CBIE 2023
CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO
Uma escola para o futuro: Tecnologia e conectividade a serviço da educação

Variant Scape: um jogo para exercitar conceitos de introdução à lógica de programação

Lucas Araújo[1], Jeniffer Macena[2], Rafaela Melo[2], Marcela Pessoa[1], Fernanda Pires[1]

[1]Escola Superior de Tecnologia – Universidade do Estado do Amazonas (EST/UEA)

[2]Instituto de Computação – Universidade Federal do Amazonas (ICOMP - UFAM)

{lsda.lic16,msppessoa,fpires}@uea.edu.br

{jeniffer.souza,rmelo}@icomp.ufam.edu.br

Abstract. *Learning to program is complex, requiring a high grade of abstraction, as evidenced by the high failure and dropout rates in Computing courses. This article presents "Variant Scape", a game that aims to promote the development of programming logic and Computational Thinking. The game uses the abstraction of programming rules in its mechanics in which the player must use variables, conditional and repetition structures, in addition to proposing activities with basic mathematics operations (addition, subtraction, multiplication and division) during gameplay. Preliminary results showed that the game had a good acceptance in terms of design and gameplay, but the need for feedback adjustments was raised by adding elements to the game's interface.*

Resumo. *Aprender a programar é algo complexo, que exige alto grau de abstração, o que se comprova pelas altas taxas de reprovação e evasão nos cursos de Computação. Este artigo apresenta "Variant Scape", um jogo que tem por objetivo promover o desenvolvimento da lógica de programação e do Pensamento Computacional. O jogo utiliza da abstração das regras da programação em sua mecânica em que o jogador deve utilizar variáveis, estruturas condicionais e de repetição, além de propor atividades com as operações de Matemática básica (soma, subtração, multiplicação e divisão) durante a gameplay. Resultados preliminares mostraram que o jogo teve uma boa aceitação em questões como design e gameplay, porém foi levantada a necessidade de ajustes em feedbacks através da adição de elementos na interface do jogo.*

1. Introdução

No primeiro período dos cursos de graduação em computação, os estudantes têm seu primeiro contato com a disciplina de introdução à programação. Existem diversas abordagens e métodos que são utilizadas pelos professores, mas o fato é que muitos estudantes possuem dificuldade no processo de aprendizagem, chegando a desistir do curso devido à frustração de não conseguir um bom desempenho [Gomes and Mendes 2007, Pires et al. 2021, Franzoia et al. 2019].

Existem dois atores no processo de aprendizagem da lógica de programação, as estruturas educacionais (professores, mediadores, matriz curricular, entre outros) e os estudantes, ambos enfrentam desafios que podem dificultar o processo de aprendizagem

do público-alvo. Os professores têm como desafios adequar suas estratégias de compartilhamento de informações de acordo com o perfil dos estudantes, e nem sempre estão preparados para isso, há necessidade de tempo e técnicas para realizar nivelamento do conhecimento dos estudantes e pode existir uma preocupação maior com a aprendizagem de uma linguagem de programação do que com o desenvolvimento da lógica de programação [Sarpong et al. 2013, Qian and Lehman 2017, Lahtinen et al. 2005]. Em contrapartida, os estudantes também possuem dificuldades na aprendizagem de lógica de programação, pois é um processo que exige algumas *skills* como: memorização, resolução de problema, abstração e pensamento lógico, tais habilidades precisam ser exercitadas, caso contrário, os alunos não terão um bom desempenho na disciplina, como também na carreira de programador [Cheah 2020, Robins 2019, Piteira and Costa 2013].

O Pensamento Computacional (PC) é um dos conceitos que vêm sendo divulgado mundialmente e aplicado em escolas e universidades para ajudar os estudantes no desenvolvimento do raciocínio lógico para resolução de problemas [Wing 2011]. A BBC (*British Broadcasting Corporation*) [2018] define o PC em quatro pilares principais: decomposição, reconhecimento de padrões, abstração e algoritmo. Cada um dos pilares possui características específicas, que ajudam a desenvolver habilidades consideradas fundamentais para o bom desenvolvimento e desempenho do cidadão moderno.

Tendo em vista as dificuldades encontradas pelos estudantes na aprendizagem de lógica de programação, professores e pesquisadores têm buscado formas de mitigar esse problema. Uma das alternativas é o uso de jogos educacionais, pois têm o potencial de divertir e motivar o estudante, além de incentivar a persistência para vencer os desafios impostos [Tarouco et al. 2004, Pires et al. 2020, Plass et al. 2015]. Estudos na área de jogos educacionais afirmam que jogos podem ser utilizados para ajudar na aprendizagem de diversos assuntos, desde o currículo da educação básico ao superior [Pires et al. 2018, Pires et al. 2021, Honda et al. 2023].

