Universidade de São Paulo
Instituto de Matemática e Estatística
Bacharelado em Ciência da Computação

Uma Ferramenta de Simulações Interativas para o Ensino de Conceitos de Programação para Crianças

Marília Takaguti Dicezare

Monografia Final

mac 499 — Trabalho de Formatura Supervisionado

Supervisora: Prof.ª Dr.ª Kelly Rosa Braghetto

São Paulo
2024

Agradecimentos

Do. Or do not. There is no try.
— Mestre Yoda

Texto texto. Texto opcional.

Resumo

Marília Takaguti Dicezare. **Uma Ferramenta de Simulações Interativas para o Ensino de Conceitos de Programação para Crianças**. Monografia (Bacharelado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2024.

Elemento obrigatório, constituído de uma sequência de frases concisas e objetivas, em forma de texto. Deve apresentar os objetivos, métodos empregados, resultados e conclusões. O resumo deve ser redigido em parágrafo único, conter no máximo 500 palavras e ser seguido dos termos representativos do conteúdo do trabalho (palavras-chave). Deve ser precedido da referência do documento. Texto texto. Texto texto.

Palavras-chave: Palavra-chave1. Palavra-chave2. Palavra-chave3.

Abstract

Marília Takaguti Dicezare. **Title of the document: *a subtitle***. Capstone Project Report (Bachelor). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2024.

Elemento obrigatório, elaborado com as mesmas características do resumo em língua portuguesa. De acordo com o Regimento da Pós-Graduação da USP (Artigo 99), deve ser redigido em inglês para fins de divulgação. É uma boa ideia usar o sítio www.grammarly.com na preparação de textos em inglês. Text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text. Text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text.

Keywords: Keyword1. Keyword2. Keyword3.

Lista de abreviaturas

DSR	Pesquisa em Design Science (*Design Science Research*)
FEUSP	Faculdade de Educação da Universidade de São Paulo
IC	Iniciação Científica
IME	Instituto de Matemática e Estatística
MVP	Produto mínimo viável (*Minimum Viable Product*)
SBC	Sociedade Brasileira de Computação
SI	Sistemas de Informação
SUS	*System Usability Scale*
TI	Tecnologia da Informação
UI	Interface do usuário (*User Interface*)
USP	Universidade de São Paulo
WCAG	Diretrizes de Acessibilidade para Conteúdo Web (*Web Content Accessibility Guidelines*)

Lista de figuras

Lista de tabelas

Sumário

Introdução

O ensino da computação estimula o desenvolvimento de importantes habilidades do mundo digital, como o raciocínio lógico, a análise e resolução de problemas, o pensamento crítico, a criatividade, a colaboração, entre outras. O pensamento computacional, que possibilita o desenvolvimento dessas habilidades, vai muito além de programar, sendo fundamental não somente para ciência da computação, mas também para leitura, escrita e aritmética, proporcionando capacidades analíticas para crianças (Wing, 2006).

Essas habilidades são cada vez mais relevantes para crianças e jovens de diversas faixas etárias. Nesse sentido, muito tem sido discutido acerca da aprendizagem de computação na educação básica nas últimas décadas. Escolas em vários países ao redor do mundo já introduziram o ensino de computação. Alguns estudos revelam os desafios e as expectativas desta implementação, além de fatores importantes na educação, em escolas k-12, as quais correspondem às séries do jardim de infância ao ensino médio.

Hubwieser *et al.* (2015) mostraram a situação do ensino de computação em escolas k-12 em 14 casos de estudo em estados de 12 países: Alemanha, Estados Unidos, Índia, Nova Zelândia, França, Coreia, Suécia, Reino Unido, Finlândia, Israel, Rússia e Itália. Eles identificaram diferenças em diversos aspectos nas abordagens, encontrando cerca de 40 termos diferentes para descrever áreas relacionadas à ciência da computação, 24 categorias de objetivos de ensino e 19 de conteúdos de computação, 4 subcategorias de linguagens de programação (sistemas baseados em hardware, ambientes de programação educacional com linguagem própria e baseados em outras linguagens, e linguagens de programação profissionais), e diferentes níveis de educação exigidos dos professores. Os autores acreditam que o mapeamento realizado pode contribuir no processo de planejamento do currículo de ciência da computação nas escolas de ensino fundamental e que a computação poderá se tornar uma matéria regular na educação básica em todos os países do mundo.

Outro trabalho, conduzido por Falkner *et al.* (2019), analisou a implementação do ensino de ciência da computação no k-12, comparando os requisitos curriculares definidos por padrões (pretendido) com o que realmente é implementado nas salas de aula pelos docentes. Foram analisados currículos em 7 países: Austrália, Inglaterra, Irlanda, Itália, Malta, Escócia e Estados Unidos, com foco apenas em conceitos de ciência da computação e linguagens de programação. Eles observaram que os tópicos mais ensinados são algoritmos, programação, pensamento computacional e representação de dados. Alguns tópicos, como inteligência artificial e robótica, constam no currículo pretendido de apenas um país cada, mas são abordados em sala de aula em diversos países. Ainda, eles descobriram que os educadores tendem a escolher as linguagens de programação baseados em fatores motivados pelos alunos ao invés de se basearem no currículo pretendido. O estudo mostrou

que os professores utilizam tanto programação visual quanto baseada em texto no ensino fundamental, além de identificarem o uso comum de atividades desplugadas, apesar destas não estarem explicitamente definidas no currículo padrão. Os autores também identificaram diferenças no alinhamento curricular entre os países, as quais acreditam que possam ajudar a guiar uma futura reforma curricular.

Ademais, Zhu e Wang (2023) realizaram uma pesquisa fenomenológica sobre fatores críticos na educação de ciência da computação em escolas k-12, a partir das perspectivas e visões de professores do ensino superior e de k-12. As entrevistas semi-estruturadas conduzidas com os docentes indicaram que as habilidades de resolução de problemas pelos estudantes, através do pensamento computacional, são as competências mais importantes no ensino de computação no k-12. Além disso, conhecimentos básicos em matemática e programação também são bastante relevantes, enquanto que o ensino de linguagens de programação específicas não foram consideradas importantes.

No Brasil, pesquisadores, instituições de ensino e organizações juntaram esforços para implementar o ensino de computação na educação básica. Segundo relatório do Conselho Nacional de Educação (CNE) (BRASIL, 2021), com a implantação da Base Nacional Comum Curricular (BNCC) em 2017, o CNE ficou responsável por elaborar as normas específicas sobre computação. Entre 2019 e 2021, foram designados os membros da comissão para formular tais normas. Também houve colaborações de pesquisadores da Sociedade Brasileira de Computação (SBC), do Centro de Inovação para a Educação Brasileira (CIEB), do Ministério da Educação (MEC), dentre outras organizações. Finalmente, em 2022, o CNE homologou as Normas sobre Computação na Educação Básica - Complemento à Base Nacional Comum Curricular (BNCC). Alguns estudos mostram o panorama do processo de implementação da computação na educação básica brasileira.

França e Amaral (2013) realizaram um mapeamento sistemático de artigos referentes ao ensino de computação na educação básica publicados entre os anos de 2009 e 2012. Eles observaram um crescente interesse dos pesquisadores brasileiros no assunto, com destaque para as instituições de pesquisa das regiões Nordeste e Sul do país. Em outro estudo, Bordini *et al.* (2016) apresentaram um levantamento de trabalhos relacionados ao pensamento computacional no ensino fundamental e médio realizados entre 2010 e 2015, com o intuito de mostrar o que já foi alcançado nessa área no Brasil. Eles identificaram diferenças nas metodologias de ensino, bem como nas ferramentas utilizadas e no público alvo dos estudos, mostrando conformidade com os trabalhos internacionais.

Ainda, P. S. Santos *et al.* (2018) analisaram a literatura sobre pensamento computacional e programação na educação básica brasileira no período de 2001 a 2016. Eles consideraram fatores como metodologia, ferramentas, público alvo e outros, mostrando novamente o crescente interesse dos pesquisadores no tema. Ademais, os autores apontaram algumas tendências e lacunas nessa área. A. d. S. d. Santos *et al.* (2021) realizaram uma revisão sistemática sobre as diferentes abordagens de ensino de computação no ensino fundamental em estudos publicados entre 2009 e 2020, em bases nacionais e internacionais. Eles analisaram diferentes tipos de atividades voltadas ao ensino de computação: plugadas, desplugadas e híbridas, apontando tendências e lacunas observadas nos trabalhos, como por exemplo, equidade e inclusão.

Diante desse cenário nacional e mundial, a SBC elaborou as Diretrizes de Ensino de

Computação na Educação Básica (Ribeiro *et al.*, 2019), visando facilitar a implementação do ensino de computação nos currículos das escolas brasileiras. Os conceitos de Computação são organizados em 3 eixos, como mostra a Figura 1:

- Pensamento Computacional: refere-se à habilidade de analisar, compreender, modelar, comparar, solucionar e automatizar problemas e soluções de maneira sistemática, por meio de abstrações, análise de informações e da construção de algoritmos.
- Mundo Digital: nele ocorre a codificação ou representação de informações, o processamento dos dados codificados e a distribuição dessas informações de forma segura.
- Cultura Digital: envolve o conhecimento das tecnologias digitais, bem como as relações da computação com outras áreas do conhecimento e a participação crítica, ética e responsável na sociedade do Mundo Digital.

