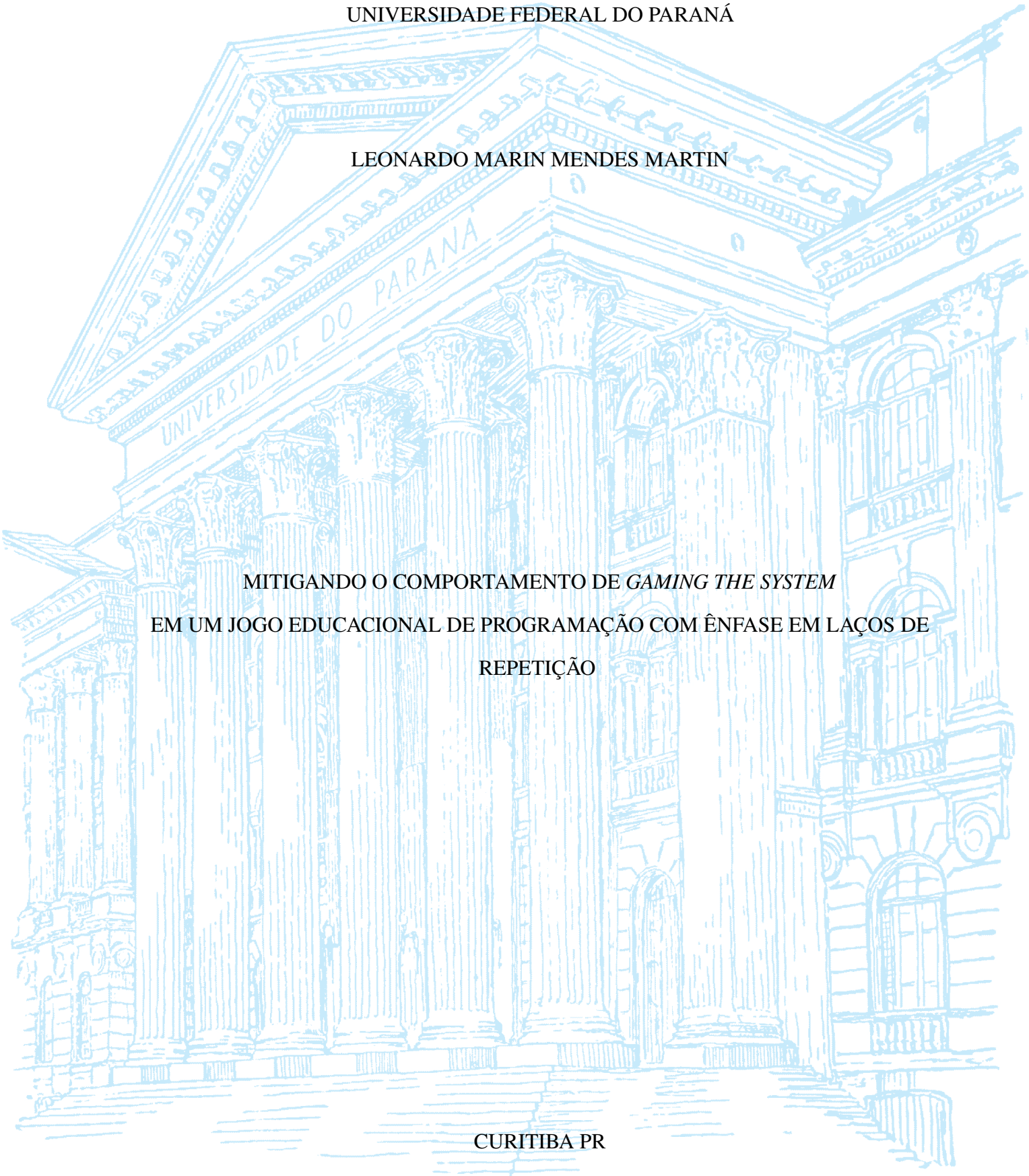
UNIVERSIDADE FEDERAL DO PARANÁ

LEONARDO MARIN MENDES MARTIN



MITIGANDO O COMPORTAMENTO DE *GAMING THE SYSTEM* EM UM JOGO EDUCACIONAL DE PROGRAMAÇÃO COM ÊNFASE EM LAÇOS DE REPETIÇÃO

CURITIBA PR

2025

LEONARDO MARIN MENDES MARTIN

MITIGANDO O COMPORTAMENTO DE *GAMING THE SYSTEM* EM UM JOGO EDUCACIONAL DE PROGRAMAÇÃO COM ÊNFASE EM LAÇOS DE REPETIÇÃO

Trabalho apresentado como requisito parcial à conclusão do Curso de Bacharelado em Ciência da Computação, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Computação*.

Orientador: Rachel Carlos Duque Reis.

CURITIBA PR

2025

Aos meus pais, por todo o apoio e incentivo ao longo da graduação.

AGRADECIMENTOS

À minha orientadora, Rachel Carlos Duque Reis, pela dedicação e orientação ao longo da realização deste trabalho. Aos meus pais, Washington Luiz Mendes Martin e Marinez Marin Mendes Martin, por todo o amor, apoio e incentivo durante a graduação. A Henrique Prokopenko e João Pedro Kieras Oliveira, pelo suporte que me deram no início do trabalho. A Deus, pela vida e pelas pessoas que tornaram essa jornada mais prazerosa.

RESUMO

Ambientes educacionais digitais de aprendizagem têm sido amplamente utilizados para apoiar o ensino em diferentes áreas, inclusive programação, favorecendo o engajamento e oferecendo *feedback* imediato ao estudante. Entretanto, esses sistemas também podem incentivar comportamentos de *gaming the system*, quando o aluno tenta "trapacear" para avançar nas atividades sem, de fato, refletir sobre a solução dos problemas. A literatura sobre *gaming the system* ainda é limitada no contexto de jogos educacionais voltados ao ensino de programação, o que evidencia uma lacuna de pesquisa. Diante disso, este trabalho tem como objetivo incorporar estratégias que utilizam um sistema de dicas para minimizar o comportamento de *gaming the system* em um jogo educacional de programação, tomando como estudo de caso o jogo *Infinity Loop*. O método adotado envolveu a pesquisa por trabalhos na literatura sobre *gaming the system*, a análise dos desafios presentes no jogo *Infinity Loop* e o desenvolvimento de novas funcionalidades, incluindo um personagem tutor e um sistema de dicas integrado a um botão com temporizador. Como resultado, obteve-se uma versão estendida do jogo, enriquecida com mecanismos que buscam desestimular tentativas aleatórias e incentivar interações mais reflexivas por parte do estudante, visando contribuir para o avanço das discussões sobre estratégias para mitigar o *gaming the system* em jogos educacionais.

Palavras-chave: *Gaming the System*. Jogo Educacional. Programação.

ABSTRACT

Digital learning environments have been widely used to support teaching in different domains, including programming, promoting student engagement and providing immediate feedback. However, these systems may also encourage *gaming the system* behaviors, in which students attempt to "cheat" or exploit the system to progress through activities without genuinely reflecting on problem-solving. The literature on *gaming the system* remains limited within the context of educational games designed for programming instruction, revealing a research gap. In this context, this work aims to incorporate strategies that employ a hint system to minimize *gaming the system* behavior in an educational programming game, using *Infinity Loop* as a case study. The adopted method involved reviewing related literature on *gaming the system*, analyzing the challenges present in *Infinity Loop*, and developing new features, including a tutor character and a hint system integrated into a timed button. As a result, an extended version of the game was obtained, enriched with mechanisms designed to discourage random attempts and promote more reflective interactions from students. This study seeks to contribute to ongoing discussions on strategies for mitigating *gaming the system* in educational games.

Keywords: *Gaming the System*. Educational Games. Programming.

LISTA DE FIGURAS

4.1 Desafio 1 do jogo educacional *Infinity Loop*. 19

5.1 Diálogo inicial do jogador com o Fazendeiro do Infinito. Adaptado de Prokopenko e Oliveira (2024). 21

5.2 Tela do Desafio 1 do jogo *Infinity Loop*. Adaptado de Prokopenko e Oliveira (2024).22

5.3 Tela do Desafio 1 após o jogador solicitar uma dica. 23

5.4 Tela do Desafio 2 do jogo *Infinity Loop*. Adaptado de Prokopenko e Oliveira (2024).24

5.5 Tela do Desafio 3 do jogo *Infinity Loop*. Adaptado de Prokopenko e Oliveira (2024).25

5.6 Tela do Desafio 4 do jogo *Infinity Loop*. Adaptado de Prokopenko e Oliveira (2024).26

5.7 Tela do Desafio 5 do jogo *Infinity Loop*. Adaptado de Prokopenko e Oliveira (2024).28

LISTA DE TABELAS

2.1 Cenário de teste no Desafio 2: tempo entre solicitações sucessivas de dicas e tempo acumulado.. 13

5.1 Dicas para o Desafio 1. 23
5.2 Dicas para o Desafio 2. 24
5.3 Dicas para o Desafio 3. 25
5.4 Dicas para o Desafio 4. 27
5.5 Dicas para o Desafio 5. 28