Portanto, visando auxiliar na aprendizagem de lógica de programação nos períodos iniciais dos cursos de computação, e exercitar o pensamento computacional, este trabalho apresenta a proposta de um jogo digital educacional intitulado “Variant Scape”. No jogo, o estudante se vê preso em uma simulação virtual e para sair é necessário resolver vários *puzzles* espalhados pelo mapa e avançar até a saída. Os desafios apresentados no jogo envolvem a utilização de conceitos de variáveis, estruturas condicionais, estruturas de repetição e funções. O artigo estrutura-se da seguinte forma: a Seção 2 aborda os trabalhos relacionados e a fundamentação teórica do trabalho; a Seção 3 mostra o processo de desenvolvimento e concepção, além da descrição das mecânicas presentes no jogo;na Seção 4 está descrito o Design Experimental utilizado para o jogo; na Seção 5 os resultados deste trabalho; e, por fim, na Seção 6 estão as considerações finais.

2. Trabalhos Relacionados e fundamentação teórica

Esta seção apresenta os trabalhos correlatos à pesquisa apresentada neste artigo e os fundamentos de como se dá a aprendizagem por meio de jogos.

2.1. Trabalhos Relacionados

Netto et al. [2017] desenvolveram um jogo chamado “Game Logic” para ser utilizado por crianças e jovens para trabalhar lógica de programação, onde são utilizados blocos

de comandos para controlar o personagem principal, realizar suas atividades, andar pelo mapa, resolver problemas e interagir com o ambiente. O jogo foi desenvolvido para dispositivos Android, para que os estudantes pudessem ter acesso a qualquer momento. Após a aplicação com um grupo de estudantes, eles deram um *feedback* positivo, com uma variação de notas de avaliação entre 8 e 10 para a qualidade do jogo.

No trabalho de Macena et al. [2022] é apresentado o desenvolvimento do jogo "Hello Food", cujo objetivo é proporcionar a aprendizagem de algoritmos, essencialmente os conceitos de estruturas condicionais, laços de repetição e vetores. O jogo foi desenvolvido para dispositivos móveis. O jogador deve trabalhar em um restaurante e utilizar a lógica para resolver desafios que são propostos durante o jogo, como ordenar valores, seguir algoritmos e controlar laços de repetição. Após testes realizados com estudantes do ensino médio, técnico e superior, o jogo obteve boa aceitação dos jogadores.

Silva et al. [2021] descrevem o processo de desenvolvimento do jogo "ProgramSE", voltado para o público de estudantes iniciantes do ensino superior, que abrange os conceitos iniciais e intermediários de lógica de programação. O jogo possui a mecânica *point and click*, onde o jogador utiliza o clique do mouse para interagir com os elementos do jogo, ele é baseado no estilo *escape room*. Os resultados da aplicação mostraram uma boa aceitação dos estudantes em aspectos como: história do jogo, desafio, mecânicas e jogabilidade, além de atingir seu objetivo principal que era estimular o raciocínio lógico.

O jogo apresentado neste trabalho utiliza de um estilo diferente dos apresentados nesta seção, ele usa da mecânica de jogos de plataformas 2D onde a aventura acontece, além disso busca tratar de forma equilibrada os conceitos de lógica de programação, baseados em desafios matemáticos e lógicos para avançar no jogo. Outra diferença entre alguns dos jogos citados está no fator algoritmico, pois *Variant Scape* não utiliza explicitamente a criação de algoritmos para controlar o personagem, mas ele deve criar mentalmente uma sequência lógica para resolver os problemas e atender aos pedidos enviados a ele, como será visto a seguir.

2.2. Aprendizagem em jogos

A digitalização do mundo mudou a forma com os estudantes recebem informação, como interagem com ela, além de ter modificado o modo como eles aprendem, portanto é necessário que as metodologias educacionais acompanhem essas mudanças [Zhonggen et al. 2019, Plass et al. 2015]. Os jogos são ferramentas que proporcionam estímulos, visuais e auditivos, se usados atrelados a fatores educacionais podem facilitar o processo de aprendizagem, causando motivação e engajamento para aprender, seja de forma implícita ou explícita.

Os jogos permitem aos seus jogadores entrarem em mundos diferentes da sua realidade, viverem outras experiências, descobrirem e conhecerem coisas novas. De forma semelhante, os jogos sérios também podem proporcionar tal experiência além do entretenimento. Eles podem promover a literacia digital, desenvolvimento cognitivo, estimular a tomada de decisão e as habilidades de resolução de problemas, criar ambientes saudáveis de competição e promove a autonomia do estudante [De Gloria et al. 2014, Anastasiadis et al. 2018, Plass et al. 2015].