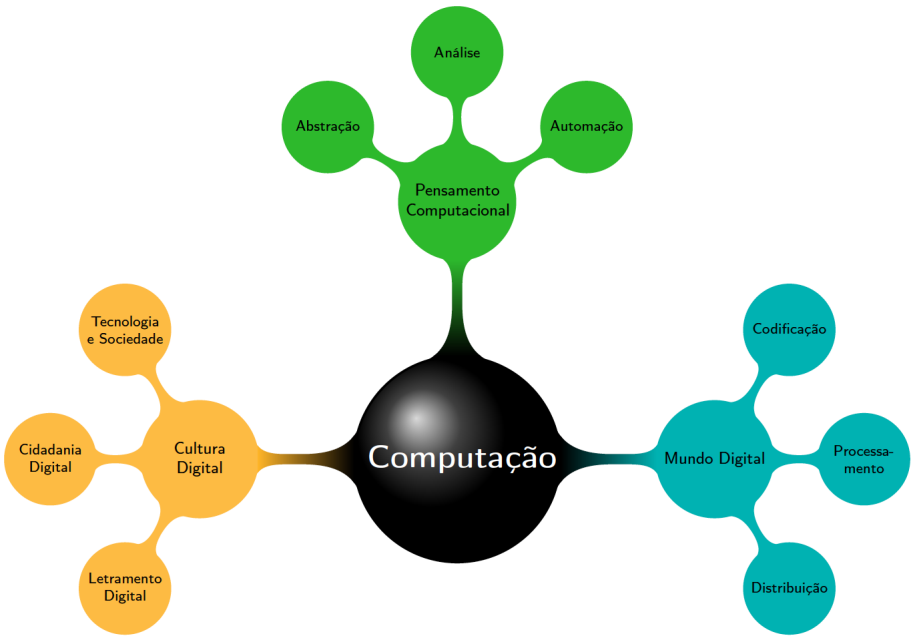

Figura 1: *Eixos da Computação (Ribeiro et al., 2019).*

Dessa forma, é possível perceber que o ensino de computação não se limita à aprendizagem de uma linguagem de programação. Ela é apenas uma ferramenta utilizada para aplicar estes conceitos de forma a chegar na solução de problemas, a partir do conhecimento de teoria da computação e paradigmas de programação (Blatt *et al.*, 2017). Assim, diferentes metodologias podem ser adotadas no ensino de programação para crianças e jovens e o uso de recursos didáticos apropriados, ferramentas ou aplicativos pedagógicos, é indispensável no aprendizado desses conceitos.

Há diversas maneiras de explorar a aprendizagem dos conceitos de computação: jogos, programação visual, programação com blocos, kits de robótica, simulações, storytelling, entre outras. Dentre as ferramentas utilizadas no ensino fundamental, nota-se uma preferência pelas linguagens visuais e plataformas focadas no ensino dos fundamentos e não no desenvolvimento (Gomes *et al.*, 2017). A linguagem de programação em blocos é uma

das mais citadas em estudos, apresentando diversas opções de ferramentas (Brezolin e Silveira, 2021; Souza *et al.*, 2021).

Deste modo, softwares de apoio ao ensino de programação devem facilitar a compreensão das abstrações envolvidas nos conceitos de computação, bem como, estimular o raciocínio lógico. Nesse sentido, o uso de simulações interativas facilita a visualização de conceitos abstratos, utilizando exemplos concretos para representá-los.

Alguns trabalhos mostram a efetividade das simulações no ensino em outras áreas. Em particular, a plataforma de simulações interativas PhET (Physics Education Technology) é um recurso bastante utilizado em várias disciplinas de ciências (Khatri *et al.*, 2013). Criado em 2002 pelo ganhador do prêmio Nobel, Carl Wieman, o projeto da Universidade de Colorado Boulder apresenta atualmente 159 simulações interativas distribuídas nas áreas de física, química, matemática, ciências da terra e biologia, destinadas tanto a alunos da educação básica como de ensino superior.

Entre 2012 e 2013, a plataforma realizou uma pesquisa com docentes que utilizavam a ferramenta, recebendo cerca de 2000 respostas de escolas nos Estados Unidos (Price *et al.*, 2018). O estudo mostrou três aspectos que contribuem para a escolha da ferramenta na sala de aula. Primeiro, devido a sua flexibilidade, as simulações são utilizadas de diversas maneiras e com diferentes objetivos de aprendizagem, como entender conceitos, processos científicos e aumentar a motivação dos estudantes. Além disso, os docentes preferem que os estudantes tenham controle da simulação. Por fim, algumas propriedades percebidas das simulações foram a visualização, manipulabilidade e a capacidade de realizar demonstrações que não poderiam ser feitas em sala.

No Brasil, diversos estudos mostram pontos positivos da utilização dessa ferramenta como forma de agregar ao ensino tradicional. Uma pesquisa bibliográfica de publicações de autores brasileiros entre os anos de 2010 e 2020, realizada por Ramos *et al.* (2020), mostrou que a utilização do simulador PhET, junto a uma metodologia de ensino, potencializou o aprendizado dos estudantes, além de torná-los participantes ativos desse processo.

Outro trabalho, conduzido por Cravo e Espartosa (2021), reforça esta conclusão. Eles avaliaram simulações de ciências e biologia, através de um protocolo de avaliação próprio e da realização de oficinas com futuros docentes, mostrando os desafios e potencialidade da ferramenta. Os autores concluíram que ela contribui positivamente no processo de ensino-aprendizagem, tornando-o mais didático e dinâmico, e que a atuação de professores como mediadores do ensino pode ajudar a contornar possíveis deficiências na sua utilização. Ainda, os trabalhos de Araújo *et al.* (2021) e E. O. Santos e Silva (2020) apresentam conclusões similares em relação ao uso de simulações no ensino de física e química, respectivamente.

Assim, esta é uma abordagem interessante a ser explorada também no ensino da computação, pois da mesma forma que as ciências da natureza e das humanidades ajudam a explicar o mundo real, a ciência da computação ajuda a explicar o mundo digital (Ribeiro *et al.*, 2019). Portanto, é natural que busquemos ferramentas análogas para o estudo dessas ciências, e até onde sabemos, não existem simulações de conceitos de lógica de programação iguais às encontradas no PhET.

Desse modo, este projeto de pesquisa tem como objetivo a criação de um produto

mínimo viável (MVP - *Minimum Viable Product*) de uma ferramenta de simulações interativas para o ensino de conceitos de lógica de programação para crianças no ensino fundamental. Para isso, iremos desenvolver uma versão da aplicação com um conjunto mínimo de requisitos, contendo uma simulação que envolva alguns conceitos de lógica de programação. Além disso, queremos que os usuários tenham contato com a estrutura real do código de programação correspondente a cada conceito simulado. Assim, pretendemos integrar à simulação uma visualização do pseudocódigo associado.

Também queremos aplicar o MVP em sala de aula para obter *feedback* para um futuro desenvolvimento do sistema completo. Dessa forma, iremos avaliar a sua usabilidade com a ajuda de estudantes do ensino fundamental, através de um questionário de usabilidade. Ademais, queremos investigar como representar os conceitos visualmente de forma que facilite a compreensão da lógica de programação por trás deles.

Capítulo 1

Ferramentas Pedagógicas para Ensino de Computação

1.1 Jogos

1.2 Linguagem em blocos

1.3 Robótica

1.4 Computação desplugada

1.5 Simulações

Capítulo 2

Metodologia de Pesquisa

Marília, é sempre bom ter um parágrafo inicial que dê uma visão geral sobre de que se trata o capítulo, para ele não começar direto em uma seção.

Design Science Research

A *Design Science Research* (DSR) é uma metodologia de pesquisa utilizada na elaboração de artefatos com propósitos práticos, constituindo um processo de resolução de problemas de domínio, em que o resultado deve ser avaliado pelo seu valor e utilidade. Esse paradigma é aplicado em diferentes áreas de pesquisa, como Sistemas de Informação, Gerenciamento de Negócios, Engenharia de Software, etc, podendo ser instanciado em variantes muito diferentes (Dresch *et al.*, 2015; Runeson *et al.*, 2020).

Hevner *et al.* (2004) definiram sete diretrizes para a realização de DSR na área de Sistemas de Informação, que têm como princípio a construção e aplicação de um artefato para adquirir conhecimento sobre um problema de design e sua solução. Assim, uma pesquisa em *Design Science* requer a criação de um artefato com propósito para tecnologia da informação e com caráter inovador (diretriz 1), o qual deve considerar a solução de um problema de negócio relevante (diretriz 2). Além disso, o seu design deve ser avaliado rigorosamente considerando sua utilidade, qualidade e eficácia (diretriz 3). O artefato também deve ser inovador, proporcionando contribuições claras e verificáveis para pesquisa (diretriz 4), e sua construção e avaliação devem ser feitas de forma rigorosa (diretriz 5). O processo de design do artefato deve ser conduzido como um processo de pesquisa de uma solução efetiva para um problema (diretriz 6). Por fim, os resultados da pesquisa em *Design Science* devem ser comunicados de forma eficaz (diretriz 7).

Wieringa (2014) estendeu a definição de *Design Science*, apresentando diretrizes para realizar pesquisas em Sistemas de Informação e Engenharia de Software. Para o autor, a interação entre o artefato e o contexto do problema contribui para chegar à solução do problema. Dessa forma, um projeto de DS itera sobre as atividades de design e investigação. A atividade de design é decomposta em três tarefas, designadas como ciclo de design. Este ciclo está inserido em um outro, de engenharia, no qual temos o resultado do ciclo de

design sendo introduzido no contexto real e avaliado. As etapas de cada um dos ciclos estão descritas a seguir:

Ciclo de Engenharia:

- **Ciclo de Design:**
  - **Investigação do Problema:** qual problema deve ser investigado e por quê?
  - **Design do Tratamento (*Treatment Design*):** projeto de um ou mais artefatos para tratar o problema.
  - **Validação do Tratamento (*Treatment Validation*):** análise para validar se esse projeto contribui para o tratamento do problema, caso seja implementado.
- **Implementação do Tratamento (*Treatment Implementation*):** tratamento do problema com um dos artefatos projetados.
- **Avaliação da Implementação (*Implementation Evaluation*):** avaliação do sucesso do tratamento. Ao final desta etapa, podemos ter uma nova iteração no ciclo de engenharia.

O autor também destaca que projetos de pesquisa em *Design Science* estão relacionados apenas às três etapas do ciclo de design.

Runeson *et al.* (2020) define etapas similares a estas para o ciclo de engenharia, envolvendo a conceitualização do problema, o projeto da solução e a validação empírica. Os autores explicam que a *Design Science* abrange duas dimensões principais: problema-solução e teoria-prática. Eles descrevem as atividades de pesquisa que são realizadas de forma iterativa através das duas dimensões, são elas:

- **Conceitualização do problema:** descrição do problema
- **Design da solução:** mapeamento do problema para uma solução geral
- **Abstração:** identificação de decisões de design importantes para uma solução válida dentro de um escopo definido
- **Instanciação:** implementação do artefato em contexto
- **Validação empírica:** avaliação de como a solução implementada abordou o problema

Ademais, projetos de pesquisa em *Design Science* devem considerar dois fatores importantes: a relevância da pesquisa para entidades na resolução de problemas reais e o rigor para que a pesquisa seja considerada válida e confiável, contribuindo para uma determinada área. Dessa forma, a DSR é importante tanto para a produção de conhecimento científico como para a resolução de problemas reais (Dresch *et al.*, 2015; Runeson *et al.*, 2020).