SUMÁRIO

1 INTRODUÇÃO 9
1.1 CONTEXTO E MOTIVAÇÃO 9
1.2 OBJETIVO 9
1.3 DESAFIOS 9
1.4 CONTRIBUIÇÃO 10
1.5 ORGANIZAÇÃO DO DOCUMENTO 10
2 *FUNDAMENTAÇÃO TEÓRICA* 11
2.1 GAMING THE SYSTEM 11
2.2 CENÁRIO DE TESTE: SOLICITAÇÕES SUCESSIVAS DE DICAS NO DESAFIO 2 12
2.3 CONSIDERAÇÕES FINAIS 13
3 TRABALHOS RELACIONADOS 14
3.1 ARTIGO I 14
3.2 ARTIGO II 15
3.3 ARTIGO III 16
3.4 CONSIDERAÇÕES FINAIS 17
4 MATERIAIS E MÉTODOS 18
4.1 ETAPA 1: SELEÇÃO E ANÁLISE DO JOGO EDUCACIONAL 18
4.2 ETAPA 2: PLANEJAMENTO DE DICAS E DAS ESTRATÉGIAS PARA MINIMIZAR O *GAMING THE SYSTEM* 19
4.3 ETAPA 3: DESENVOLVIMENTO DE UM PERSONAGEM TUTOR 20
4.4 ETAPA 4: INTEGRAÇÃO DO SISTEMA DE DICAS, DO PERSONAGEM TUTOR E DAS ESTRATÉGIAS PARA MINIMIZAR O COMPORTAMENTO DE *GAMING THE SYSTEM* 20
5 RESULTADOS. 21
5.1 NOVOS ELEMENTOS ADICIONADOS AO JOGO *INFINITY LOOP* 21
5.2 CONSIDERAÇÕES FINAIS 29
6 CONCLUSÃO 30
REFERÊNCIAS 31

1 INTRODUÇÃO

1.1 CONTEXTO E MOTIVAÇÃO

Nos últimos anos, os ambientes digitais de aprendizagem tornaram-se parte do cotidiano escolar e universitário, com o objetivo de aumentar o engajamento e a motivação dos alunos no processo de aprendizagem, oferecendo, em alguns casos, atividades interativas e com *feedback* imediato. Esses ambientes podem assumir diferentes domínios e propostas pedagógicas, como, por exemplo, os jogos educacionais *Formigas em Grafos*, que busca facilitar o entendimento do comportamento dos algoritmos de busca em largura e em profundidade (Santos e Ferreira, 2021), *RobotCode*, voltado ao apoio no ensino de lógica de programação (Honda et al., 2023) e *Robotim*, de caráter multidisciplinar, para o ensino de Matemática, Ciências e Geografia (de Oliveira et al., 2023).

Contudo, a interação com os ambientes digitais de aprendizagem nem sempre é proveitosa. É comum que parte dos estudantes tente adotar estratégias para avançar nos desafios sem refletir sobre os problemas propostos pelo ambiente de aprendizagem. Esse comportamento é conhecido na literatura como *gaming the system* (Baker et al., 2004). A prática de *gaming the system* se dá quando o aluno adota uma postura negligente ao interagir com um sistema educacional, explorando propriedades do sistema para avançar nas atividades sem raciocinar sobre formas adequadas de resolver o problema.

A manifestação desse comportamento pode ocorrer quando o estudante testa combinações aleatórias para resolver um problema até chegar à resposta correta (Baker et al., 2004). Também aparece quando, diante de um sistema com dicas, o aluno passa a solicitar pistas de forma sucessiva com o objetivo de obter a solução.

No contexto do ensino de programação, Rocha et al. (2023) apontam que os estudantes iniciantes possuem maior dificuldade no uso de laços de repetição. Para apoiar o aprendizado desse conteúdo, jogos educacionais digitais têm sido desenvolvidos. Um exemplo é o jogo *Infinity Loop*, que explora principalmente a construção e o uso de laços de repetição para resolver desafios no cenário da vila medieval para reconstruir uma ponte, consertar uma cerca, plantar árvores, entre outros (Prokopenko e Oliveira, 2024).

1.2 OBJETIVO

Este trabalho tem como objetivo incorporar estratégias para minimizar o comportamento de *gaming the system* em um jogo educacional digital para apoiar o ensino de programação**, com ênfase em laços de repetição**. No contexto deste estudo, utilizou-se o jogo *Infinity Loop*, um jogo educacional digital para auxiliar os estudantes de programação em erros causados por *misconceptions*[1](Araujo et al., 2021) em laços de repetição isolados e combinados a outras estruturas de dados, como vetores e matrizes.

1.3 DESAFIOS

A inclusão de um sistema de dicas que utiliza estratégias para mitigar o comportamento de *gaming the system* no jogo educacional *Infinity Loop* enfrentou um obstáculo central: a escassez de trabalhos científicos que relacionem o tema ao ensino de programação. A revisão de

[1]Falta de compreensão adequada dos conceitos, no contexto de programação.

literatura realizada neste Trabalho de Conclusão de Curso (TCC) reforça essa lacuna de pesquisa, sobretudo no contexto de jogos educacionais.

Na fase de implementação, as dificuldades envolveram compreender a mecânica do jogo, os desafios de programação e o código existente do *Infinity Loop* para incorporar novas funcionalidades alinhadas aos objetivos do jogo. Também foi necessário aprender sobre a Godot Engine[2] e a linguagem GDScript[3], tecnologias com as quais o autor não possuía familiaridade. Nesta etapa, destacam-se os ajustes visuais em personagens e diálogos, a criação do botão de dicas, elementos de interface, temporizadores, além do acionamento das dicas específicas de cada desafio.

Por fim, o último desafio consistiu na criação de dicas que estimulassem a reflexão do aluno sobre a resolução dos desafios sem revelar a resposta, nem mesmo na última dica. Essa atividade exigiu bastante tempo e dedicação, além de uma análise profunda e detalhada de cada desafio, para definir a quantidade e o conteúdo das dicas.

1.4 CONTRIBUIÇÃO

A principal contribuição deste TCC consiste na inclusão de um sistema de dicas **que não revela diretamente a resposta do problema, integrado a um botão com temporizador que restringe e bloqueia temporariamente novas solicitações de dicas, com o objetivo de minimizar o comportamento de *gaming the system* em um jogo educacional voltado ao ensino de programação, com foco em laços de repetição**. No ensino de programação, é comum o uso de questões de múltipla escolha nos desafios, como acontece no jogo *Infinity Loop*, em que cada desafio é composto por campos desse tipo, o que favorece a ocorrência desse comportamento (Baker et al., 2004).

1.5 ORGANIZAÇÃO DO DOCUMENTO

Este trabalho de TCC foi estruturado em seis capítulos. O Capítulo 1 apresentou o contexto e motivação, o objetivo da pesquisa, os desafios enfrentados e sua principal contribuição. O Capítulo 2 apresenta a base teórica do conceito de *gaming the system*. O Capítulo 3 elenca os trabalhos relacionados que envolvem a detecção, observação e estratégias para minimizar o comportamento de *gaming the system* em ambientes educacionais. O Capítulo 4 exibe os materiais e métodos empregados na incorporação do sistema de dicas, com estratégias voltadas à minimização do comportamento de *gaming the system* no jogo educacional *Infinity Loop*. O Capítulo 5 relata os resultados obtidos com a adição de novas funcionalidades ao jogo. Por fim, o Capítulo 6 apresenta as conclusões e sugestões de trabalhos futuros, seguido das referências bibliográficas.

[2]Engine de jogos gratuita e de código aberto que fornece as ferramentas base para criar jogos (cenas, gráficos, física, áudio e entrada)

[3]Linguagem de script nativa da Godot, simples e parecida com Python, usada para programar a lógica e o comportamento dos objetos do jogo.

2 *FUNDAMENTAÇÃO TEÓRICA*

Neste capítulo, apresenta-se o conceito de *gaming the system* e como esse comportamento se manifesta, incluindo seus tipos mais comuns e seus possíveis impactos, motivações e objetivos. Além disso, também é realizado um comparativo com outros comportamentos inadequados no processo de aprendizagem e são apresentadas abordagens para sua mitigação.

2.1 GAMING THE SYSTEM

No âmbito educacional, a prática de *gaming the system* caracteriza-se por comportamentos inadequados do aluno ao interagir com um sistema de aprendizagem (Baker et al., 2004). Nesse contexto, o estudante adota uma postura negativa no processo de aprendizagem, tentando se aproveitar das propriedades do sistema para obter uma vantagem imediata. Isso permite que o aprendiz avance nas atividades ou desafios propostos de maneira superficial, sem refletir ou raciocinar sobre as formas adequadas de resolver o problema (Baker et al., 2004).

Muitas vezes, o *gaming the system* é associado ao termo "trapaça". Embora os termos tenham características semelhantes, eles não são equivalentes. "Trapacear o sistema" implica em violar as regras de uma aplicação, enquanto o *gaming the system* diz respeito a tirar proveito das brechas do sistema computacional sem necessariamente infringir suas normas (Baker et al., 2008).

De acordo com Baker et al. (2004), o comportamento de *gaming the system* pode se manifestar de diversas formas, tais como: **(i) pedidos recorrentes de ajuda**, aproveitando-se do suporte que é dada pelo sistema, onde, em algum momento, a resposta do exercício ou atividade é dada ao aluno; **(ii) tentativas sucessivas de resolver um problema de maneira aleatória**, selecionando as alternativas disponíveis sem refletir sobre o conteúdo ou até que o sistema identifique a resposta correta. Em ambas as situações, o aprendiz explora as características do sistema como uma forma de obter a resposta correta e progredir nas atividades, sem necessariamente compreender ou se engajar no processo de aprendizagem.

Segundo Baker et al. (2004, 2008), há diversas razões pelas quais os alunos praticam o *gaming the system*. Um dos fatores ocorre quando o estudante foca apenas em obter a resposta correta, sem se preocupar com o raciocínio necessário para resolver o problema. Outro motivo está relacionado à falta de conhecimento ou domínio sobre o conteúdo. De modo geral, fatores como desinteresse pelo conteúdo, falta de motivação intrínseca, frustração, aversão a computadores e rejeição ao ambiente de aprendizagem.