"Variant Scape" foi projetado para ajudar a mitigar os problemas que algumas pessoas podem possuir com relação à abstração, em que muitas vezes imaginar uma

estrutura condicional ou de repetição representada apenas por palavras não é suficiente para aprender como elas funcionam, sendo necessário algum meio visual para auxiliar na compressão de tais estruturas [Hoed 2016, Saccaro et al. 2019, Honda et al. 2023]. Os recursos visuais que os jogos proporcionam podem transformar as estruturas utilizadas nas linguagens de programação, em mecânicas, *pop-ups*, dispositivos e fases, permitindo a ludificação da aprendizagem e ajudando o jogador na visualização de conceitos de programação de forma mais concreta[Macena et al. 2020, Honda et al. 2022].

Em "Variant Scape", são cruzadas as definições de aprendizagem significativa com os pilares do Pensamento Computacional para auxiliar no processo de aprendizagem dos jogadores. No jogo são aplicados os conceitos mais comuns abordados com iniciantes em programação, como variáveis, a atribuição de uma operação aritmética à uma variável, estruturas de repetição e condicionais, funções e declaração de variáveis, para que os jogadores sintam-se familiarizados com esses conceitos ao lidarem com códigos reais.

3. O jogo "Variant Scape"

"Variant Scape" é um jogo educacional idealizado para auxiliar estudantes iniciantes nos cursos de computação, e tem como objetivo ajudar na compreensão de conceitos de introdução à lógica de programação e exercitar o pensamento computacional. Além disso, o jogo também foi arquitetado para abordar problemas do campo da matemática básica, como as quatro operações e comparação de grandezas. O jogo foi idealizado com base nos conceitos do pensamento computacional, com a *gameplay* baseada no gênero *escape room*, onde o jogador deve resolver *puzzles* para avançar nas fases e vencer o desafio.

3.1. Desenvolvimento do Jogo

O desenvolvimento do jogo foi realizado seguindo etapas baseadas no modelo de desenvolvimento criado por Pires [2021], e foram resumidas nesta seção. Cada etapa tem como resultado artefatos que, após validação, seguem para a próxima fase. Cada etapa e seus artefatos estão descritos a seguir:

Brainstorming: a partir da definição de que seria introdução à lógica de programação, foram escolhidos os principais conceitos que deveriam ser tratados no jogo, para isso, definiu-se um esquema didático que permitiu cruzar entre todas as fases do jogo três aspectos: operações matemáticas, tópico do tema e os assuntos específicos que deveriam ser abordados. O esquema pode ser visto na Tabela 1.

Levantamento de requisitos e planejamento: onde se estabeleceu a proposta conceitual do jogo. Foram levantados pontos sobre como os assuntos poderiam ser abordados e como tornar menos abstratos certos conceitos da programação. Nessa fase foi pensado na escolha do estilo de *gampeplay*, mecânicas, descrição de todo o *level design* e cruzamento com o conteúdo didático.[1]

Prototipação do jogo: foram colocadas em prática as requisições da etapa anterior. O estilo escolhido para o jogo foi baseado nos jogos de plataforma e *escape room*, onde o personagem deve avançar pelo mapa interagindo com os elementos espalhados pela área, desviar de inimigos e ficar atento às plataformas móveis. A peça chave para que a

[1]Essa etapa considera pesquisas de trabalhos correlatos em diferentes áreas do conhecimento, inspirações podem vir de filmes, livros, mangás, séries, etc.

Tabela 1. Esquema didático das fases do jogo.

Fase	Operação Matemática	Tópico	Assuntos
1	Soma	Tipos de Dados e Variáveis	Inteiros, Declaração de Variáveis e Atribuição de Valor
2	Soma e Subtração	Variáveis e Estruturas Condicionais	Inteiros, Booleanos, Declaração de Variáveis, Comparação de Grandezas
3	Multiplicação	Variáveis e Estruturas Condicionais	Inteiros, Char e Declaração de Variáveis, Atribuição de Valores e Igualdade
4	Divisão e Soma	Variáveis, Estruturas de Repetição e Condicionais	Inteiros, Literais, Igualdade, Booleanos
5	Multiplicação e Divisão	Variáveis, Estruturas Condicionais e Funções	Literais, Booleanos, Char, Igualdade, Comparação de Grandezas, Variáveis Auxiliares

gameplay aconteça é a interação do personagem com as variáveis existentes no mundo do jogo e com dispositivos como os "Criadores de Variáveis". Os criadores de variáveis são dispositivos que permitem a interação entre duas variáveis, no conceito de programação representam diretamente a atribuição do resultado de uma operação matemática à uma nova variável, como por exemplo na linguagem de programação *Python*, que pode ser representada da seguinte forma SOMA = A + B. Além dos criadores de variáveis, foram idealizados dispositivos como os Copiadores e os Divisores de variáveis, que geram uma série de variáveis baseadas nos parâmetros fornecidos a elas.