Portanto, a *Design Science* aborda problemas gerais através do estudo de instâncias específicas de problemas no contexto de pesquisa. Para realizar este projeto, seguimos a metodologia de *Design Science Research* em Engenharia de Software proposta por Wieringa (2014), englobando o ciclo de design e de engenharia. Nas seções a seguir, descrevemos as etapas de cada ciclo referentes a esta pesquisa.

2.1 Ciclo de Design

2.1.1 Investigação do Problema

O problema investigado neste estudo foi o de uso de simulações interativas no ensino de conceitos de lógica de programação para crianças. Como apresentado no Capítulo 1, diferentes metodologias são adotadas para isso, utilizando recursos didáticos variados, sendo as linguagens visuais as mais utilizadas. Dessa maneira, investigamos o uso de simulações interativas para o ensino de computação, uma vez que elas podem facilitar a visualização de ideias abstratas envolvidas nos conceitos de programação, utilizando exemplos concretos para representá-las. Ademais, essa metodologia se mostrou eficaz em outras ciências, com o uso da ferramenta PhET. Portanto, queremos expandi-la e testá-la em outros ambientes de aprendizado, como na área de Ciência da Computação.

Para investigar o problema, primeiramente, realizamos uma pesquisa sobre o estado atual do ensino de computação para crianças em escolas no Brasil e no mundo. Em seguida, fizemos uma busca extensiva dos diversos tipos de ferramentas de apoio pedagógico existentes e similares a proposta neste trabalho, procurando entender as principais diferenças entre elas. Grande parte dessa pesquisa foi realizada durante o desenvolvimento do projeto de Iniciação Científica (IC) intitulado "Uma Ferramenta de Simulações Interativas para Ensino de Computação para Crianças", realizado entre outubro de 2022 e outubro de 2023. O resumo dos principais resultados dessa investigação se encontra nos Capítulos **??** e 1 desta monografia.

2.1.2 Projeto do Artefato

O artefato projetado como tratamento do problema é um MVP de uma ferramenta de simulações interativas de conceitos de lógica de programação destinada a crianças do ensino fundamental. Um MVP (*Minimum Viable Product*) ou produto mínimo viável é um produto com funcionalidades suficientes para ser utilizado por usuários iniciais que possam fornecer *feedback* com o intuito de validar uma ideia.

No ensino da computação, alguns conceitos introdutórios são fundamentais para o aprendizado da lógica de programação, tais como variáveis, entrada e saída, operadores lógicos e aritméticos, condicionais, laços de repetição, funções, vetores, matrizes, entre outros. Desse modo, para testar a viabilidade do protótipo proposto, seu projeto deve compreender um conjunto mínimo desses conceitos, a fim de verificar o entendimento deles pelos usuários.

Assim, para projetar os artefatos ou protótipos do MVP, primeiramente, definimos alguns requisitos mínimos, listados a seguir:

- O MVP deve apresentar uma tela com uma área de trabalho e uma opção de ajuda com uma documentação explicativa de uso;
- A área de trabalho deve apresentar um espaço fixo para a simulação e outro para o pseudocódigo correspondente;
- A simulação deve envolver os seguintes conceitos de lógica de programação: variá-

veis, entrada, operadores (aritméticos, lógicos e de comparação), condicionais e laços de repetição;

- A simulação deve apresentar um número limitado de interações possíveis com o usuário;
- A simulação deve encorajar os usuários a explorá-la livremente, com controles intuitivos e uma interface que possibilite boa usabilidade;
- O pseudocódigo deve ser gerado automaticamente conforme as interações com a simulação vão ocorrendo.

Além disso, também definimos e descrevemos os conceitos de lógica de programação que foram simulados e a sintaxe do pseudocódigo gerado. A partir dos protótipos iniciais, estudamos ideias similares e validamos os artefatos projetados para analisar alterações necessárias e as melhores opções de implementação. O Capítulo 3 apresenta em detalhes o desenvolvimento dos protótipos propostos.

2.1.3 Validação do Artefato

A validação dos artefatos foi realizada com base nas opiniões de profissionais de ensino de computação e matemática. Foram consultados: a professora Dra. Kelly Braghetto do Departamento de Ciência da Computação do IME, minha orientadora no Trabalho de Formatura, que também coordena o projeto CodificADAs USP, oferecendo cursos introdutórios de programação voltados para meninas do ensino médio; a estudante Victoria Nóvoa, do curso de Licenciatura em Educomunicação da USP, instrutora de tecnologia educacional de crianças no Colégio Santa Cruz; e o professor Henri Silva da Escola de Aplicação da Faculdade de Educação da USP, que leciona matemática para alunos do 8° ano do Ensino Fundamental II.

Em conversas com os avaliadores, foi possível obter *feedbacks* para verificar a usabilidade e utilidade do MVP a partir dos protótipos projetados, possibilitando o refinamento deles e a escolha de um artefato para implementação, além da definição do público alvo específico para testar a aplicação. No Capítulo 3, descrevemos o processo de validação e as alterações realizadas.

2.2 Ciclo de Engenharia

Após a execução das etapas anteriores, completando uma iteração no ciclo de design, realizamos os demais passos do ciclo de engenharia, descritos a seguir.

2.2.1 Implementação do MVP

O MVP foi desenvolvido utilizando o *framework* de código aberto Vue.js,[1] que é muito utilizado para criar aplicações de página única (*single-page applications*), com a intenção de agilizar o desenvolvimento, permitindo a utilização de soluções existentes e reduzindo a

[1] https://vuejs.org/

necessidade de escrever códigos do zero. O projeto criado com o *framework* foi inicializado para ser utilizado com a linguagem de programação Typescript. Em conjunto com o Vue.js, utilizamos o Vuetify,[2] um *framework* de componentes de UI de código aberto, que contém diversos layouts prontos e componentes dinâmicos.

O código desenvolvido ao longo do projeto, com a implementação do artefato, está disponível no seguinte repositório: https://github.com/mariliatd/logic-sims-mvp, sob a licença MPL-2.0 (Mozilla Public License Version 2.0). Para desenvolver o MVP, utilizamos a organização em componentes que o *framework* Vue possibilita. Dessa maneira, foi possível definir cada conceito de lógica de programação como um componente isolado a ser reutilizado dentro de um template de simulação, conforme necessário. Também separamos os componentes referentes aos elementos visuais da simulação dos componentes de pseudocódigo, criando maior modularidade no código.

Assim, com o artefato implementado, criamos uma página para disponibilizar o MVP, através da ferramenta de versionamento de códigos GiHub, para a aplicação e avaliação da ferramenta em sala de aula.

2.2.2 Avaliação do MVP

A avaliação do MPV foi realizada utilizando um formulário de usabilidade e uma atividade para verificação do aprendizado dos conceitos de lógica de programação. Primeiramente, por meio do questionário de usabilidade, métrica comum na avaliação de um protótipo ou sistema, o MVP foi analisado considerando a facilidade de uso, de aprendizado e satisfação. Em particular, utilizamos o questionário SUS (*System Usability Scale*), aplicado aos usuários finais.

Apesar de não haver medidas absolutas de usabilidade, é possível utilizar escalas gerais para comparar usabilidade em determinados contextos. O SUS representa uma escala de usabilidade com 10 itens que pode ser utilizada para avaliar sistemas (Brooke *et al.*, 1996). Ele é baseado na escala Likert, a qual contém afirmações e os avaliadores indicam o grau de acordo ou desacordo com cada uma, que varia de 1 a 5. Recomenda-se que as respostas para cada item sejam registradas de imediato, sem que os respondentes levem muito tempo pensando nelas.

Para obter o *score* de usabilidade do sistema a partir do questionário SUS, primeiramente, a contribuição de cada item é normalizada para uma escala de 0 a 4. Em seguida, a soma das pontuações dos itens é multiplicada por 2.5, obtendo um *score* em uma escala de 0 a 100. Entretanto, os autores não revelaram como analisar esta pontuação.

Bangor *et al.* (2008) e Bangor *et al.* (2009) realizaram um estudo onde avaliaram a usabilidade de diversos produtos e serviços utilizando o questionário SUS. Eles analisaram a média de pontuação do SUS em 273 estudos com cerca de 3500 pesquisas individuais. Os autores realizaram uma interpretação dessa pontuação, comparando os quartis dos *scores*. Além disso, eles adicionaram uma afirmação ao questionário para avaliar a usabilidade do sistema através de uma escala de classificação de adjetivos, obtendo uma correlação

[2] https://vuetifyjs.com/en/

entre eles e a média das pontuações. As médias que correspondem a cada adjetivo são as seguintes:

1. pior imaginável (*worst imaginable*): 12.5
2. horrível (*awful*) : 20.3
3. ruim (*poor*): 35.7
4. ok (*ok*): 50.9
5. bom (*good*): 71.4
6. excelente (*excellent*): 85.5
7. melhor imaginável (*best imaginable*): 90.9

Os autores ainda expressaram preocupações com o uso do adjetivo "ok", uma vez que ele sugere uma experiência aceitável com o sistema, enquanto a média de pontuações na faixa dele sugere deficiências. Eles acreditam que o termo "razoável" (*fair*) seria melhor indicativo da usabilidade percebida pelos usuários.

Ademais, o SUS é considerado um questionário robusto e confiável, tendo sido utilizado em diversos projetos de pesquisa e avaliações na indústria (Brooke *et al.*, 1996). Embora não tenha sido projetado considerando necessidades específicas de compreensão por crianças, o questionário tem sido utilizado em vários estudos de testes e avaliações de ferramentas com crianças de diversas faixas etárias com sucesso (i.e. Wrońska *et al.*, 2015; Dexheimer *et al.*, 2017; Sánchez-Morales *et al.*, 2020; Tasfia *et al.*, 2023).

Putnam *et al.* (2020) realizaram uma adaptação e teste do SUS com crianças na faixa etária de 7 a 11 anos. Os autores adaptaram o questionário, com o auxílio de professores da educação básica, em um contexto de aplicativos móveis de jogos focados no ensino de programação e pensamento computacional. As afirmações foram ainda modificadas pensando na separação de dois grupos de faixa etária, entre 7 e 8 anos e entre 9 e 11 anos. A Tabela 2.1 mostra os enunciados do SUS original e as adaptações propostas para os dois grupos mencionados, em inglês.