Ao adotar a prática do *gaming the system*, o aprendiz opta por não enfrentar os desafios relacionados aos conteúdos em que ele não tem domínio, evitando os problemas em que ele teria que se esforçar mais para resolver (Baker et al., 2004). Uma análise realizada por Baker et al. (2004) mostra que os alunos que adotam esse comportamento o fazem por se sentirem incapazes de lidar com o material de estudo ou por estarem excessivamente focados em obter bons resultados.

De maneira geral, estudantes que "abusam" do sistema não demonstram empenho em utilizar *softwares* educacionais como ferramentas de aprendizado (Baker et al., 2004). Além do comportamento de *gaming the system*, existem outros comportamentos inadequados que influenciam negativamente o rendimento dos alunos durante a interação com sistemas educacionais. Baker et al. (2008) destacam, por exemplo, os comportamentos *off-task*, como

conversas paralelas que não abordam o conteúdo da atividade, bem como ações que desviam o foco do sistema educacional, como navegar na internet ou se engajar em outras atividades não relacionadas, como ler uma revista.

Segundo Baker et al. (2008), nenhum comportamento está tão relacionado à queda de rendimento quanto a prática de *gaming the system*. Além disso, os autores concluem que a frequência de *gaming the system* é um preditor mais assertivo para a queda de rendimento na aprendizagem do que os comportamentos "*off-task*".

O fato do *gaming the system* ser considerado o comportamento mais relacionado à queda de desempenho na aprendizagem deve-se, em grande parte, à falta de conhecimento prévio em determinado assunto (Baker et al., 2008). Essa deficiência leva os alunos a se aproveitarem do sistema, o que resulta em menos aprendizado em comparação com aqueles que persistem na tentativa de resolver o problema de maneira adequada (Baker et al., 2008).

Para minimizar o *gaming the system*, é fundamental identificar como esse tipo de comportamento é desencadeado em um sistema computacional de aprendizagem, o que permite definir a abordagem mais adequada para lidar com essa situação. Nesse contexto, Baker et al. (2004) apresentam algumas estratégias para identificar os pedidos consecutivos de ajuda e tentativas sucessivas de resolver os exercícios de forma aleatória.

Ao identificar os pedidos sucessivos de dicas, Baker et al. (2004) propõem duas abordagens para lidar com esse problema. A primeira consiste no sistema não fornecer a resposta ao aluno, mesmo no último estágio de dicas. A outra envolve a inserção de "travas de tempo" no uso das dicas, obrigando o aluno a esperar um intervalo de tempo antes de poder solicitar uma nova dica. Além disso, quando o comportamento é identificado por tentativas sucessivas de resolver o problema de forma aleatória, recomenda-se que o aluno seja orientado a resolver outro problema com o intuito de verificar se ele realmente compreendeu o conteúdo (Nunes e Jaques, 2014).

No contexto de programação, identificar os conceitos que levam programadores novatos a praticar *gaming the system* é fundamental para desenvolver sistemas educacionais que minimizem esse comportamento e potencializem o aprendizado. Nesse sentido, Rocha et al. (2023) identificaram os principais conceitos que contribuem para essa prática. Entre eles, destacam-se vetores, matrizes, estrutura condicional simples, estrutura condicional aninhada e estrutura de repetição, sendo esta última a mais recorrente.

2.2 CENÁRIO DE TESTE: SOLICITAÇÕES SUCESSIVAS DE DICAS NO DESAFIO 2

Este cenário de teste investiga se um jogador consegue obter vantagem ao solicitar dicas de forma repetida no Desafio 2, escolhido por possuir uma quantidade intermediária de dicas. O procedimento adotado foi acionar o botão *Dicas* e, em seguida, solicitar novas dicas assim que o botão fosse liberado novamente, sem tentar resolver o desafio entre uma dica e outra. Os resultados mostram a existência de um custo temporal acumulado imposto pelo mecanismo de espera entre solicitações: foram observados 13s entre a Dica 1 e a Dica 2, 18s entre a Dica 2 e a Dica 3, e 21s entre a Dica 3 e a Dica 4, totalizando 52s apenas para acessar todas as dicas (Tabela 2.1). Além disso, a análise do conteúdo das dicas indica que elas não revelam diretamente a resposta. As mensagens orientam o raciocínio e destacam erros comuns no uso do laço while, mantendo a necessidade de interpretação e aplicação do conhecimento para selecionar corretamente as opções do desafio.

Tabela 2.1: Cenário de teste no Desafio 2: tempo entre solicitações sucessivas de dicas e tempo acumulado.

Dica	Descrição	Tempo entre (s)	Tempo acumulado (s)
1	Não deve existir ponto e vírgula após a condição do laço *while*, pois isso impede a execução do corpo e pode gerar laço infinito.	0	0
2	Não devemos ter atribuições na condição do *while*. Isso pode manter a condição verdadeira e causar laço infinito.	13	13
3	Na condição do *while*, pode-se usar operador relacional (ex.: <), mas é necessário comparar uma variável com uma constante; o uso de duas constantes pode gerar laço infinito.	18	31
4	O laço *while* inicia com o iterador valendo 3, e o objetivo no corpo do laço é adicionar 5 estacas na cerca (não remover).	21	52

2.3 CONSIDERAÇÕES FINAIS

O *gaming the system* ocorre quando o estudante se aproveita das propriedades do sistema para avançar em um desafio sem refletir sobre o problema. Esse comportamento pode se manifestar, por exemplo, em tentativas aleatórias de resolução ou em solicitações sucessivas de dicas, e costuma estar associado a desinteresse, desmotivação e falta de domínio do conteúdo.

Embora existam outros comportamentos relacionados à queda de desempenho do aluno, a literatura aponta o *gaming the system* como o principal entre eles. Além disso, estudos que relacionam conceitos de programação a esse comportamento indicam as estruturas de repetição como o conceito mais recorrente, o que reforça a importância de avaliar formas de mitigá-lo em problemas envolvendo laços de repetição.

3 TRABALHOS RELACIONADOS

No contexto educacional, o comportamento de *gaming the system* é uma prática recorrente em ambientes virtuais de aprendizagem, desafiando a eficácia de jogos e sistemas educacionais (Baker et al., 2024; Nunes e Jaques, 2014). Frequentemente, estudantes exploram as limitações do sistema para avançar nas atividades sem adquirir um aprendizado real e significativo. Nesse sentido, nas próximas seções serão apresentados três artigos que possuem relação com o trabalho proposto. O primeiro investiga a influência de diferentes abordagens no ensino apoiado por agentes pedagógicos animados (Nunes e Jaques, 2014). O segundo tem como objetivo analisar o impacto dos gêneros com relação à prática de *gaming the system* (Baker et al. 2024). Por fim, o terceiro se propõe a construir um detector de *gaming the system* baseado em algoritmos de aprendizado de máquina, com o objetivo de identificar quais conceitos de programação estão mais associados a essa prática em programadores iniciantes (Rocha et al., 2023).

3.1 ARTIGO I

Nunes e Jaques (2014) propuseram a utilização de Agentes Pedagógicos Animados para minimizar o comportamento de *gaming the system* no ensino de matemática. Neste estudo, os autores desenvolveram dois agentes com posturas diferentes, um que adota uma abordagem motivacional, demonstrando preocupação com o aprendizado do aluno, e outro uma abordagem crítica, ao perceber que o aluno está praticando *gaming the system* para tentar avançar sem esforço nos desafios. Esses dois agentes foram integrados ao *PAT2Math*, um Sistema Tutor Inteligente (STI) cujo ambiente de aprendizagem é projetado para auxiliar na resolução de equações de 1º e 2º grau com uma única incógnita.

O comportamento de *gaming the system* no trabalho de Nunes e Jaques (2014) está relacionado a pedidos sucessivos de dicas. Esse comportamento é evidenciado quando o aluno pede duas ou mais dicas em sequência sem realizar nenhuma interação com o sistema educacional. Caso essas interações não ocorram, o STI considera que o aluno praticou *gaming the system*.

De maneira geral, os agentes integrados ao STI *PAT2Math* realizam intervenções quando percebem a ocorrência de *gaming the system*. No entanto, para que o sistema educacional não ficasse restrito a exibir mensagens sobre a postura do aluno, era necessário que, entre cada intervenção do agente com o estudante, fosse fornecida pelo menos uma dica relacionada ao problema em questão.

Ao realizar intervenções, os agentes exibem frases dependendo de sua personalidade. Por exemplo, o agente com abordagem motivacional poderia exibir a mensagem *"Eu confio em você, sei que é capaz de resolver o problema com seu conhecimento"*. Por outro lado, o agente com postura crítica poderia exibir a mensagem *"Tentar me enganar para conseguir a resposta só vai prejudicar seu aprendizado"*. Adicionalmente, a representação visual dos agentes alinhava-se com suas expressões verbais, permitindo transmitir emoções como tristeza, exaustão, desmotivação e até mesmo indiferença.

A fim de verificar o comportamento dos alunos, os autores submeteram 41 estudantes à resolução dos problemas propostos pelo STI *PAT2Math*. Para avaliar o comportamento desses estudantes, os autores observaram os *logs* das interações dos alunos com o STI para verificar a ocorrência do comportamento, considerando também a influência dos dois tipos de agentes pedagógicos implementados.

Os resultados mostraram que o agente crítico demonstrou ser mais efetivo para combater o comportamento de *gaming the system*, entretanto, inibiu pedidos de dicas que poderiam ser necessários, uma vez que os estudantes solicitaram menos dicas ao interagir com o STI. Além disso, os alunos que foram abordados pelo agente crítico resolveram, em média, menos equações no período de interação com o STI, indicando que a postura do agente influenciou na maneira com que os alunos interagem com o sistema.