Na Figura 1 é possível ver os primeiros conceitos dos Criadores de Variáveis, do Divisor de Variáveis, cujas mecânicas serão descritas na seção de mecânicas, e ao centro das ilustrações está um esboço de uma das fases.

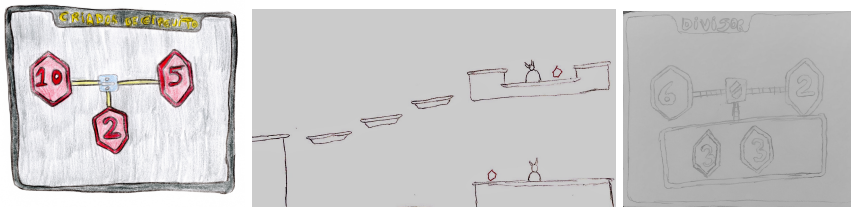

Figura 1. Criador de variáveis, trecho de uma das fases do jogo e o esboço do divisor de variáveis.

Semelhante aos Criadores de Variáveis, também foi criado o "Verificador de Variáveis", que pode realizar comparações entre o valor de variável e o valor fixo do verificador, tal qual ocorre em programas de computador, por exemplo para verificação de média escolar onde temos: MEDIA $>$ 8, que retorna Verdadeiro ou Falso. Todos esses dispositivos foram pensados para diminuir o esforço de abstração que ocorre quando se está escrevendo um código. Para algumas pessoas não é claro apenas escrever aquelas operações na IDE (vem da sigla em inglês para Ambiente de Desenvolvimento Integrado), por esta razão houve a necessidade de ilustrar essas estruturas no jogo.

Validação do protótipo: antes que o jogo possa ser desenvolvido em uma *engine* e ser executado em algum dispositivo, ele foi testado em um modelo no Figma e precisou passar por testes. Os testes escolhidos para serem aplicados foram: emoti-SAM e MEEGA+, o primeiro teste busca avaliar a afeição e sentimentos do jogador para com o

jogo, o segundo teste foi utilizado para verificar os aspectos educacionais do jogo. Os resultados mostraram uma recepção positiva por parte dos jogadores e proporcionaram *feedbacks* para melhoria do jogo.

Desenvolvimento do protótipo de alta fidelidade: é a etapa atual do desenvolvimento deste jogo, a *engine* escolhida para construí-lo, aplicando todas as mudanças e melhorias sugeridas durante a fase de testes.

3.2. História

Alan é um jovem que precisa pagar algumas dívidas, buscando uma maneira "fácil" de conseguir dinheiro, ele se inscreveu no projeto "Variant Scape", onde será submetido a uma simulação em sua mente, um mundo virtual, onde terá que passar por alguns desafios, testando sua capacidade mental. Ao entrar no universo simulado, ele se encontra em Analog (Figura 2), uma cidade virtual onde ele será guiado por Ada, uma assistente que em determinados momentos do desafio diz quais são as tarefas que devem ser realizadas em cada fase. Resolver as tarefas permitirá que Alan avance e chegue ao seu objetivo. Após chegar no final do desafio ele finalmente está livre e consegue sua recompensa.

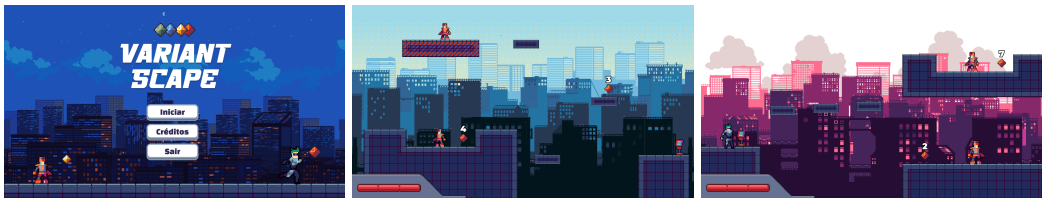

Figura 2. Telas do jogo: menu inicial e fases 3 e 4.

3.3. Mecânicas do jogo e de aprendizagem

Com os conceitos do jogo definidos, foram idealizadas de forma mais aprofundada, as mecânicas do jogo, como o personagem deve interagir com os itens, qual fluxo deve ser seguido, como representar graficamente as estruturas utilizadas em um código real. Toda a mecânica e fluxo foi pensado como um código de um programa, onde devem-se criar operações para chegar a um valor, entrar em *loops*, entre outros conceitos. A definição da mecânica seguiu os preceitos de uma sequência didática em formato espiral, ou seja o *level design* do jogo está de acordo com a ordem em que habilidades de programação são usualmente desenvolvidas, seguindo as regras de conhecimento prévio necessário.