Afirmativa	SUS original	SUS adaptado: Grupo 9-11 anos	SUS adaptado: Grupo 7-8 anos
1	I think that I would like to use this system frequently.	If I had this [app] on my iPad, I think that I would like to play it a lot.	I would like to play [app] a lot more.
2	I found the system unnecessary complex.	I was confused many times when I was playing [app].	[app] was hard to play.
3	I thought the system was easy to use.	I thought [app] was easy to use.	I thought [app] was easy to use.
4	I think that I would need the support of a technical person to be able to use this system.	I would need help from an adult to continue to play [app].	I would need help to play [app] more.
5	I found the various functions in this system were well integrated.	I always felt like I knew what to do next when I played [app].	I knew what to do next when I played [app].
6	I thought there was too much inconsistency in the system.	Some of the things I had to do when playing [app] did not make sense.	Some things in [app] made no sense.
7	I would imagine that most people would learn to use this system very quickly.	I think most of my friends could learn to play [app] very quickly.	[app] would be easy for my friends to learn.
8	I felt the system was cumbersome to use.	Some of the things I had to do to play [app] were kind of weird.	To play [app] I had to do some weird things.
9	I felt very confident using the system.	I was confident when I was playing [app].	I was proud of how I played [app].
10	I needed to learn a lot of things before I could get going with this system.	I had to learn a lot of things before playing [app] well.	There was a lot to learn to play [app].

Tabela 2.1: *Afirmativas em inglês do questionário de avaliação de usabilidade de sistemas SUS e adaptações propostas por Putnam et al. (2020) para crianças entre 7 e 11 anos, dividida nos dois grupos.*

Além das simplificações das afirmações, os autores também utilizaram uma representação visual da escala de Likert (Figura 2.1), sugerida pelos docentes participantes do experimento.

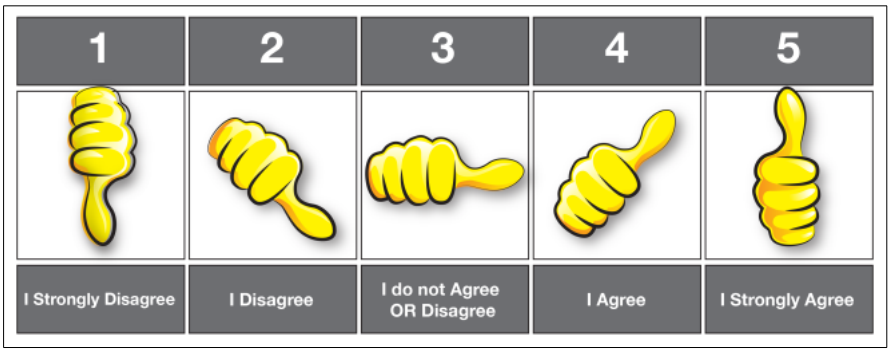

Figura 2.1: *Representação visual da escala de Likert (Putnam et al., 2020).*

Os resultados obtidos nos experimentos mostraram que o questionário modificado juntamente com a escala visual foi compreendido pelas crianças participantes, necessitando apenas de clarificações mínimas. Eles ainda sugeriram alterações nas afirmações 6, 8 e 10 para melhorar a compreensão e a confiabilidade delas.

Dessa forma, utilizamos uma adaptação do questionário SUS para obter o *feedback* dos estudantes sobre o MVP para avaliá-lo em relação a sua usabilidade.

Ademais, através da atividade de aprendizado dos conceitos de programação, analisamos o potencial do ensino de computação utilizando simulações interativas. No Capítulo 5, descrevemos a atividade realizada em sala de aula com a aplicação dos dois formulários para avaliação do artefato implementado.

Capítulo 3

Criação e Validação dos Protótipos do MVP

A partir dos requisitos listados no Capítulo 2.1.2, criamos dois protótipos iniciais de simulações a serem desenvolvidas no Figma,[1] uma plataforma web colaborativa para projetar interfaces. As Figuras 3.1 e 3.2 mostram esses protótipos.

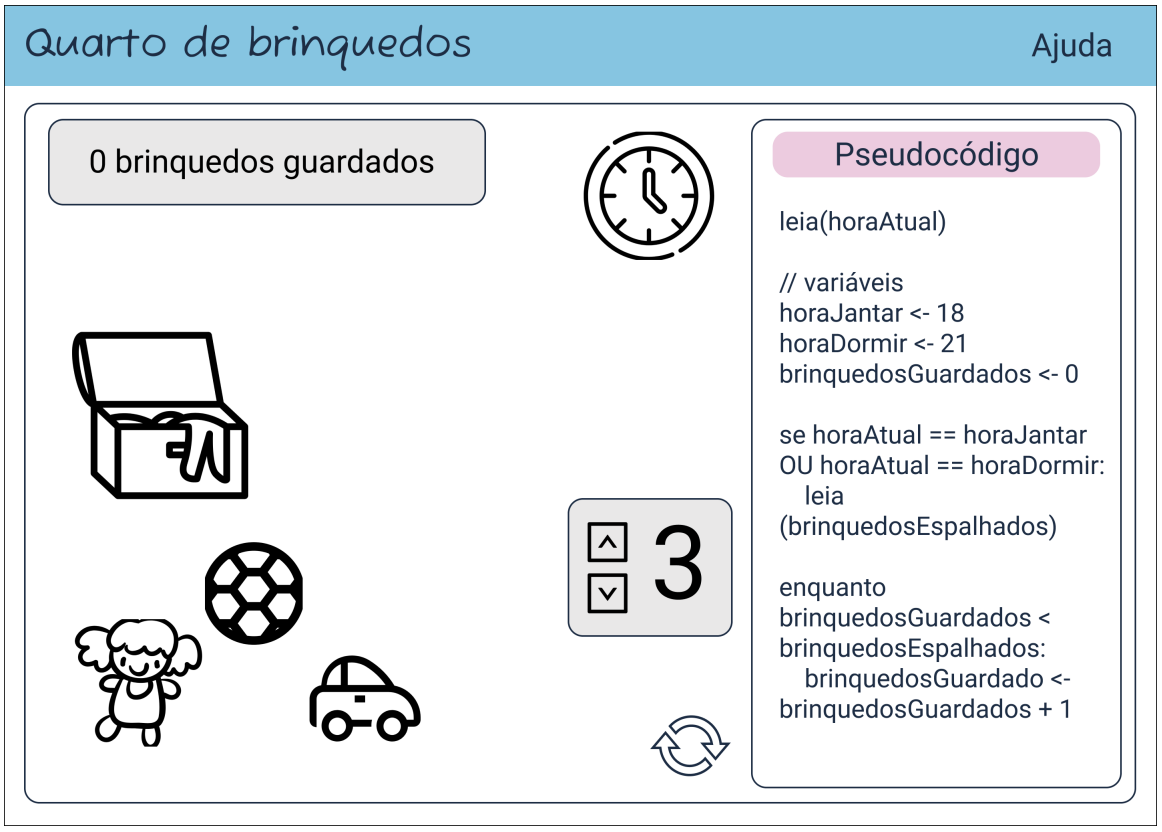

Figura 3.1: *Protótipo inicial da simulação "Quarto de Brinquedos".*

[1] https://www.figma.com/

Na Figura 3.1, temos um cenário de quarto de brinquedos. O protótipo da simulação apresenta elementos visuais como um relógio de parede, um baú para guardar objetos e brinquedos espalhados pelo chão. Além disso, temos dois painéis, um controle para regular a quantidade de brinquedos espalhados e um contador que mostra quantos brinquedos foram guardados. Ainda, o protótipo apresenta um pseudocódigo associado à simulação.

A princípio, a ideia desta simulação é apresentar duas interações possíveis: alterar a hora no relógio e guardar os brinquedos no baú de forma iterativa. Com isso, queremos abordar o conceito de variáveis, principalmente, com a ação de guardar os objetos em um contêiner; de entrada, com a informação recebida através da interação com o usuário ao alterar o horário; de operadores, na comparação de expressões simples e complexas com conectivos lógicos, e na operação de adição contida na ação de guardar os brinquedos; de condicionais, verificando se é o momento de organizar o quarto; e de laços de repetição, repetindo a ação de guardar cada objeto até que o quarto esteja organizado.

Figura 3.2: *Protótipo inicial da simulação “Planejando a Festa”.*

A Figura 3.2 apresenta um cenário de planejamento de uma festa. Nele há elementos visuais que representam itens que podem ser escolhidos para incorporar uma festa de aniversário. Ademais, temos um contador que mostra quantos itens referentes ao escolhido já estão preparados. Nesta simulação, o usuário poderá ter duas interações: escolher um item para a festa e “prepará-lo” (incrementar um determinado item) até estar completo. Deste modo, neste cenário, abordamos os conceitos de variáveis, por meio do total de objetos que devemos ter de acordo com a escolha do item; de entrada, a partir da escolha de um item para a festa; de operadores, na comparação de expressões simples, e na operação

de adição contida na ação de preparar um item; de condicionais, verificando qual item foi escolhido; e de laços de repetição, conforme incremento dos objetos até atingirem o total de acordo com cada elemento.

Com os protótipos iniciais projetados, foram realizadas algumas mudanças antes de apresentá-los, conforme sugestões da professora Kelly. As Figuras 3.3 e 3.4 mostram as alterações sugeridas para os artefatos.