O agente motivacional encorajou consideravelmente os pedidos de dicas por parte dos alunos, porém não aumentou de forma relevante o número de ocorrências de *gaming the system*. Apesar dos estudantes terem solicitado mais que o dobro de dicas se comparado ao agente crítico, eles resolveram, em média, mais equações do que na abordagem crítica.

Por fim, as diferentes implementações de agentes indicam que a atitude do agente motivacional pode influenciar o comportamento dos alunos em relação ao *gaming the system*. É possível observar que agentes críticos tendem a reduzir pedidos de ajuda, enquanto agentes motivacionais são mais procurados, sem aumentar a frequência de *gaming the system* de forma significativa.

3.2 ARTIGO II

Em uma pesquisa desenvolvida por Baker et al. (2024), os autores investigaram como o comportamento de *gaming the system* influencia os resultados de aprendizagem entre estudantes do sexo feminino e masculino, com o objetivo de aprimorar o desenvolvimento de jogos e promover melhores resultados de aprendizagem para todos os estudantes. Este estudo foi motivado por pesquisas anteriores que revelaram que estudantes do sexo feminino apresentavam desempenho superior em termos de aprendizagem com jogos educacionais em comparação aos estudantes do sexo masculino.

Para analisar o comportamento dos estudantes, os autores utilizaram o jogo *Decimal Point*, desenvolvido para alunos do 5º ao 7º ano do ensino fundamental, com o objetivo de ensinar números decimais, operações e conceitos relacionados. O jogo se passa em um parque de diversões e é composto por 24 mini-jogos. No total, foram apresentados 72 problemas distribuídos em cinco categorias: ordenação de decimais, posicionamento na linha numérica, preenchimento de sequências decimais, classificação em “menor que” e “maior que” e adição de decimais. A escolha dos conteúdos foi feita para abordar conceitos errôneos comuns sobre números decimais, como a ideia de que decimais mais longos têm maior valor (ex.: 0,234 > 0,9).

Cada problema do jogo é composto por duas etapas: resolução de problemas e autoexplicação. Na etapa de resolução do problema, o aluno joga para resolver o desafio, para, por exemplo, ordenar números decimais. Por outro lado, na etapa de autoexplicação, o aluno escolhe a explicação correta que resolve o problema a partir de uma lista de múltipla escolha. Para avançar no jogo, os alunos precisam responder corretamente em ambas as etapas. Além disso, não há penalidades ou limite de tentativas para respostas incorretas, deixando o ambiente de aprendizado mais flexível.

Para investigar como o comportamento de *gaming the system* influencia os resultados de aprendizagem entre estudantes do sexo feminino e masculino, foram realizados três estudos distintos, cada um projetado para investigar diferentes aspectos desse comportamento em relação ao desempenho dos alunos no jogo *Decimal Point*. Exemplos de *gaming the system* observados incluíram clicar rapidamente para obter dicas e selecionar todas as opções de múltipla escolha em sequência.

O Estudo 1 teve como objetivo a construção de modelos de aprendizado de máquina para detectar esses comportamentos. Neste estudo, os estudantes foram submetidos a três testes para

medir o aprendizado sobre operações com números decimais e detectar *misconceptions*. Os testes consistiam em: pré-teste para avaliar o conhecimento prévio, pós teste aplicado imediatamente após a intervenção para medir o aprendizado imediato e o teste tardio realizado uma semana após a intervenção para avaliar a retenção de conhecimento. Por meio da construção de modelos de detecção de *gaming the system*, foi possível observar correlação negativa entre o comportamento de *gaming the system* e o desempenho dos estudantes nos testes, especialmente no teste tardio. Em geral, foi observado que meninos praticaram mais *gaming the system* do que as meninas na condição lúdica, principalmente durante a fase de autoexplicação.

O Estudo 2 se propôs a explorar o impacto da autonomia dos alunos no comportamento de *gaming the system* e no aprendizado. Os participantes usaram uma versão do jogo *Decimal Point*, dividida em duas condições: condição de baixa autonomia, em que os alunos não podiam escolher mini-jogos ou desistir, e condição de alta autonomia, que permitia escolhas na ordem dos mini-jogos e desistência após completar 24 rodadas. Os resultados reforçaram a tendência de maior *gaming the system* entre meninos e indicaram que esse comportamento mediava a relação entre gênero e desempenho nos testes, sugerindo que as meninas obtiveram um desempenho acadêmico mais consistente devido ao menor envolvimento com essa prática.

O Estudo 3 replicou os testes de conhecimento e as investigações dos Estudos 1 e 2, adicionando uma terceira abordagem, que introduziu alta autonomia ao remover o caminho visual pelo parque. Pesquisas anteriores indicaram que, mesmo com a opção de escolher a ordem dos mini-jogos, os alunos tendiam a seguir o caminho visual, o que poderia ter influenciado suas escolhas no Estudo 2. Os resultados deste estudo mostraram que os meninos continuaram praticando o comportamento de *gaming the system* com mais frequência do que as meninas, especialmente nas etapas de autoexplicação.

Uma análise detalhada dos três estudos evidenciou uma tendência de gênero em relação ao comportamento de *gaming the system*. Observou-se que os meninos praticaram mais esse comportamento nas etapas de autoexplicação e apresentaram um desempenho inferior nos testes realizados, enquanto as meninas, que praticaram menos *gaming the system*, demonstraram um aprendizado mais consistente ao longo do tempo.

Os resultados dos três estudos evidenciaram uma forte correlação negativa entre o comportamento de *gaming the system* e o desempenho nas atividades propostas pelo jogo, especialmente no pós-testes, sugerindo que, ao evitar uma abordagem mais engajada com assunto estudado, os alunos comprometem seu aprendizado a longo prazo.

Por fim, a pesquisa indica que reformular as questões em formatos menos suscetíveis ao *gaming the system* pode ajudar a reduzir essa prática e melhorar o impacto dos jogos educativos no aprendizado. Dessa forma, as contribuições dos estudos realizados favorecem o desenvolvimento de ferramentas educacionais mais eficazes, que busquem introduzir estratégias para diminuir o *gaming the system* em estudantes do sexo masculino, promovendo melhores resultados de aprendizagem para todos os alunos.

3.3 ARTIGO III

No trabalho realizado por Rocha, Costa & Tedesco (2023), os autores analisaram a ocorrência de *gaming the system*, em estudantes novatos na área de programação, com o objetivo de construir um detector de *gaming the system* utilizando algoritmos de aprendizado de máquina e entender quais conceitos de programação estavam mais relacionados a manifestação deste comportamento. Neste estudo, foi utilizado um conjunto de dados obtidos a partir de um curso universitário durante o semestre de 2022. Esse conjunto de dados é composto por informações

sobre tentativas de resolução de problemas de programação contendo 5307 instâncias. Do número total de instâncias, 1263 foram classificadas com o comportamento de *gaming the system*.

Foram analisados 16 conceitos de programação, entre eles: comentários, tipos de dados, declaração de variáveis, manipulação de variáveis, escopo da variável, constantes, incremento de variável, decréscimo de variável, operadores relacionais, operadores booleanos, operadores aritméticos, estrutura condicional simples, estrutura condicional aninhada, estrutura de repetição, vetores e matrizes.

Neste trabalho, o principal conceito relacionado à prática de *gaming the system* foram as estruturas de repetição, representando 15,8% das observações. Além deste conceito, estruturas condicionais e estruturas condicionais aninhadas foram outros dois conceitos mais relacionados a esta prática. Em geral os problemas enfrentados pelos alunos com esses assuntos estão associados a controle de fluxo.

3.4 CONSIDERAÇÕES FINAIS

Com base nos trabalhos apresentados neste capítulo, observou-se que o comportamento de *gaming the system* ocorreu em disciplinas como matemática e programação, nas quais os alunos tentaram progredir nas atividades contornando os objetivos pedagógicos dos sistemas. Os principais objetivos dos estudos foram identificar as formas pelas quais o *gaming the system* ocorre, os conceitos que influenciam essa prática, além de desenvolver e analisar estratégias para minimizá-lo. Entre as manifestações mais comuns de *gaming the system* observadas, destacam-se os pedidos sucessivos de dicas e as tentativas aleatórias de resolução, especialmente em problemas de múltipla escolha.

Diversas estratégias foram utilizadas para investigar e mitigar a prática de *gaming the system*. Em jogos educacionais, os pesquisadores variaram a autonomia dos alunos, permitindo a escolha de atividades ou restringindo a sequência dos desafios (Baker et al., 2024). Em sistemas educacionais, foram introduzidos agentes pedagógicos animados com diferentes abordagens para influenciar a interação dos estudos com um STI (Nunes e Jaques, 2014). De forma complementar, a análise dos perfis dos alunos, como o gênero, revelou que características individuais também influenciam a propensão ao *gaming the system* (Baker et al., 2024).

Com base nas estratégias e resultados apresentados, destaca-se a importância de identificar os fatores específicos que contribuem para o comportamento de *gaming the system* em diferentes contextos educacionais. No ensino de programação, em particular, compreender os principais conceitos que levam programadores novatos a adotar esse comportamento é essencial para projetar soluções mais assertivas. Nesse sentido, este trabalho se propõe a implementar estratégias que minimizem o *gaming the system* em um jogo educacional voltado ao ensino de programação, com foco em laços de repetição.