Variáveis: em programação, uma das protagonistas dos códigos são as variáveis, por meio delas acontece a maioria das interações. Em "Variant Scape", as variáveis são peças-chave para avançar nos desafios. Alan é capaz de armazenar apenas uma variável de um determinado tipo por vez, ao se aproximar de uma variável o jogador pode pressionar o botão C do teclado e guardar o valor da variável que está interagindo, podendo armazená-lo em um dispositivo do jogo, ou apenas devolvendo ao local que desejar na fase pressionando o botão V no teclado. No jogo o personagem deve interagir com determinados tipos de variáveis, conforme apresentado na Tabela 2.

Criador de variáveis: são dispositivos que possuem a função de gerar novas variáveis a partir da operação matemática de duas variáveis de entrada, ou seja, é necessário que o jogador insira duas variáveis para que seja criada uma nova, as duas variáveis são utilizadas

Tabela 2. Tipos de Variáveis.

Variável	Descrição
	Variáveis do tipo inteiro armazenam números que não possuem casas decimais, são o valor que representam em sua totalidade.
	Variáveis do tipo *float* armazenam números que possuem casas decimais, neste jogo as variáveis do tipo *float* apresentam no máximo uma casa decimal.
	Variáveis do tipo booleano podem apresentar no máximo dois valores diferentes, são eles: Verdadeiro ou Falso.
	Variáveis do tipo *char* não recebem nenhum valor numérico, apenas símbolos, como letras, porém guardam apenas uma símbolo por vez.
	São variáveis que armazenam vários *chars* encadeados, mas para este jogo elas armazenarão até 5 *chars*.

na operação e o resultado é guardado com o personagem principal. Existem Criadores de Variáveis que realizam operações diferentes, que são: soma, subtração, divisão e multiplicação.

Figura 3. Criadores de Variáveis.

Verificadores de Variáveis: os Verificadores de Variáveis são dispositivos para realizar testes de grandeza ou igualdade, verificando se uma constante é maior, menor ou igual a uma variável, ele gera apenas uma variável do tipo booleana. Diferente dos criadores, os Verificadores possuem apenas uma entrada para variáveis, que será comparada a uma constante, cada Verificador durante o jogo pode apresentar uma constante própria. Na Figura 4 a primeira e segunda ilustração mostram os Verificadores de Variáveis, que representam as estruturas condicionais, o resultado da operação sendo verdadeiro ou falso.

Copiador de Variáveis: o Copiador de Variáveis (Figura 4) recebe duas variáveis como entrada, porém cada uma tem uma função diferente. A primeira entrada do copiador indica a quantidade de cópias que serão geradas, a segunda entrada é o valor que as cópias receberão, por exemplo: a primeira variável possui valor 3 e a segunda possui valor 8, serão então geradas três variáveis com o valor 8.

Decompositor de Variáveis: os Decompositores de Variáveis recebem duas variáveis como entrada. A primeira entrada indica o valor a ser dividido entre as novas variáveis geradas, a segunda entrada indica quantas vezes o valor da primeira variável será dividida, como por exemplo: a primeira variável possui o valor 6 e a segunda variável 2, serão geradas duas variáveis com o valor 3.

3.4. A aprendizagem e o Pensamento Computacional em "Variant Scape"

Considerando a natureza do jogo e dos objetivos de aprendizagem, optou-se por usar a teoria da Aprendizagem Significativa. Na aprendizagem, a peça mais importante para que o aprender aconteça, é o conhecimento prévio do estudante, mesmo que não específico [Ausubel et al. 1980]. Ao receber uma nova informação, o cérebro procura em seu vasto

Figura 4. Verificador e Copiador de Variáveis

catálogo de saberes algo que possa ajudá-lo a compreender a informação que se apresenta, como ele pode classificar esse conhecimento, se já existe algo parecido com a nova informação para que seja armazenada. Os responsáveis por essa tarefa são os subsunçores ou âncoras de aprendizagem. Pensando nas funções dos subsunçores, supõe-se que, ao expor os estudantes previamente a mecânicas e atividades que sejam análogas às desempenhadas durante a programação, pode diminuir o esforço necessário para a compreensão, pois eles poderão associar aquela nova experiência a algo que eles já vivenciaram no jogo.

Dessa forma, todo o processo de mecânica e *level design* do jogo foi pensado abstraindo-se as regras e processos de funcionamento estudados em introdução à programação e aplicados em um cenário de jogo. Essas ações podem ser chamadas de subsunçores, ou seja, podem ser classificadas como âncoras para aprendizagem, permitindo que os estudantes conectem o que já sabem com as ações executadas por si no jogo. De forma paralela, durante toda a *gameplay*, o jogador é estimulado a exercitar os quatro pilares do pensamento computacional. A seguir é explicado como cada pilar do PC se relaciona com as mecânicas do jogo.