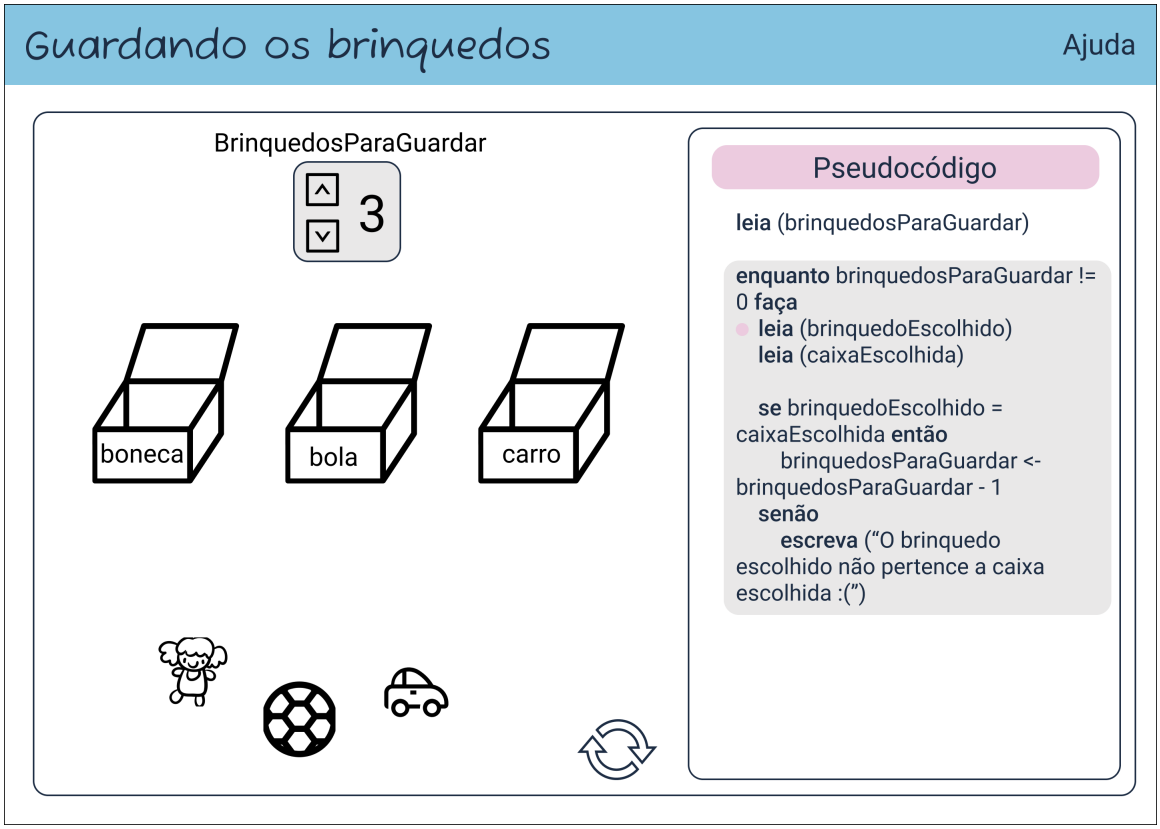

Figura 3.3: *Protótipo da simulação "Guardando os Brinquedos" após validação.*

Na Figura 3.3 podemos observar a nova simulação "Guardando os Brinquedos", que teve sua lógica modificada para simplificar as interações com os elementos visuais e melhorar a usabilidade. Nele, há um contador que permite selecionar até três brinquedos para guardar em suas respectivas caixas. As interações possíveis nessa simulação são: incrementar o contador para escolher quantos brinquedos guardar e, após esta escolha, escolher brinquedo e caixa correspondente. A cada brinquedo guardado corretamente, o valor do contador deve ser subtraído por um. Neste cenário abordamos os conceitos de variáveis, na quantidade de brinquedos para guardar e nos itens (brinquedo e caixa) escolhidos; de entrada e saída, na leituras das variáveis e na escrita da mensagem quando o brinquedo e a caixa escolhida não correspondiam; de operadores, na comparação de expressões simples e na operação de subtração contida na ação de guardar os brinquedos; de condicionais, verificando se o brinquedo e a caixa escolhido correspondiam; e de laços de repetição, repetindo a ação de guardar cada objeto até não sobrar mais nenhum.

Já na nova versão da simulação "Planejando a Festa" (Figura 3.4), podemos observar a mudança no controle do incremento do contador que indica a quantidade de itens totais

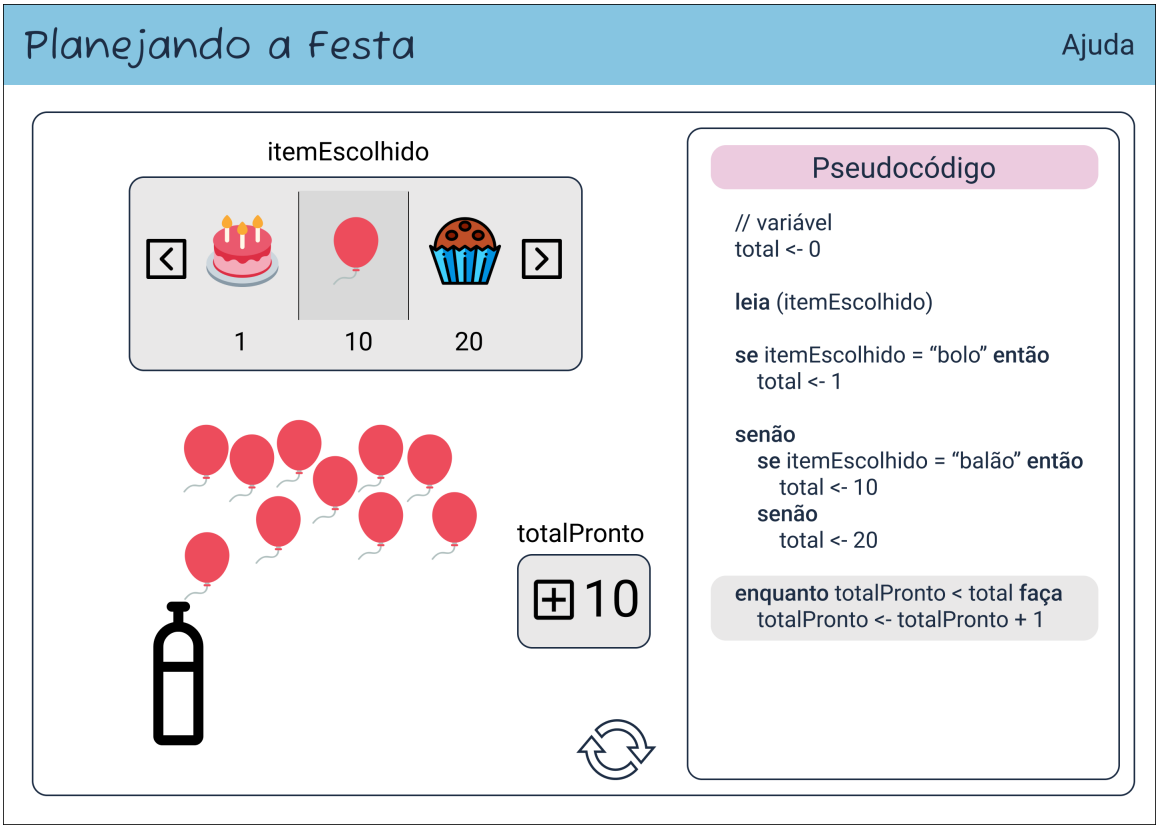

Figura 3.4: *Protótipo da simulação "Planejando a Festa" após validação.*

prontos. Além disso, no geral, as sugestões implementadas nos protótipos envolveram a melhoria nos nomes das variáveis, a incorporação do conceito de saída e o destaque na parte do pseudocódigo em que está ocorrendo a interação na simulação.

Em seguida, em conversas com os avaliadores, foi possível obter *feedbacks* para verificar a usabilidade e utilidade do MVP a partir dos protótipos projetados, possibilitando o refinamento deles e a escolha de um artefato para implementação. A simulação "Planejando a Festa" se mostrou mais interessante e com maiores possibilidades de interações e, por isso, foi escolhida como artefato a ser implementado. Ademais, o público alvo definido para a aplicação do MVP foram crianças no Ensino Fundamental II, a partir do 6° ano.

As demais sugestões recebidas pelos avaliadores levaram em conta, portanto, o protótipo escolhido e foram aplicadas diretamente na implementação do artefato. Dessa forma, a lógica da simulação "Planejando a Festa" foi alterada para melhorar a sua usabilidade e as interações com os elementos visuais. No capítulo a seguir, descrevemos os detalhes da implementação com as modificações realizadas.

Capítulo 4

Implementação do MVP

O MVP da ferramenta de simulações interativas de conceitos de lógica de programação está disponível no link: https://mariliatd.github.io/logic-sims-mvp/. A tela inicial da simulação "Planejando a Festa" pode ser observada na Figura 4.1. Ela apresenta uma barra de navegação no topo da página, com o nome da simulação e um link de "ajuda", o qual abre um diálogo contendo descrições do conceitos de programação abordados. Abaixo dela, há o espaço da simulação que, inicialmente, apresenta um quadro com informações sobre a ferramenta, e o espaço do pseudocódigo a direita.

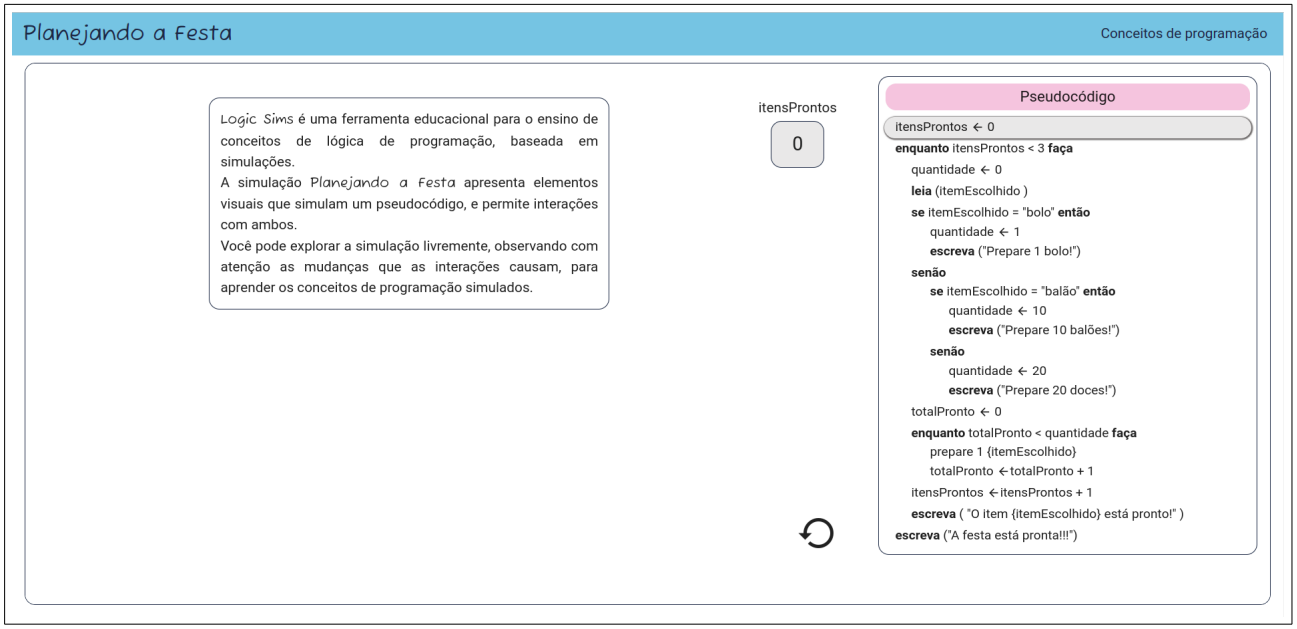

Figura 4.1: *Tela inicial do MVP da simulação "Planejando a Festa".*

Nessa tela, é possível visualizar algumas das alterações sugeridas na etapa anterior. Essa nova versão da simulação também permite interações com o pseudocódigo, o qual tem as linhas destacadas com uma borda com uma animação que pisca quando é necessário interagir com ela. Esse recurso foi implementado para facilitar a visualização da "execução" de cada passo do pseudocódigo, o qual foi modificado. Adicionamos à lógica da simulação

um laço de repetição externo para realizar a iteração na preparação dos três itens disponíveis para compor a festa, que faltava no projeto do protótipo.