4 MATERIAIS E MÉTODOS

Com o intuito de alcançar o objetivo estabelecido, este trabalho foi desenvolvido em quatro etapas. A primeira consistiu na seleção e análise detalhada de um jogo educacional, com foco nos desafios. A segunda etapa envolveu o planejamento das dicas e das estratégias para minimizar o comportamento de *gaming the system*. Na terceira etapa foi desenvolvido um personagem tutor, responsável por apresentar as dicas para auxiliar o jogador na solução dos desafios. Por fim, na quarta etapa, foi feita a integração das dicas, das estratégias para minimizar o comportamento de *gaming the system* e do personagem tutor ao jogo educacional selecionado. Cada uma dessas etapas será detalhada nas próximas seções.

4.1 ETAPA 1: SELEÇÃO E ANÁLISE DO JOGO EDUCACIONAL

Para o desenvolvimento desta pesquisa, foi selecionado o jogo educacional *Infinity Loop* (Prokopenko e Oliveira, 2024), cujo objetivo é auxiliar estudantes no aprendizado de programação com ênfase em laços de repetição, sendo destinado a alunos de cursos técnicos e a estudantes em início de graduação na área de tecnologia. O jogo educacional *Infinity Loop* foi escolhido por apresentar desafios de múltipla escolha que podem levar o jogador a tentar "adivinhar" a resposta correta por tentativa e erro, sem refletir sobre a solução. Por não haver limite de tentativas, isso pode incentivar a exploração aleatória das opções ao invés da análise do enunciado, comprometendo o objetivo pedagógico de promover a compreensão dos conceitos. Como os desafios abordam os principais *misconceptions* que ocorrem com estruturas de repetição, conteúdo em que os estudantes costumam ter maior dificuldade segundo Rocha et al. (2023), a tendência de buscar atalhos sem reflexão se torna ainda maior.

O jogo Infinity Loop visa corrigir erros cometidos pelos alunos, causados por *misconceptions*, ou seja, pela falta de compreensão adequada dos conceitos de programação (Araújo et al., 2021), especialmente no uso de laços de repetição. No contexto do ensino de programação, os *misconceptions* manifestam-se em erros de sintaxe, dificuldades na compreensão da semântica e outros obstáculos recorrentes enfrentados pelos estudantes, como a tentativa de utilizar variáveis fora do escopo ou a confusão entre valores e índices em laços de repetição (Araujo et al., 2021). Por exemplo, o código da Figura 4.1 ilustra um erro conceitual comum em que o jogador tenta acessar uma posição inválida de um vetor. Considerando a utilização da linguagem C no jogo, essa situação ocorre ao atribuir um número negativo como valor inicial da variável de controle do laço (i = -1), ou ainda, ao definir uma condição de parada em que o número de iterações exceda o tamanho do vetor (i <=7) (Prokopenko e Oliveira, 2024).

O jogo se desenvolve em uma fazenda, com cinco personagens secundários (NPCs - *Non-Playable Characters*), enquanto o jogador assume o papel do personagem principal, que deve passar por cinco fases. Em cada fase, é necessário resolver um desafio que envolve o uso de estruturas de repetição para avançar no jogo. Por exemplo, conforme mostrado na Figura 4.1, o jogador deve inserir sete tábuas em sequência para reconstruir uma ponte quebrada, representada no desafio por um vetor que inicia na posição de índice zero.

No início de cada fase do jogo, é exibida uma caixa de diálogo em que os NPCs pedem ajuda ao jogador, (no caso da Figura 4.1, para consertar a ponte) enquanto o desafio correspondente é apresentado posteriormente no canto direito da tela. Cada desafio exibe um código com opções de múltipla escolha, permitindo que o jogador selecione a resposta que

Figura 4.1: Desafio 1 do jogo educacional *Infinity Loop*.

considera correta. Abaixo do código, estão disponíveis os botões de executar (*Run Code*) e fechar (*Close*), além de um enunciado com as instruções sobre o desafio.

Vale ressaltar que as fases do jogo possuem níveis de dificuldade crescente, começando com desafios mais simples e evoluindo para situações mais complexas. Cada fase está associada à mecânica do ambiente da fazenda, na qual o jogador deve consertar ou completar elementos do cenário ao mesmo tempo em que enfrenta problemas de programação relacionados a *misconceptions* relacionados a estruturas de repetição.

No Desafio 1, o jogador deve reconstruir uma ponte quebrada, trabalhando o erro de acesso a posição inválida do vetor. No Desafio 2, é necessário consertar uma cerca, lidando com o problema de laço infinito. O Desafio 3, por sua vez, consiste em plantar árvores para completar a floresta, abordando o uso incorreto de operadores relacionais. No Desafio 4, o objetivo é plantar sementes de trigo e milho, enfrentando o uso incorreto de operadores lógicos. Por fim, o Desafio 5 propõe plantar flores em um jardim, reunindo uma combinação de erros: acesso a posição inválida do vetor, laço infinito e uso incorreto de operadores relacionais.

4.2 ETAPA 2: PLANEJAMENTO DE DICAS E DAS ESTRATÉGIAS PARA MINIMIZAR O *GAMING THE SYSTEM*

Para projetar as dicas a serem disponibilizadas nos desafios do jogo *Infinity Loop*, foi necessário realizar a análise dos enunciados, das respostas corretas e também das possíveis respostas incorretas, sendo estas últimas definidas com base nos principais *misconceptions* relacionados a cada desafio. Essa etapa foi fundamental para o planejamento das dicas que posteriormente foram integradas ao jogo.

Para a criação do "banco de dicas", todas as dicas elaboradas para os cinco desafios foram registradas em um arquivo de texto `.txt`. Inicialmente, as dicas foram construídas pelo autor deste TCC a partir de uma análise cuidadosa de cada desafio e, posteriormente, revisadas por uma especialista com formação na área de Informática na Educação. De modo geral, o objetivo das dicas é orientar o raciocínio do jogador de forma progressiva, auxiliando-o a identificar a resposta correta por meio da eliminação das opções incorretas, sem revelar a solução, nem mesmo na última dica. A quantidade foi definida de acordo com o número de dicas que poderiam ser extraídas de cada desafio. No primeiro desafio foram disponibilizadas três dicas, enquanto os demais contaram com quatro ou cinco dicas.

A fim de implementar uma estratégia que minimizasse o *gaming the system* nos desafios, inicialmente foram analisados os comportamentos que poderiam favorecer essa prática. Identificou-se que um deles seria a realização de tentativas aleatórias sucessivas de resolução dos desafios, até chegar à resposta correta. Além disso, estudos mostram que o mecanismo de dicas, quando utilizado de forma recorrente, pode favorecer esse tipo de comportamento (Baker et al., 2004). Logo, para minimizar a prática de *gaming the system*, as dicas foram elaboradas de modo a não revelar diretamente a resposta, mesmo em seu último estágio. Adicionalmente, foram introduzidas travas de tempo que exigem que o jogador aguarde um intervalo antes de solicitar uma nova dica, conforme proposto por Baker et al. (2004). **É importante salientar que o tempo para pedir uma nova dica aumenta progressivamente conforme são solicitadas novas dicas.**

4.3 ETAPA 3: DESENVOLVIMENTO DE UM PERSONAGEM TUTOR

Com o objetivo de guiar o jogador, foi criado um novo personagem denominado fazendeiro do infinito. Esse personagem é exibido no início do jogo para instruir o jogador e auxiliá-lo na resolução dos desafios.

Para manter a coerência com o cenário do jogo, o novo personagem foi idealizado com vestimenta típica de fazendeiro e uma expressão jovial, transmitindo a imagem de alguém capaz de orientar o jogador por meio de dicas ao longo dos desafios. A geração das artes do personagem foi realizada com o uso de inteligência artificial, que produziu duas imagens: uma representação de corpo inteiro, utilizada no mapa do jogo, e outra apenas do rosto, utilizada nos diálogos entre o personagem e o jogador.

4.4 ETAPA 4: INTEGRAÇÃO DO SISTEMA DE DICAS, DO PERSONAGEM TUTOR E DAS ESTRATÉGIAS PARA MINIMIZAR O COMPORTAMENTO DE *GAMING THE SYSTEM*

Para a integração do sistema de dicas com o personagem tutor no jogo *Infinity Loop*, foi utilizada a linguagem GDScript em conjunto com a Godot Engine[1], mesma plataforma em que a versão anterior do jogo havia sido construída. No decorrer da integração, foi necessário aprender a manipular o sistema de diálogos *Dialogue Manager*[2], a fim de inserir as dicas no ambiente do jogo, bem como aplicar conhecimentos fundamentais de GDScript[3] para que essas dicas fossem apresentadas em formato de diálogo quando o jogador acionasse o botão de dicas. Além disso, esses conhecimentos também foram empregados na implementação de um temporizador associado ao botão, que desabilita sua utilização por alguns segundos e altera o texto para exibir a contagem regressiva até que o próximo uso seja liberado. **Por fim, todo o material implementado foi disponibilizado no GitHub (link).**

[1] Documentação oficial da Godot Engine disponível em: `https://docs.godotengine.org`.

[2] Documentação oficial do addon Dialogue Manager disponível em: `https://github.com/nathanhoad/godot_dialogue_manager`.

[3] Documentação oficial da linguagem GDScript em: `https://docs.godotengine.org/pt-br/4.x/tutorials/scripting/gdscript`.

5 RESULTADOS

Este capítulo apresenta as telas modificadas e adicionadas ao jogo *Infinity Loop*, introduz os desafios e as respectivas dicas, e discute o papel dessas dicas no jogo. Para cada desafio, foi criada uma tabela que descreve as dicas e a ordem em que são exibidas, mostrando respostas corretas e incorretas que são possíveis de inferir a partir de cada uma delas. Em seguida, foi feita uma discussão evidenciando como as dicas orientam o raciocínio do jogador e quais conhecimentos são necessários para a conclusão do desafio.

