, os alunos demonstraram segurança ao explicar seus jogos e responder às perguntas da gestora da unidade de ensino, mostrando que compreenderam os conceitos trabalhados na oficina. O desenvolvimento de projetos, mesmo com desafios, permanece uma estratégia pedagógica eficaz (Hmelo-Silver, 2020).

5. Considerações Finais

A oficina de criação de jogos no Scratch voltada para o ensino de pensamento computacional por meio da criação de jogos proporcionou uma experiência enriquecedora tanto para os alunos quanto para nós, os educadores. Durante o processo, foi possível observar como a programação visual facilita a compreensão de conceitos fundamentais da computação tornando-os acessíveis e intuitivos.

Um dos principais aprendizados foi a compreensão de como os conceitos do pensamento computacional, como decomposição, reconhecimento de padrões, abstração e algoritmo podem ser aplicados na resolução de problemas e na criação dos jogos.

A relevância dessa experiência é evidenciada pela forma como contribuiu para a prática educativa, incentivando a aprendizagem ativa e o desenvolvimento de habilidades do século XXI, como criatividade, colaboração e resolução de problemas. O Scratch se mostrou uma ferramenta poderosa para estimular o engajamento dos alunos, permitindo que aprendessem de maneira lúdica e interativa.

Assim, a oficina reforçou a importância de integrar o pensamento computacional ao ensino, preparando os alunos para os desafios do mundo digital. A combinação de aprendizado técnico e criatividade demonstrou que a programação pode ser acessível e envolvente, incentivando uma cultura de experimentação e inovação no ambiente escolar.

Agradecimentos

Agradecemos ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq); a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e a Unidade de Tecnologia da Educação e Cidadania - (UTEC Nova Descoberta) que proporcionaram apoio de infraestrutura e financeiro para o desenvolvimento da pesquisa.

7. References

- Ausubel, D. P. (1968). Educational Psychology: A Cognitive View. New York: Holt, Rinehart and Winston.
- Batista, E. J. S. (2024). Pensamento computacional: teoria e prática. Pensamento computacional: teoria e prática.

- Bell, S. (2010). Project-Based Learning for the 21st Century: Skills for the Future. *The Clearing House*, 83(2), 39-43.
- Brennan, K., & Resnick, M. (2020). Using design-based research to explore computational thinking in elementary school. *Proceedings of the 19th International Conference on Interaction Design and Children*, 1-10.
- Denning, P. J., & Tedre, M. (2021). Computational thinking: A disciplinary perspective. *Informatics in Education*, 20(3), 361.
- Dos Anjos, S. M., Perin, A. T., Meda, De O. P. M., Andrade, I. R. H., Freires, K. C. P., Minetto, A. V. *Tecnologia na educação: Uma jornada pela evolução histórica, desafios atuais e perspectivas futuras*. Editora: Quipá. 2024.
- Gatti, B. A. (2021). *O trabalho docente: avaliação, valorização, controvérsias*. Campinas: Autores Associados.
- Grover, S., Pea, R., & Cooper, S. (2019). Factors influencing participation and persistence in K-12 computer science education: A synthesis of literature. *Computer Science Education*, 29(2-3), 127-160.
- Hmelo-Silver, C. E. (2020). Problem-based learning: What and how do students learn? *Interdisciplinary Journal of Problem-Based Learning*, 14(1).
- Ito, M., Martin, L., Rafalow, M., Wortman, A., & Pfister, R. C. (2019). *Affinity online: How connection and shared interest fuel learning*. MIT Press.
- Luo, F., Antonenko, P. D., & Davis, E. C. (2020). Exploring the evolution of two girls' conceptions and practices in computational thinking in science. *Computers & Education*, 146, 103759.
- Madureira, J. S., & Schneider, H. N. (2024). Desenvolvendo o Pensamento Computacional por meio da estratégia de autoria de jogos associada ao ambiente de programação Scratch. *EaD & Tecnologias Digitais na Educação*, 12(16), 78-89.
- Maloney, J. Et Al. (2010). The Scratch Programming Language and Environment. *ACM Transactions on Computing Education*, 10(4), 1-15.
- Moursund, D. (1999). *Project-Based Learning Using Information Technology*. ISTE.
- Oliveira, E. (2009). O Uso do Scratch no Ensino de Programação para Crianças. *Anais do Simpósio Brasileiro de Informática na Educação (SBIE)*.
- Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books.
- Román-González, M., Moreno-León, J., & Robles, G. (2021). Assessment of computational thinking in educational contexts: A systematic literature review. *Computers & Education*, 160, 104033.
- Souza, A.; Lencastre, M. (2013). Scratch e o Desenvolvimento do Pensamento Computacional. *Anais do Workshop de Informática na Escola (WIE)*.
- Tang, X., Yin, Y., Lin, Q., Hadad, Y., Zhai, X., & Chen, G. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education*, 148, 103798.

- Wef. World Economic Forum. Schools of the Future. Defining New Models of Education for the Fourth Industrial Revolution. Switzerland. 2020.
- Wang, Hsiu Ying; Huang, Iwen; Hwang, Gwo Jen. Effects of an integrated Scratch and project-based learning approach on the learning achievements of gifted students in computer courses. In: 2014 IIAI 3rd International Conference on Advanced Applied Informatics. IEEE, 2014. p. 382-387.
- Wing, J. M. (2006). Computational Thinking. Communications of the ACM, 49(3), 33-35.
- Wing, J. M. (2021). Pensamento computacional. Educação e Matemática, (162), 2-4.
- Zaharija, G. et al. (2013). Using Scratch to Teach Programming Concepts. Journal of Computing Sciences in Colleges, 28(4), 38-44.
- Zeiser, K. Et Al. (2014). Evidence of Deeper Learning Outcomes. American Institutes for Research.