- **Decomposição:** é a capacidade de dividir um problema complexo em problemas menores, facilitando a sua resolução, sabendo que só pode levar uma variável por vez, o jogador precisa passar a particionar o seu trabalho geral, que é criar uma nova variável, planejando pequenas tarefas.
- **Reconhecimento de padrões:** serve para perceber aspectos repetitivos de um problema e também ser capaz de utilizar métodos de resolução de problemas antigos e aplicá-los em novos, assim, ainda que os tipos de criadores e verificadores mudem, eles utilizam mecânicas semelhantes, permitindo ao jogador reutilizar estratégias que foram criadas em fases ou desafios anteriores.
- **Abstração:** é responsável por filtrar apenas as informações relevantes para a resolução do problema, o jogador é levado a interpretar que os cristais com valores são representações das variáveis, que cada uma possui um tipo, e que elas podem ser utilizadas em diversos contextos nos algoritmos.
- **Algoritmo:** consiste na capacidade organizar uma sequência de ações que resultem na resolução. Ada sempre dá uma missão para o jogador em cada etapa da fase, a partir desse momento cabe ao jogador criar uma sequência de passos que o permita pegar as variáveis que deseja para serem usadas nos dispositivos do jogo.

4. Design Experimental

O protótipo de média fidelidade utilizado para testar o jogo, desenvolvido no *Figma*, utilizou as mecânicas de operações matemáticas para resolver os *puzzles*, essa medida foi tomada para testar a *gameplay* do jogo e observar como os jogadores interagem com o estilo proposto e o nível de compreensão e aceitação da ferramenta para aprendizagem.

Para validar o protótipo foram realizados testes com cinco pessoas do curso de Licenciatura em Computação, que foram escolhidos por uma seleção aleatória de estudantes que estavam dispostos a participar de testes de software. Participaram do testes uma iniciante, três entre o 5º e 7º período e uma egressa, todos do curso de licenciatura em computação. A participação de estudantes mais avançados ajudou a perceber outros aspectos que poderiam evidenciar mais a lógica de programação, já que eles possuem mais experiência e alguns iniciantes ainda não possuem com tanta clareza esse conceito em comparação aos avançados. Os testes aplicados para validação e captura de dados foram o Emoti-SAM [Hayashi et al. 2016], para registrar questões como emoção e sentimento dos jogadores. O segundo teste foi o MEEGA+ [Petri et al. 2019], onde foi construído um formulário com 35 questões que avalia aspectos como: desafio, confiança, acessibilidade e aprendizibilidade. Durante os testes foi disponibilizado para os participantes um computador, onde ele deveria interagir com o jogo, dizendo quais ações gostaria de executar. Ao receber as instruções, o aplicador do teste fazia modificações na interface do protótipo no *Figma*, movimentando o personagem, abrindo *pop-ups*, movendo paredes e outros elementos, agindo como um "motor" e simulando a *gameplay* de um jogo real.

A avaliação dos participantes em relação ao jogo foi feita de forma presencial, no mesmo dia e por haver apenas um aplicador de testes e um computador, cada um esperou seu momento para realizar o teste, os jogadores levaram cerca de 1h30m a 1h50m para finalizar as cinco fases do protótipo do jogo. Após a *gameplay*, foi disponibilizado um *link* com um formulário on-line para que eles avaliassem o jogo. Antes de responder o formulário, os participantes tiveram que assinalar um Termo de Consentimento Livre para utilizar os dados coletados na pesquisa, nos formulários estavam integrados os modelos do emoti-SAM e MEEGA+ adaptados para os fins da pesquisa, também. Além dos formulários foi levado em consideração, também para a avaliação, o *feedback* oral dado durante a utilização do jogo, que foi registrado por meio de gravações de áudio, sob o consentimento dos participantes.

5. Resultados e discussão

Os resultados do teste emoti-SAM apontaram que cerca de 80% dos jogadores ficaram felizes por terem jogado e ficaram animados enquanto jogavam. O teste MEEGA+ foi utilizado para realizar a avaliação do jogo, onde foi possível capturar dados importantes para melhorar a experiência do jogador e aprendizagem, que é o principal objetivo.