5.1 NOVOS ELEMENTOS ADICIONADOS AO JOGO *INFINITY LOOP*

Três novos elementos foram adicionados ao jogo *Infinity Loop*: (1) o personagem Fazendeiro do Infinito, (2) o sistema de dicas e (3) o botão destinado à solicitação dessas dicas. Conforme mostrado na Figura 5.1, assim que o jogo inicia, o jogador é apresentado ao Fazendeiro do Infinito, que explica a dinâmica dos desafios (*Como jogar?*), se apresenta como o personagem que irá desempenhar o papel de tutor (*Quem é você?*) e exibe os nomes dos autores do jogo (*Quem são os autores deste jogo?*). Esse personagem, contudo, não se limita à introdução, ele também acompanha o jogador ao longo dos desafios, fornecendo dicas sempre que necessário. A Figura 5.1 apresenta a tela inicial do jogo, no momento em que ocorre a primeira interação do jogador com o Fazendeiro do Infinito.

Figura 5.1: Diálogo inicial do jogador com o Fazendeiro do Infinito. Adaptado de Prokopenko e Oliveira (2024).

Ao iniciar o primeiro desafio, após o diálogo inicial entre o NPC Camponês e o jogador, é exibida uma tela contendo o enunciado do desafio à direita, como mostrado na Figura 5.2. Nesse desafio, o jogador deve reconstruir uma ponte inserindo 7 tábuas por meio de um laço *for*, preenchendo os campos de inicialização e condição de parada. Nessa tela, o usuário deve selecionar as combinações adequadas para o problema e utilizar o botão "*Run Code*" a fim de validar se a resposta fornecida corresponde à correta ou não.

Para apoiar o jogador durante os desafios, foi adicionado um botão específico para a solicitação de dicas, acompanhado por um contador que representa a quantidade de dicas disponíveis. Cada desafio possui um número previamente definido de dicas, que são disponibilizadas de forma progressiva, seguindo uma sequência lógica que orienta o raciocínio do jogador na

escolha das alternativas corretas. No caso da Figura 5.2, foram disponibilizadas três dicas para auxiliar o jogador na resolução do Desafio 1.

Figura 5.2: Tela do Desafio 1 do jogo *Infinity Loop*. Adaptado de Prokopenko e Oliveira (2024).

Assim que o botão é acionado pelo jogador e a dica é apresentada, uma janela de diálogo é exibida no canto inferior esquerdo, mostrando a orientação fornecida pelo Fazendeiro do Infinito (Figura 5.3). Além disso, o número disponível no botão de dicas é decrementado e esse botão é temporariamente desativado por meio de um temporizador, implementado para evitar solicitações consecutivas e estimular o jogador a refletir sobre a orientação recebida antes de requisitar uma nova dica.

O tempo de inatividade do botão, ilustrado na Figura 5.3, foi definido em 10 segundos, considerando o intervalo necessário para ler a dica exibida na caixa de diálogo (aproximadamente 6 segundos) e resolver o Desafio 1. Esse tempo aumenta conforme o número de dicas restantes diminui, de modo a proporcionar ao jogador um período maior de reflexão sobre o desafio a cada nova orientação recebida. Quando o tempo de inatividade esgota e ainda existem dicas disponíveis, o botão é habilitado novamente e o jogador pode solicitar uma nova dica. **No entanto, quando as dicas se esgotam, o botão permanece desabilitado e o jogador é informado visualmente de que não há mais dicas disponíveis.**

Para o Desafio 1, foram elaboradas três dicas para auxiliar o jogador. Essas dicas podem ser visualizadas na Tabela 5.1. Essa tabela apresenta, para cada dica, um identificador no campo "Dicas", o texto exibido ao jogador no campo "Descrição" e as inferências possíveis decorrentes dessa informação, ou seja, quais opções incorretas, indicadas no campo "Opção incorreta" podem ser descartadas e, eventualmente, qual opção correta pode ser deduzida conforme o campo "Opção correta". Essas opções correspondem aos campos que o usuário deve completar para executar o desafio e validar a solução.

A Dica 1, apresentada na Tabela 5.1, faz referência ao primeiro campo a ser preenchido pelo jogador, que diz respeito à inicialização do laço *for*. As Dicas 2 e 3, por sua vez, estão relacionadas ao segundo campo, correspondente à condição de parada do laço. O propósito das dicas é chamar a atenção do jogador para alguns detalhes importantes sobre iterações em vetores, como: (1) vetores na linguagem C são indexados a partir de zero (Dica 1); e (2) ao utilizar o operador < ou <= na condição de parada de um laço, para percorrer um vetor, é comum que o número máximo de iterações corresponda, respectivamente, às expressões TAMANHO DO VETOR (Dica 3) e TAMANHO DO VETOR - 1 (Dica 2). Para interpretar adequadamente

Figura 5.3: Tela do Desafio 1 após o jogador solicitar uma dica.

Tabela 5.1: Dicas para o Desafio 1.

Dica	Descrição	Opção incorreta	Opção correta
1	Note que os vetores na Linguagem C iniciam na posição 0, não sendo possível utilizar valores negativos como posições do vetor.	$i = -1$	$i = 0$
2	Lembre-se que, na condição de parada de um laço, podemos utilizar o operador relacional menor ou igual, "≤", desde que o valor seja até o TAMANHO DO VETOR −1.	$i \leq 7$	$i \leq 6$
3	Também podemos utilizar o operador relacional menor, "<", desde que o valor seja até o TAMANHO DO VETOR.	–	$i < 7$

essas orientações, espera-se que o jogador tenha familiaridade com o conceito de vetor e com os operadores relacionais < e <=.

A Figura 5.4 apresenta o Desafio 2, cujo objetivo é restaurar a cerca da fazenda utilizando a estrutura de repetição *while*. Nesse desafio, o jogador deve preencher dois campos: o primeiro define a condição de parada do laço, que é iniciado com o iterador valendo três *(i = 3)*, e o segundo indica como o laço deve ser incrementado.

Para o Desafio 2, foram elaboradas quatro dicas destinadas a auxiliar o jogador durante a resolução da atividade. Essas dicas estão apresentadas na Tabela 5.2.

Na Tabela 5.2, as três primeiras dicas dizem respeito ao primeiro campo do desafio (a condição do laço *while*). A Dica 1 alerta o jogador que inserir um ponto e vírgula imediatamente após a condição (*while(...);*) faz com que o corpo do laço seja ignorado e pode levar à execução de um laço infinito, caso a condição seja verdadeira. A Dica 2 orienta o jogador a não realizar atribuição na condição do laço (*while (i = 8)*), pois isso pode tornar a condição sempre verdadeira, levando também à execução de um laço infinito. A Dica 3 ressalta que ao comparar constantes numéricas (*while (1 < 8)*), isso pode resultar em laço infinito caso a condição seja verdadeira. Por outro lado, a Dica 4 refere-se ao segundo campo do desafio, responsável pela atualização da variável de iteração *i*. Como o objetivo é adicionar estacas à cerca, o contador deve ser incrementado a cada passo e não decrementado.

Figura 5.4: Tela do Desafio 2 do jogo *Infinity Loop*. Adaptado de Prokopenko e Oliveira (2024).

Tabela 5.2: Dicas para o Desafio 2.

Dica	**Descrição**	**Opção incorreta**	**Opção correta**
1	Lembre-se que não deve existir ponto e vírgula após a condição do laço *while*, pois o corpo do laço entre chaves nunca será executado. Além disso, isso gera laço infinito.	$(i \leq 7);$	$(i \leq 7)$ $(i < 8)$
2	Observe que não podemos ter atribuições na condição do laço *while*. Isso também irá gerar laço infinito se o valor atribuído for diferente de zero.	$(i = 8)$	$(i \leq 7)$ $(i < 8)$
3	Note que, na condição do laço *while*, podemos usar o operador relacional menor, "<". No entanto, em um dos lados temos que usar uma variável e, no outro, uma constante numérica. O uso de duas constantes numéricas pode gerar laço infinito.	$(1 < 8)$	$(i < 8)$
4	Perceba que o laço *while* é iniciado com o iterador valendo 3 e que o objetivo, no corpo do laço, é adicionar 5 estacas na cerca, e não remover.	$i - -;$	$i + +;$

Em geral, o papel das dicas no Desafio 2 é evidenciar que existem diferentes maneiras de gerar laço infinito ao manipular estruturas de repetição de forma inadequada. Além disso, as dicas reforçam ao jogador que, para adicionar estacas, é necessário somar ao contador e não subtrair.

A Figura 5.5 ilustra o Desafio 3, cujo propósito é plantar quatro árvores no lado esquerdo de um caminho, garantindo o alinhamento com as árvores do lado direito. Nesse desafio, o jogador deve preencher dois campos: o primeiro corresponde à condição de parada do laço *for*, iniciado com o iterador igual a um $(i = 1)$; e no segundo, o jogador deve selecionar um operador relacional *(<, <=, ==, >=, >, !=)* para assegurar que as árvores do lado esquerdo fiquem alinhadas às árvores do lado direito.

Para o Desafio 3, foram elaboradas três dicas com o objetivo de orientar o jogador na resolução da atividade. Essas dicas podem ser visualizadas na Tabela 5.3.

Figura 5.5: Tela do Desafio 3 do jogo *Infinity Loop*. Adaptado de Prokopenko e Oliveira (2024).

Tabela 5.3: Dicas para o Desafio 3.