Assim, esta simulação permite selecionar três itens em quantidades distintas e preparar cada um até atingir as quantidades correspondentes até que a festa esteja pronta. A seleção de um item pelo usuário representa a entrada ou leitura de um valor no pseudocódigo, que condiciona a quantidade de itens a serem preparados. Ao escolher um item ou finalizar o seu preparo e quando a festa está pronta são impressas mensagens na tela em uma janela representando a saída de dados no pseudocódigo, como mostra a Figura 4.2.

Figura 4.2: *Representação visual do conceito de saída com a mensagem "impressa" na tela.*

Ao selecionar um item para preparo, desabilitamos a escolha dos demais para que a interação com a simulação siga o fluxo do pseudocódigo e não permita que o usuário troque de item enquanto está dentro do laço de repetição interno, por exemplo. A Figura 4.3 mostra o cenário em que o item "bolo" foi selecionado e os demais estão sombreados, não sendo possível clicar neles. Ainda, após o preparo de cada item, este fica desabilitado para escolha na iteração seguinte, limitando o preparo dos três itens (em qualquer ordem) para finalizar o preparo da festa e a simulação.

Na Figura 4.3 também podemos observar que cada variável está separada em um elemento visual próprio. Além disso, removemos o controle do incremento do contador que indica a quantidade total pronta de cada item, deixando essa interação apenas nas figuras de preparo de cada item (fogão, cilindro de gás e pratos) para representar o laço de repetição interno do pseudocódigo. Ainda, adicionamos o pseudocódigo para a execução das atividades de preparo do item escolhido dentro desse laço.

Outro recurso incorporado no pseudocódigo da simulação foi a adição de cores para diferenciar os conceitos de lógica de programação, o qual é comumente utilizado nas linguagens de programação em blocos. A escolha das cores foi feita através da ferramenta

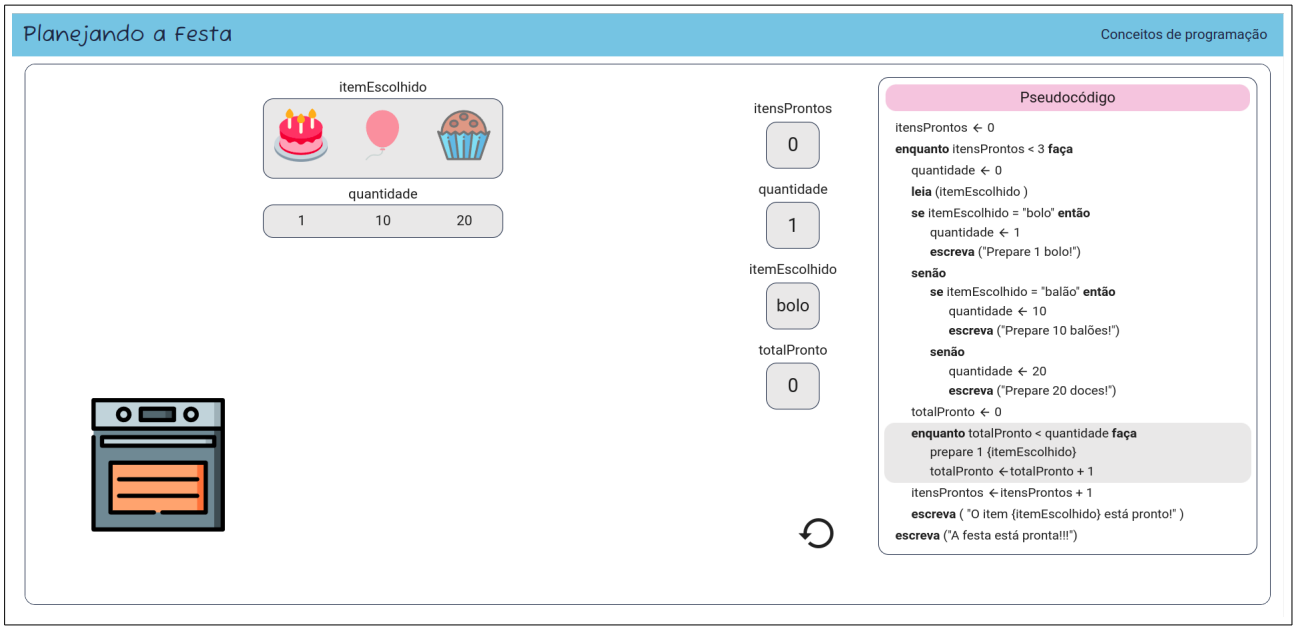

Figura 4.3: *Escolhendo um item na simulação "Planejando a Festa".*

Coolors[1] que além de apresentar um gerador de paleta de cores, também disponibiliza um verificador de contraste. Ela segue as Diretrizes de Acessibilidade para Conteúdo Web (Web Content Accessibility Guidelines - WCAG), a qual define uma série de recomendações para tornar a web mais acessível (Caldwell *et al.*, 2008). A categorização das cores foi feita da seguinte forma:

Além disso, indicamos o nome de cada conceito de lógica de programação. Esses recursos podem ser visualizados ao passar o mouse em cima de cada conceito. A Figura 4.4 apresenta as alterações destacadas.

Por fim, o link para "Ajuda" na barra de navegação foi renomeado para "Conceitos de programação" e ele abre uma caixa de diálogo contendo definições de cada conceito, com figuras e exemplos (Figura 4.5). As definições contêm pseudocódigos de exemplo que seguem o esquema de cores descrito acima.

[1] coolors.co

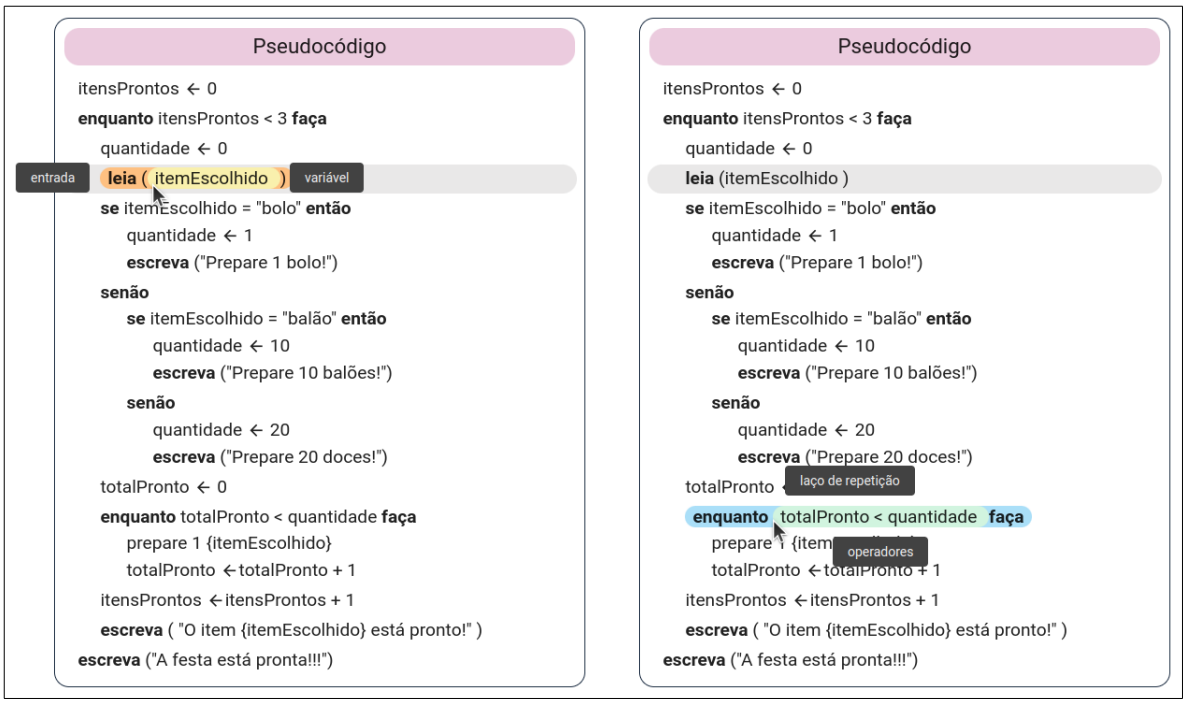

Figura 4.4: *Destaques do pseudocódigo da simulação "Planejando a Festa" mostrando a separação em cores por conceito de lógica de programação e a indicação do nome deles ao passar o mouse por cima de cada um.*

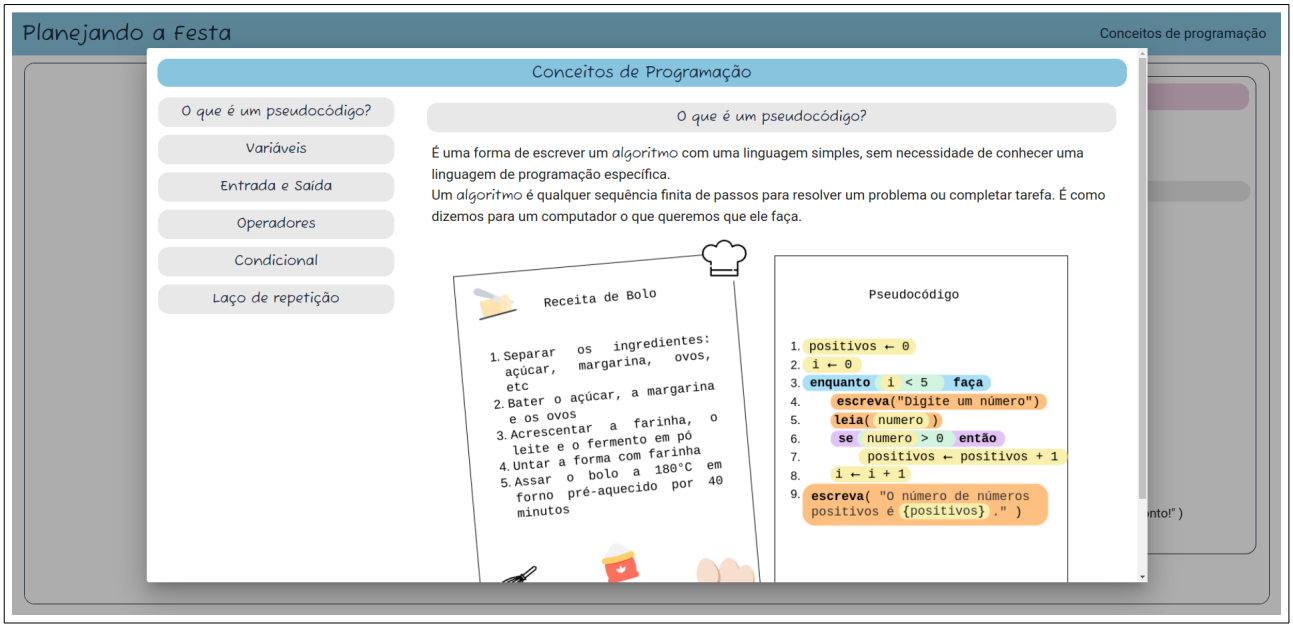

Figura 4.5: *Caixa de diálogo contendo as definições dos conceitos de lógica de programação abordados na simulação "Planejando a Festa". O menu lateral permite a escolha de um conceito, mostrando a sua definição e exemplos ao lado.*