Com o *feedback* do MEEGA+ observou-se que o principal aspecto do jogo, que são as variáveis, teve seu destaque percebido pelos jogadores, que conseguiram entender a sua função e como podem ser utilizadas nos algoritmos. Porém verificou-se que há a necessidade de deixar mais claro aspectos como: *feedback* de acerto ou erro; deixar sinalizado qual o tipo de operação que será realizada antes de interagir com o Criador de Variável; e deixar à mostra qual o valor da variável que deve ser criado, pois só é avisado no momento do diálogo com a Ada. Houve também a ressalva da necessidade de mais fases. Atualmente, o jogo possui apenas cinco fases, e os jogadores indicaram que o "Variant Scape" pode ter mais fases, onde podem ser abordadas mais tipos de variáveis, outras estruturas e também se aprofundar mais nos assuntos de lógica de programação. Essa indicação mostra que embora os estudantes tenha tido dificuldade de compreensão do jogo, a princípio, se sentiram motivados a jogá-lo, como indica a literatura sobre o uso de jogos [Pires et al. 2021, Plass et al. 2015, Honda et al. 2023].

Nas gravações realizadas durante os testes, há registros da dificuldade que os jogadores tiveram, principalmente com as operações matemáticas, pois se confundiam com as ordens dos fatores da operação e o tipo da variável, isso levantou o questionamento se os problemas que alguns estudantes têm com a lógica de programação estão intrinsecamente ligados à problemas com a matemática. No decorrer da *gameplay* os jogadores se habituaram com as missões que Ada fornecia, porém houve uma missão em que foi percebida uma necessidade de "parar pra pensar" o que deveria ser feito, foi na parte final da fase 4, onde Alan deveria armazenar os valor das variáveis nas plataformas e depositar até que uma certa condição fosse atendida por todas as caixas, podendo ser representada em uma linguagem de programação real como *Python*, porém muitos tiveram dificuldade em entender o que a condicional da primeira e da segunda caixa estavam exigindo, como mostra a Figura 5.

```
while caixa1 < 15 and caixa2 < 14 and
caixa3 != "casa":
        coleteVariaveis()
```

Figura 5. Trecho da Fase 4 e pseudocódigo da fase.

O jogo está em fase de desenvolvimento, ainda não foi implementado em uma *engine*, estando em versão de protótipo digital. Com os resultados do protótipo, foi possível listar melhorias significativas que farão com o que o jogo implementado tenha um melhor desempenho em relação ao seu objetivo, que é auxiliar na aprendizagem e desenvolvimento da lógica de programação.

6. Considerações Finais

Este trabalho apresentou a proposta do jogo educacional "Variant Scape", que tem por objetivo auxiliar na compreensão de conceitos relacionados à lógica de programação. Cada etapa do desenvolvimento do jogo, desde sua concepção até a prototipação, foi pensada para que o jogo ficasse bem estruturado e bem fundamentado, tanto em questões de desenvolvimento, quanto de aprendizagem.

O protótipo de média fidelidade desenvolvido para este trabalho teve como principal objetivo verificar se os jogadores conseguiriam interagir com este tipo de mecânica de coleta e consumo de itens, exploração em jogos 2D e abstração de estruturas de algoritmo. Como primeiras impressões o jogo teve uma boa recepção dos jogadores, mesmo trabalhando assuntos complexos. Pretende-se, com os *feedbacks* coletados, aumentar a quantidade de fases e trazer mais claramente os assuntos propostos no esquema didático, além de dar mais oportunidade aos jogadores de interagirem com outros tipos de mecânicas que estão no conceito do jogo mas não foram implementadas no protótipo, como os Verificadores de Variáveis e os Criadores Dimensionais.

"Variant Scape" é um jogo que possui potencial de ajudar no entendimento de lógica em suas versões futuras, com a possibilidade de aplicar outras mecânicas e, para além de melhorar em questões de *gameplay*, pretende-se aplicar técnicas de *Game Learning Analytics* (GLA), visando fornecer para os professores e desenvolvedores mais informações de como melhorar questões didáticas e técnicas.

Referências

Anastasiadis, T., Lampropoulos, G., and Siakas, K. (2018). Digital game-based learning and serious games in education. *International Journal of Advances in Scientific Research and Engineering*, 4(12):139–144.

Ausubel, D. P., Novak, J. D., and Hanesian, H. (1980). *Psicologia educacional*. Interamericana.

BBC (2018). Introduction to computational thinking. *https://www.bbc.co.uk/bitesize/guides/zp92mp3/revision/1*.

Cheah, C. S. (2020). Factors contributing to the difficulties in teaching and learning of computer programming: A literature review. *Contemporary Educational Technology*, 12(2):ep272.

De Gloria, A., Bellotti, F., and Berta, R. (2014). Serious games for education and training. *International Journal of Serious Games*, 1(1).

Franzoia, F., Pires, F., and Pessoa, M. (2019). Mentorando meninas iniciantes em programação: um estudo de caso. In *Anais do XIII Women in Information Technology*, pages 199–203. SBC.

Gomes, A. and Mendes, A. J. (2007). Learning to program-difficulties and solutions. In *International Conference on Engineering Education–ICEE*, volume 7.