Dica	Descrição	Opção incorreta	Opção correta
1	Observe que, nesse caso, o vetor inicia na posição de índice 1 e não 0, pois a variável de iteração é iniciada com valor 1.	$i < tam\_floresta$ $i < tam\_floresta - 1$ $i \leq tam\_floresta - 1$	$i \leq tam\_floresta$
2	Note que queremos plantar árvores nas iterações 2, 4, 6 e 8, ou seja, iterações pares.	$>=$, $<$, $!=$	$==$, $\leq$
3	Como queremos plantar árvores em iterações pares, o resto da divisão não pode ser diferente de 0.	$!=$	$==$, $\leq$
4	Note que, como o iterador do laço inicia em 1 e é incrementado a cada iteração ($i++$), não é possível que $i\%2$ seja um número menor que zero.	$<$	$==$, $\leq$
5	Descarte a opção que planta as árvores em iterações pares e ímpares.	$>=$	$==$, $\leq$

Na Tabela 5.3, a Dica 1 está associada ao primeiro campo do desafio, que define a condição de parada do laço. Essa dica chama a atenção para o fato de que o iterador do laço começa em 1 (um) e não em 0 (zero). Esse detalhe tem como proposta auxiliar o jogador a descartar os operadores relacionais incorretos. As demais dicas estão ligadas ao segundo campo do desafio, responsável por decidir em quais iterações serão plantadas as árvores. Essas dicas conduzem o jogador em direção à alternativa correta, descartando gradualmente opções incorretas. A Dica 2 informa que as árvores devem ser plantadas apenas em iterações pares, o que implica selecionar as iterações em que o resto da divisão por 2 é igual a zero *(i % 2 == 0)*. A Dica 3 reforça essa ideia ao indicar que, se o objetivo é plantar somente nas iterações pares, não faz sentido aceitar resultados em que *i % 2* seja diferente de zero. A Dica 4 esclarece que o valor de *i % 2* nunca será negativo, pois *i* é um inteiro que assume valores não negativos, de modo que condições do tipo *(i % 2 < 0)* não são válidas. Por fim, a Dica 5 orienta o jogador a eliminar também a alternativa que planta árvores em todas as iterações (pares e ímpares). Nessa alternativa, a condição utilizada é *i % 2 >= 0*. Como o resto da divisão de *i* por 2 (*i % 2*) só pode assumir os valores 0 ou 1, essa condição é sempre verdadeira. Isso faz com que o plantio ocorra

tanto em iterações pares quanto em iterações ímpares, o que não atende ao objetivo do desafio, que é plantar apenas nas iterações pares.

Em termos gerais, as dicas do Desafio 3 têm dois propósitos. Primeiro, destacar que a análise cuidadosa do valor inicial do iterador influencia diretamente a definição correta da condição de parada do laço. Segundo, guiar o jogador no raciocínio sobre onde as árvores serão plantadas, evidenciando que a plantação deve ocorrer apenas nas iterações pares e mostrando porque cada alternativa incorreta não satisfaz o enunciado, conduzindo o jogador à chegar na resposta correta, a partir da eliminação das incorretas. Para resolver o Desafio 3, espera-se ainda que o jogador tenha conhecimento prévio sobre o operador de resto da divisão inteira *(i % 2)* e saiba interpretar seus resultados, entendendo que valores pares geram resto igual a zero e valores ímpares geram resto igual a um.

A Figura 5.6 apresenta o Desafio 4, cujo objetivo é gerenciar o plantio em uma área restrita da fazenda (ao lado da casa), considerando um número limitado de sementes de milho e trigo, utilizando a estrutura de repetição *while*. Nesse desafio, o jogador deve completar dois campos do laço: (1) selecionar o operador lógico que relaciona as quantidades de milho e de trigo, (2) selecionar o operador lógico que relaciona as quantidades de sementes e o limite da área disponível para plantação. A cada iteração, a quantidade de sementes de milho e trigo é reduzida, o tamanho da plantação é aumentado e o plantio avança para o próximo espaço disponível.

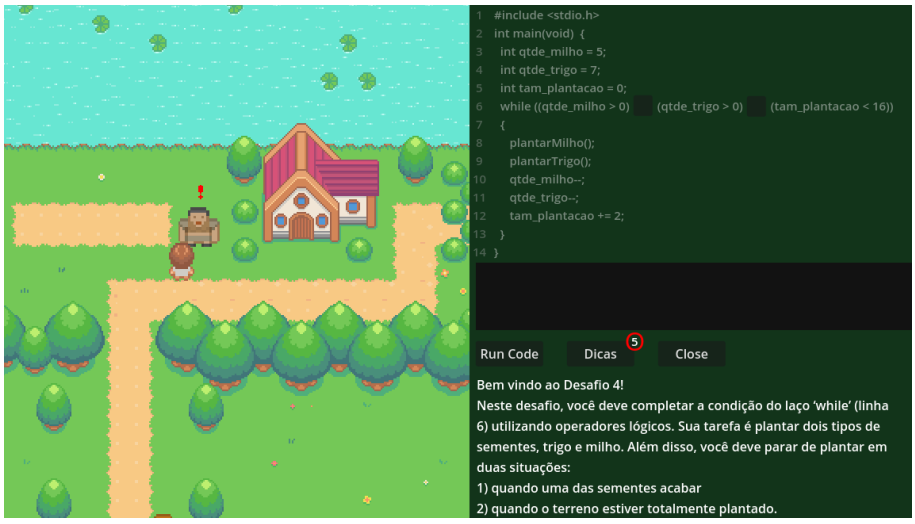


Figura 5.6: Tela do Desafio 4 do jogo *Infinity Loop*. Adaptado de Prokopenko e Oliveira (2024).

Para o Desafio 4, foram elaboradas cinco dicas que orientam a seleção dos operadores lógicos levando em conta as quantidades de milho e trigo, assim como a área disponível para plantação. Essas dicas estão apresentadas na Tabela 5.4.

Conforme mostrado na Tabela 5.4, a Dica 1 chama atenção para o fato de que não é possível continuar plantando no terreno se uma das sementes de milho ou trigo acabar. Além disso, essa dica busca antecipar uma informação importante do cenário: o valor final da variável `tam_plantacao` é 10, ou seja, após o laço *while* encerrar sua execução, a variável `tam_plantacao` estará com valor igual a 10.

A Dica 2 tem a função de desencorajar o uso do operador || (OU) no primeiro campo do laço *while*. Dessa forma, busca-se enfatizar que o laço continuará sendo executado independente da condição falsa (`qtde_milho` > *0*) aparecer antes ou depois da condição verdadeira (`qtde_trigo` > *0*). A Dica 3 reforça a ideia de não usar o operador || (OU) no primeiro campo, só que agora deixando explícito que a semente de milho acaba primeiro que a

Tabela 5.4: Dicas para o Desafio 4.

Dica	Descrição	Opção incorreta	Opção correta

1	Lembre-se que você não pode continuar plantando se uma das sementes acabar. Logo, o valor final da variável `tam_plantacao` é 10.	`||` − `||`
		`&&` − `||`
		`||` − `&&`
			`&&` − `&&`

2	Se a quantidade de uma das sementes acabar e você usar o operador lógico OU, o laço *while* continuará sendo executado se houver espaço disponível no terreno.	`||` − `&&` `||` − `||`	`&&` − `&&`
3	As quantidades de milho e trigo são diferentes. Ao escolher o operador lógico OU, você vai tentar plantar a semente de milho, que já não está mais disponível.	`||` − `&&` `||` − `||`	`&&` − `&&`
4	Apesar de o terreno permitir que sejam plantadas 8 sementes de milho e 8 sementes de trigo, isso nunca vai acontecer. Logo, o operador lógico OU não pode ser usado, pois a quantidade de sementes de milho é 5 e de trigo é 7; ou seja, o terreno nunca será totalmente preenchido.	`||` − `||` `&&` − `||`	`&&` − `&&`

5	Para plantar é necessário ter sementes de milho e trigo, e área de plantação disponível, ou seja, todas as condições devem ser satisfeitas.	`||` − `||`
		`&&` − `||`
		`||` − `&&`
			`&&` − `&&`

de trigo. Logo, uma condição baseada em || (OU) ainda permitiria tentar plantar milho quando já não há milho disponível, o que está incorreto para o objetivo do desafio.

A Dica 4 tem como proposta desencorajar o uso do || (OU) na segunda lacuna do laço *while*, chamando a atenção do jogador para o terreno, isto é, para a quantidade máxima de sementes de cada tipo (milho e trigo) que poderia ser plantada. Além disso, essa dica elimina de vez a opção baseada em OU, pois mesmo que ainda haja espaço no terreno para plantio, o laço não pode continuar se não houver sementes, ou seja, a semente em menor quantidade limita a plantação. Por fim, a Dica 5 conduz o jogador explicitamente ao raciocínio correto, indicando que a condição adequada deve garantir simultaneamente três requisitos: ainda há milho e ainda há trigo e ainda há área livre para plantar.

Para resolver esse desafio, espera-se que o jogador seja capaz de interpretar operadores relacionais e lógicos, em especial a diferença entre usar || (OU) e usar o operador lógico *&&* (E), que exige que todas as condições da expressão sejam verdadeiras ao mesmo tempo para que o corpo do laço seja executado. Além disso, o jogador precisa raciocinar sobre o consumo de recursos limitados (estoque de sementes de milho e de trigo) e acompanhar a evolução dos contadores durante a execução do laço, entendendo que a iteração deve parar assim que uma das sementes acabar ou quando o campo atingir sua capacidade máxima.

A Figura 5.7 apresenta o Desafio 5, cujo objetivo é preencher uma matriz 10×10 com três tipos de flores: brancas, amarelas e vermelhas. As brancas ocupam a diagonal principal, as amarelas as posições acima da diagonal, e as vermelhas as demais posições. Nesse desafio, o jogador deve completar dois campos relativos às condições de parada de dois laços *for* (externo e interno). Em seguida, ele deve selecionar as expressões relacionais nos três condicionais que realizam o plantio nas regiões corretas, isto é, escolher entre relações como $i == j$, $i < j$ e $i > j$ conforme o enunciado.