Capítulo 5

Avaliação do MVP

A avaliação do MVP, através do formulário de usabilidade e da atividade de aprendizado, foi realizada em sala de aula com os usuários finais. Para isso, realizamos uma atividade com a duração de uma aula de 50 minutos com uma turma de 26 alunos, entre 13 e 14 anos, do 8° ano do Ensino Fundamental II da Escola de Aplicação da Faculdade de Educação da USP (FEUSP). Durante a aula, contamos com o apoio do professor Henri Silva, que leciona matemática para a turma.

A atividade realizada em aula foi dividade em três partes: na primeira, apresentamos um exemplo de pseudocódigo e código em Python de um programa que, dados cinco números de entrada, calcula a quantidade de números pares e de números ímpares; na segunda parte, os alunos exploraram o MVP com a simulação de forma livre; e na terceira, eles preencheram dois formulários, o de usabilidade e o da atividade de aprendizado, nessa ordem.

As 10 afirmações adaptadas do SUS e traduzidas para o português, apresentadas no formulário de usabilidade, estão enumeradas a seguir e a escala visual de Likert utilizada pode ser observada na Figura 5.1.

1. Eu gostaria de explorar mais a simulação.
2. Eu fiquei confuso(a) muitas vezes enquanto explorava a simulação.
3. Eu achei que a simulação foi fácil de usar.
4. Eu precisaria de ajuda para conseguir explorar a simulação.
5. Eu sabia o que era preciso fazer quando explorei a simulação.
6. Algumas coisas que eu tive que fazer enquanto explorava a simulação não fizeram sentido.
7. Eu acho que a maioria dos meus amigos podem aprender a usar a simulação muito rápido.
8. Algumas das coisas que eu tive que fazer para explorar a simulação foram estranhas.
9. Eu me senti confiante enquanto explorava a simulação.

10. Eu tive que aprender muitas coisas antes de explorar a simulação.

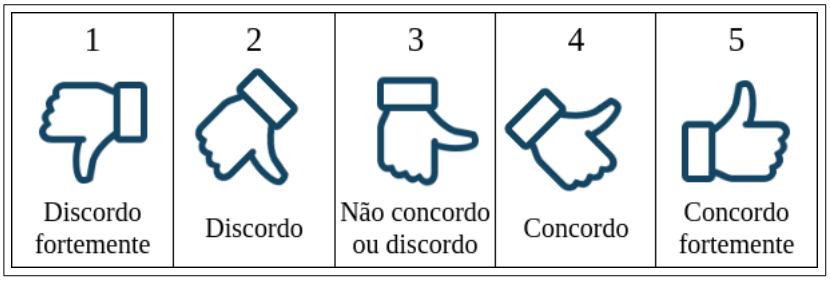

Figura 5.1: *Representação visual da escala de Likert utilizada no formulário de usabilidade.*

Ainda, ao final do formulário foram feitas duas perguntas dissertativas sobre o que o usuário mais gostou na simulação e o que menos gostou, além de um espaço adicional para deixar quaisquer comentários ou sugestões sobre ela.

Em relação à atividade de aprendizado, ela foi dividida em duas partes com 6 perguntas cada. Em cada pergunta, era apresentado um trecho do pseudocódigo com um ou mais conceitos de lógica de programação e era pedido para selecionar a opção que melhor representasse aquele pedaço de pseudocódigo, entre as seguintes opções: variáveis, entrada, saída, operadores, condicional e laço de repetição. A primeira parte da atividade continha perguntas sobre o pseudocódigo da simulação "Planejando a Festa", explorada pelos estudantes. A Figura 5.2 mostra os trechos de pseudocódigo apresentados em cada questão. A segunda parte era referente à simulação "Guardando os Brinquedos", a qual foi apresentada no formulário, para fornecer contexto para as questões, por meio de uma imagem (Figura 5.3). Esse protótipo foi modificado do projeto dos artefatos, apresentado no Capítulo 3. A figura 5.4 mostra os pedaços de pseudocódigo utilizados nessas questões.

O Capítulo a seguir apresenta os resultados da avaliação do MVP feita a partir das respostas dos formulários.

```
se itemEscolhido = "bolo" então
    quantidade ← 1
    escreva ("Prepare 1 bolo!")
senão
    se itemEscolhido = "balão" então
        quantidade ← 10
        escreva ("Prepare 10 balões!")
    senão
        quantidade ← 20
        escreva ("Prepare 20 doces!")
```

(a) *Pseudocódigo referente à Pergunta 1*

```
quantidade ← 0
```

(b) *Pseudocódigo referente à Pergunta 2*

```
enquanto totalPronto < quantidade faça
    prepare 1 {itemEscolhido}
    totalPronto ← totalPronto + 1
```

(c) *Pseudocódigo referente à Pergunta 3*

```
leia (itemEscolhido)
```

(d) *Pseudocódigo referente à Pergunta 4*

```
escreva ("A festa está pronta!!!")
```

(e) *Pseudocódigo referente à Pergunta 5*

```
itensProntos ← itensProntos + 1
```

(f) *Pseudocódigo referente à Pergunta 6*

Figura 5.2: *Trechos de pseudocódigo apresentados em cada pergunta da atividade de aprendizado referente à simulação "Planejando a Festa".*

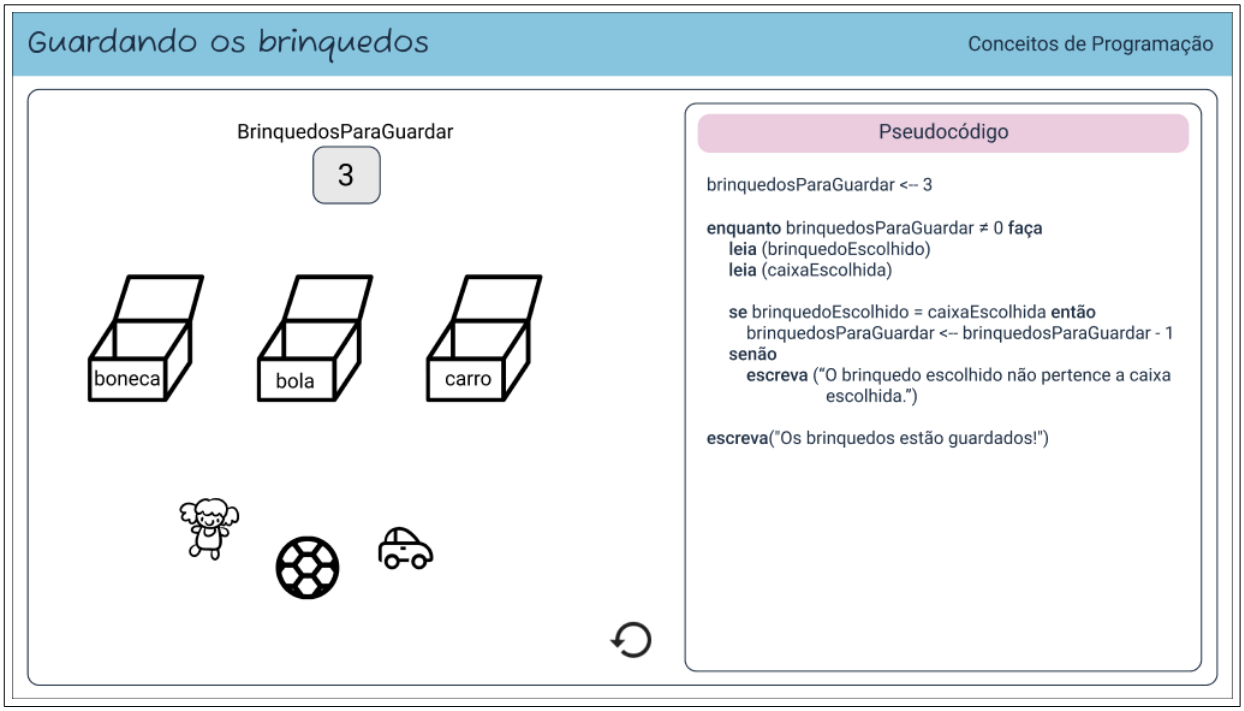

Figura 5.3: *Protótipo da simulação "Guardando os Brinquedos" apresentado na atividade de aprendizado para fornecer contexto para as questões.*

```
se brinquedoEscolhido = caixaEscolhida então
   brinquedosParaGuardar <-- brinquedosParaGuardar - 1
senão
   escreva ("O brinquedo escolhido não pertence a caixa
            escolhida.")
```

(a) *Pseudocódigo referente à Pergunta 1*

```
escreva("Os brinquedos estão guardados!")
```

(b) *Pseudocódigo referente à Pergunta 2*

```
enquanto brinquedosParaGuardar ≠ 0 faça
   leia (brinquedoEscolhido)
   leia (caixaEscolhida)

   se brinquedoEscolhido = caixaEscolhida então
      brinquedosParaGuardar <-- brinquedosParaGuardar - 1
   senão
      escreva ("O brinquedo escolhido não pertence a caixa
               escolhida.")
```

(c) *Pseudocódigo referente à Pergunta 3*

```
brinquedosParaGuardar <-- brinquedosParaGuardar - 1
```

(d) *Pseudocódigo referente à Pergunta 4*

```
brinquedosParaGuardar <-- 3
```

(e) *Pseudocódigo referente à Pergunta 5*

```
leia (brinquedoEscolhido)
leia (caixaEscolhida)
```

(f) *Pseudocódigo referente à Pergunta 6*

Figura 5.4: *Trechos de pseudocódigo apresentados em cada pergunta da atividade de aprendizado referente à simulação "Guardando os Brinquedos".*

Capítulo 6

Resultados da Avaliação do MVP

Os resultados da avaliação do MVP se encontram divididos em duas seções. A primeira se refere ao formulário de usabilidade, no qual obtivemos 26 respostas dos alunos presentes. A segunda é referente às respostas da atividade de aprendizado, na qual tivemos 13 respostas. Em virtude do tempo limitado durante a aula, alguns dos alunos não conseguiram terminar de preencher o segundo formulário, da atividade de aprendizado, por isso, obtivemos menos respostas nele.