Hayashi, E. C., Posada, J. E. G., Maike, V. R., and Baranauskas, M. C. C. (2016). Exploring new formats of the self-assessment manikin in the design with children. In *Proceedings of the 15th Brazilian Symposium on Human Factors in Computing Systems*, pages 1–10.

Hoed, R. M. (2016). Análise da evasão em cursos superiores: o caso da evasão em cursos superiores da área de computação. *Dissertação (Mestrado Profissional em Computação Aplicada)*.

Honda, F., Pires, F., Pessoa, M., and Maia, J. (2022). Cadê minha pizza? um jogo para exercitar matemática e pensamento computacional através de grafos. In *Anais Estendidos do XXI Simpósio Brasileiro de Jogos e Entretenimento Digital*, pages 876–885. SBC.

Honda, F., Pires, F., Pessoa, M., and Oliveira, E. H. (2023). Automigos: learning design para ludificação de autômatos finitos determinísticos. In *Anais do XXXI Workshop sobre Educação em Computação*, pages 545–556. SBC.

Lahtinen, E., Ala-Mutka, K., and Järvinen, H.-M. (2005). A study of the difficulties of novice programmers. *Acm sigcse bulletin*, 37(3):14–18.

Macena, J., Pires, F., and Melo, R. (2022). Hello food: uma jornada de aprendizagem lúdica em algoritmos, programação e pensamento computacional. In *Anais do XXXIII Simpósio Brasileiro de Informática na Educação*, pages 561–572. SBC.

Macena, J., Pires, F., and Pessoa, M. (2020). Operaçao lovelace: uma abordagem lúdica para introduçao de aprendizagem em algoritmos. *SBC–Proceedings of SBGames*.

Netto, D., Medeiros, L. M., de Pontes, D., and de Morais, E. (2017). Game logic: Um jogo para auxiliar na aprendizagem de lógica de programação. In *Anais do XXV Workshop sobre Educação em Computação*. SBC.

Petri, G., Gresse von Wangenheim, C., and Borgatto, A. F. (2019). Meega+: Um modelo para a avaliação de jogos educacionais para o ensino de computação. *Revista Brasileira de Informática na Educação*, 27(3).

Pires, F., Serique Bernardo, J. R., Pessoa, M., Melo Ferreira, R., and Maquiné de Lima, F. M. (2020). O livro do conhecimento: um serious game educacional para aprendizagem de ortografia da língua portuguesa. *Revista Brasileira de Informática na Educação*, 28(1).

Pires, F. G. d. S. et al. (2021). Thinkted lab, um caso de aprendizagem criativa em computação no nível superior.

Pires, F. G. d. S., Ferreira, R. M., Silva, M. G., Batista, J., Franzoia, F., and Freitas, R. d. (2018). Ecologic: um jogo de estratégia para o desenvolvimento do pensamento computacional e da consciência ambiental. *CBIE 2018*.

Piteira, M. and Costa, C. (2013). Learning computer programming: study of difficulties in learning programming. In *Proceedings of the 2013 International Conference on Information Systems and Design of Communication*, pages 75–80.

Plass, J. L., Homer, B. D., and Kinzer, C. K. (2015). Foundations of game-based learning. *Educational psychologist*, 50(4):258–283.

Qian, Y. and Lehman, J. (2017). Students' misconceptions and other difficulties in introductory programming: A literature review. *ACM Transactions on Computing Education (TOCE)*, 18(1):1–24.

Robins, A. V. (2019). 12 novice programmers and introductory programming. *The Cambridge handbook of computing education research*, page 327.

Saccaro, A., França, M. T. A., and Jacinto, P. d. A. (2019). Fatores associados à evasão no ensino superior brasileiro: um estudo de análise de sobrevivência para os cursos das áreas de ciência, matemática e computação e de engenharia, produção e construção em instituições públicas e privadas. *Estudos Econômicos (São Paulo)*, 49:337–373.

Sarpong, K. A.-M., Arthur, J. K., and Amoako, P. Y. O. (2013). Causes of failure of students in computer programming courses: The teacher-learner perspective. *International Journal of Computer Applications*, 77(12).

Silva, R. R., Rivero, L., and dos Santos, R. P. (2021). Programse: Um jogo para aprendizagem de conceitos de lógica de programação. *Revista Brasileira de Informática na Educação*, 29:301–330.

Tarouco, L. M. R., Roland, L. C., Fabre, M.-C. J. M., and Konrath, M. L. P. (2004). Jogos educacionais. *RENOTE: revista novas tecnologias na educação [recurso eletrônico]. Porto Alegre, RS*.

Wing, J. (2011). Research notebook: Computational thinking—what and why. *The link magazine*, 6:20–23.

Zhonggen, Y. et al. (2019). A meta-analysis of use of serious games in education over a decade. *International Journal of Computer Games Technology*, 2019.