Figura 5.7: Tela do Desafio 5 do jogo *Infinity Loop*. Adaptado de Prokopenko e Oliveira (2024).

Para o Desafio 5, foram elaboradas quatro dicas que orientam a definição das condições de parada do laço considerando a indexação da matriz iniciando em zero. Essas dicas guiam a escolha das expressões relacionais entre i e j tendo a diagonal principal como referência para particionar a matriz em duas regiões: triângulo superior ($i < j$) e triângulo inferior ($i > j$). Essas dicas estão apresentadas na Tabela 5.5.

Tabela 5.5: Dicas para o Desafio 5.

Dica	**Descrição**	**Opção incorreta**	**Opção correta**
1	Note que os iteradores dos laços iniciam em 0. O primeiro laço irá percorrer as linhas da matriz e o segundo laço irá percorrer as colunas.	1º *for*: $i \leq 10$ 2º *for*: $j \leq 10$	1º *for*: $i < 10$ 2º *for*: $j < 10$
2	Para todos os valores da diagonal principal, o número da linha coincide com o da coluna.	$i < j$, $i > j$	$i == j$
3	Todos os valores acima da diagonal principal possuem a linha menor que a coluna.	$i > j$, $i == j$	$i < j$
4	Todos os valores abaixo da diagonal principal possuem a linha maior que a coluna.	$i < j$, $i == j$	$i > j$

As dicas deste desafio orientam o jogador a configurar adequadamente os dois laços *for* (interno e externo) e a escolher, com base em propriedades da matriz, as relações entre i e j a serem usadas em cada uma das três instruções *if*. A primeira dica refere-se à escolha das condições de parada dos laços, e as demais estão relacionadas à escolha das relações entre i e j em cada condicional.

A Dica 1 ressalta que os iteradores dos dois laços (i e j) iniciam com valor 0 (zero) e que, por se tratar de uma matriz 10×10, o mesmo operador relacional deve ser usado em ambos os laços; nesse caso, < 10 mantém a iteração dentro dos limites dos índices de linha e coluna da matriz. Na Dica 2 espera-se que o jogador saiba o conceito de diagonal principal. Além disso, essa dica apresenta a propriedade central da diagonal principal, na qual os índices de linha e coluna são iguais. Por fim, as dicas 3 e 4 caracterizam a divisão da matriz em duas partes excluindo a diagonal principal. A condição $i < j$ concentra os elementos acima da diagonal (partição superior), e $i > j$ concentra os elementos abaixo dela (partição inferior).

Essas dicas pressupõem a decomposição da matriz em três regiões: diagonal principal (*i == j*), posições acima da diagonal (*i < j*) e abaixo da diagonal (*i > j*), o que orienta a escolha da relação entre *i* e *j* nos *if's* para plantar flores brancas, amarelas e vermelhas em seus respectivos lugares. Além disso, o jogador deve relacionar o início dos laços em 0 (zero) com os limites da matriz para definir o mesmo operador de comparação nos dois *for*, pois se trata de uma matriz quadrada 10×10. Se usar *<= 10*, o iterador tenta acessar o índice 10 e extrapola os limites válidos, que vão de 0 (zero) a 9 (nove). Se o laço interno *for* tiver a condição de parada *i < 10* em vez de *j < 10*, o valor de *j* pode não atingir o limite esperado e o programa pode entrar em laço infinito. Para concluir o desafio, é necessário compreender operadores relacionais e o papel da diagonal principal, que particiona a matriz em dois triângulos, garantindo que cada condição trate uma região distinta da matriz e que o plantio ocorra da forma correta.

5.2 CONSIDERAÇÕES FINAIS

Os resultados evidenciam a integração do personagem tutor, do sistema de dicas progressivas e do botão com temporizador ao *Infinity Loop*, além da documentação das dicas por desafio em tabelas que relacionam descrições, inferências e alternativas corretas e incorretas. Dessa forma, os recursos adicionados buscam orientar o raciocínio do jogador durante a resolução de problemas envolvendo laços de repetição, reduzindo solicitações consecutivas de dicas e incentivando maior reflexão entre as interações.

Apesar disso, é necessário avaliar com potenciais usuários e especialistas em ensino de programação e jogos educacionais se as dicas são de fácil entendimento e se o tempo entre solicitações é adequado ao contexto de cada desafio. Como limitação, o trabalho não propõe uma estratégia direta e específica para combater tentativas sucessivas e aleatórias de resolução, restringindo-se a desestimular esse comportamento de forma indireta por meio do sistema de dicas e do temporizador.

6 CONCLUSÃO

Este trabalho teve como objetivo incorporar estratégias para mitigar o comportamento de *gaming the system* no jogo educacional *Infinity Loop*. Sua principal contribuição consiste na incorporação, ao jogo, de um sistema de dicas que não revela diretamente a resposta do problema, integrado a um botão com temporizador que restringe e bloqueia temporariamente novas solicitações, definindo um intervalo mínimo (ex.: 10 segundos) entre a exibição de dicas. Essa solução busca desencorajar tentativas aleatórias de resolução dos desafios e pedidos sucessivos de dicas, incentivando interações mais reflexivas por parte do estudante.

Adicionalmente, foram realizadas melhorias na interface do jogo com o intuito de tornar a experiência do usuário mais agradável. A partir de ajustes nas configurações da Godot Engine, foi possível aumentar a resolução e a nitidez do cenário do jogo. Além disso, as caixas de diálogo também foram redesenhadas para exibir o avatar de cada NPC, a fim de tornar a comunicação mais personalizada e a experiência mais envolvente.

Como trabalho futuro, sugere-se a realização de testes com estudantes iniciantes em programação para validar as dicas e os intervalos de tempo definidos, bem como coletar dados de interação que permitam caracterizar com mais precisão o comportamento do aluno no jogo. Também se mostra importante realizar uma avaliação com especialistas em ensino de programação e em jogos educacionais, a fim de analisar a clareza, a qualidade pedagógica e a adequação das dicas aos objetivos de aprendizagem, bem como o tempo entre as dicas. Por fim, recomenda-se associar as dicas de acordo com a resposta do jogador, de modo a oferecer orientações mais específicas e contextualizadas com o problema que ele está lidando.

REFERÊNCIAS

Araújo, A., Filho, D. L. Z., Oliveira, E. H. T., ao de Carvalho, L. S. G., Pereira, F. D. e Oliveira, D. B. F. (2021). Mapeamento e análise empírica de misconceptions comuns em avaliações de introdução à programação. Em *Anais do I Simpósio Brasileiro de Educação em Computação (EduComp 2021)*, páginas 123–131, Porto Alegre – RS, Brasil. Sociedade Brasileira de Computação (SBC).

Baker, R. S., Corbett, A. T., Koedinger, K. R. e Wagner, A. Z. (2004). *Off-task behavior in the cognitive tutor classroom: when students "game the system"*, páginas 383–390. ACM, New York.

Baker, R. S., Richey, J. E., Zhang, J., Karumbaiah, S., Andres-Bray, J. M., Nguyen, H. A., Andres, J. M. A. L. e McLaren, B. M. (2024). Gaming the system mediates the relationship between gender and learning outcomes in a digital learning game. *Instructional Science*, 53(2):203–238.

Baker, R. S., Walonoski, J. e Heffernan, N. (2008). Why students engage in "gaming the system" behavior in interactive learning environments. *Journal of Interactive Learning Research*, 19(2):185–224.

de Oliveira, L. R. E., Valli, A. M. P., Boeres, M. C. S. e Catabriga, L. (2023). Robotim: um jogo educacional multidisciplinar com análise de dados em tempo real. Em *Anais do XXXIV Simpósio Brasileiro de Informática na Educação (SBIE 2023)*, páginas 560–571, Passo Fundo – RS, Brasil. Sociedade Brasileira de Computação (SBC).

Honda, F., Melo, R., Pires, F. e Pessoa, M. (2023). RobotCode: um jogo educacional para auxiliar na aprendizagem de lógica de programação. Em *Anais do Laboratório de Ideias - Simpósio Brasileiro de Educação em Computação (EduComp 2023)*, páginas 32–33, Porto Alegre – RS, Brasil. Sociedade Brasileira de Computação (SBC).

Nunes, T. M. e Jaques, P. A. (2014). Utilizando agentes pedagógicos animados como uma abordagem não restritiva ao Gaming The System. *Revista Brasileira de Informática na Educação*, 22(1):147–163.

Prokopenko, H. e Oliveira, J. P. K. (2024). Infinity Loop: um jogo educacional sobre erros causados por misconceptions no ensino de programação. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação), Universidade Federal do Paraná.

Rocha, H. J. B., de B. Costa, E. e de A. R. Tedesco, P. C. (2023). Analisando a relação de conceitos de programação com o comportamento "abuso do sistema" em programadores novatos. Em *Anais do XXXIV Simpósio Brasileiro de Informática na Educação (SBIE 2023)*, páginas 1716–1725, Passo Fundo – RS, Brasil. Sociedade Brasileira de Computação (SBC).

Santos, A. V. e Ferreira, A. B. (2021). Formigas em grafo: um jogo educacional para apoio ao ensino e aprendizagem dos algoritmos de busca em largura e busca em profundidade. Em *Anais da 9ª Escola Regional de Computação do Ceará, Maranhão e Piauí (ERCEMAPI 2021)*, páginas 59–66, Quixadá – CE, Brasil. Sociedade Brasileira de Computação (SBC).