6.1 Formulário de Usabilidade

O formulário de usabilidade adaptado do questionário SUS foi preenchido pelos alunos logo após eles terminarem de explorar a simulação. Dos 26 alunos que o responderam, 12 nunca tiveram contato com programação e 14 já haviam tido contato anterior. Para analisar as respostas, separamos os resultados entre esses dois grupos de estudantes.

A Figura 6.1 apresenta as respostas dos alunos que já tiveram contato anterior com programação. Dentro desse grupo de alunos, a maioria respondeu que gostaria de explorar mais a simulação (85.8%). Apenas 14.3% ficaram confusos com a ferramenta, sendo que 64.3% dos estudantes concordaram que ela foi fácil de usar. Ademais, 64.3% desses alunos acreditam que não precisariam de ajuda para explorar o MVP e 21.4% precisariam. Em consonância, 64.3% responderam que sabiam o que era preciso fazer para usar a simulação. Também podemos observar que somente 21.4% das crianças acharam que algumas coisas na simulação não fizeram sentido e 14.3% delas, que tiveram que fazer algumas coisas estranhas na ferramenta. Além disso, a maioria dos usuários (78.6%) acredita que seus amigos poderiam aprender a usar a simulação muito rápido e se sentiram confiantes enquanto a exploravam. Por fim, 28.6% dos alunos acham que tiveram que aprender muitas coisas antes de usar o MVP, enquanto a maioria discorda (57.2%).

Já na Figura 6.2, podemos observar as respostas dos alunos que não tiveram contato anterior com programação. Dentre eles, mais da metade (58.4%) respondeu que gostaria de explorar mais a simulação, porém um grande número (41.7%) não concordou ou discordou com a afirmação. Ainda, 50% dos alunos deste grupo relatou que ficou confuso muitas vezes enquanto explorava o MVP e 33.3% não concordou ou discordou com isso. A maioria

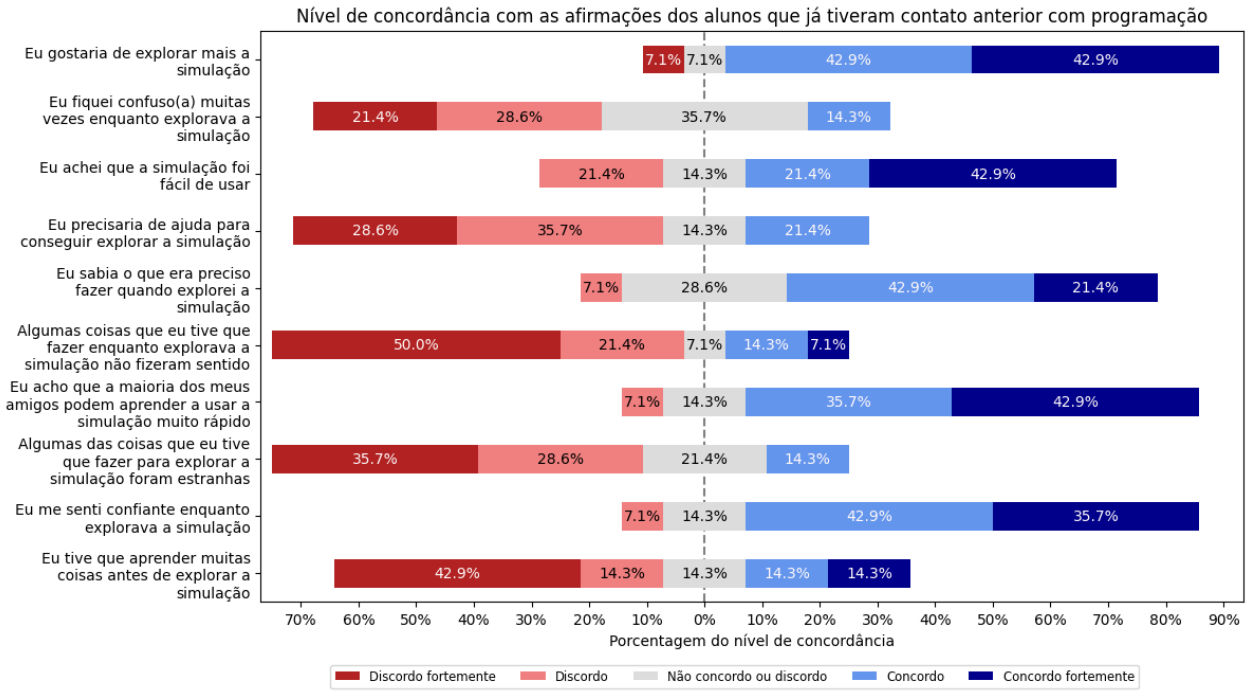

Figura 6.1: *Porcentagem do nível de concordância com as afirmações dos alunos que já tiveram contato anterior com programação.*

(66.7%) também achou que precisaria de ajuda para usar a simulação e apenas 16.6% sabia o que era preciso fazer quando usou a ferramenta. Neste ultimo caso, 50% não concordou ou discordou da afirmação. De forma contraditória, 41.7% acharam que a ferramenta foi fácil de usar e apenas 16.7% discordaram. Ademais, metade dos alunos não concordaram ou discordaram que algumas coisas que tiveram que fazer não fizeram sentido, e apenas 16.7% acharam que algumas coisas foram estranhas, enquanto 41.7% não concordaram ou discordaram. Apesar das respostas anteriores, a maioria dos usuários (72%) acredita que seus amigos poderiam aprender a usar a simulação muito rápido e apenas 25% acha que teve que aprender muitas coisas antes de explorá-la. Por fim, 33.3% das crianças relataram que se sentiram confiantes ao explorar o MVP enquanto 41.7% discordaram.

Ademais, a Figura 6.3 apresenta a média dos níveis de concordância com cada afirmativa dos dois grupos de alunos.

Podemos observar que as afirmações do formulário SUS apresentam duas conotações contrárias, tendo algumas delas um sentimento positivo em relação a ferramenta avaliada e outras um sentimento negativo. No questionário, as questões estão intercaladas, de forma que a primeira apresenta um sentimento positivo, a segunda apresenta um sentimento negativo, e assim por diante. Assim, analisando as respostas dos alunos que já tinham um contato prévio com programação, podemos notar que elas estão dentro do esperado, ou seja, em média, os estudantes concordam com as afirmações de caráter positivo e discordam daquelas de caráter negativo. Isso indica que o MPV projetado demonstrou potencial para a continuação ou complementação do aprendizado dos conceitos de lógica de programação das crianças.

Em relação às respostas dos alunos que não tinham contato prévio com programação,

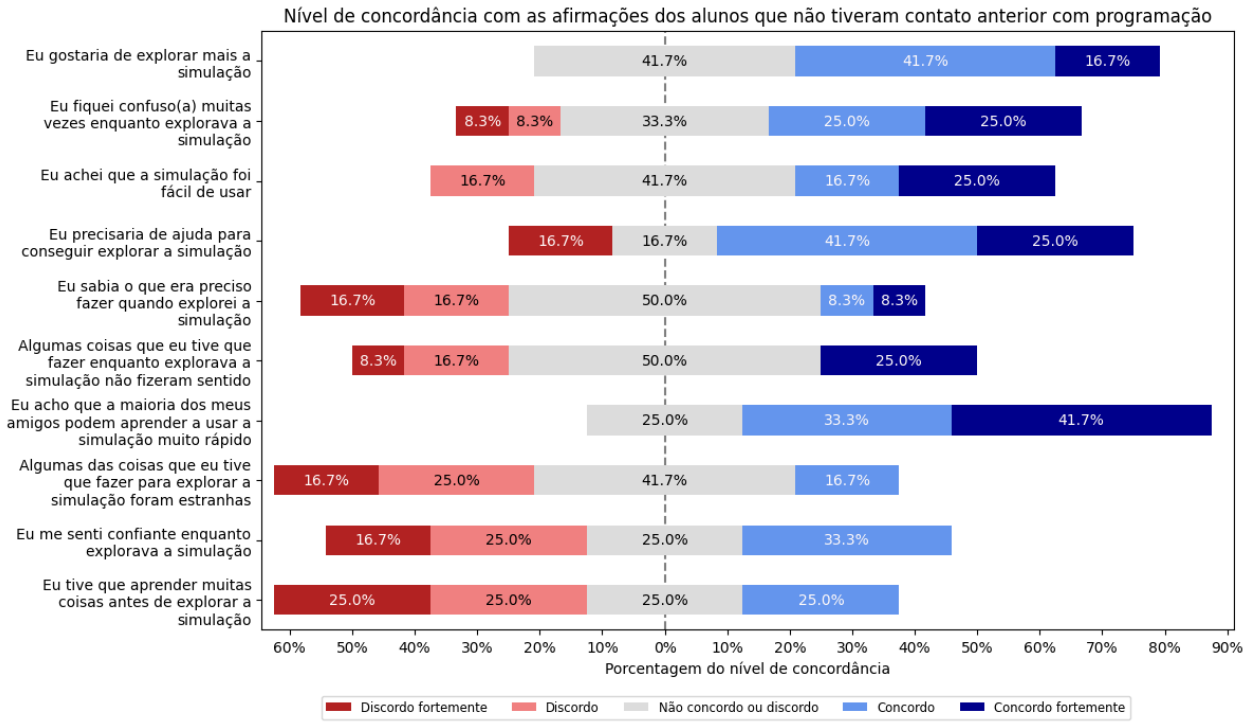

Figura 6.2: *Porcentagem do nível de concordância com as afirmações dos alunos que não tiveram contato anterior com programação.*

observamos algumas inconsistências entre os níveis de concordância com as afirmações de caráter positivo e negativo. Encontramos muitas respostas concordando com afirmações complementares de conotações contrárias. Além disso, em pelo menos metade das afirmações, grande parte dos estudantes permaneceu neutro, optando por não concordar ou discordar, o que não aconteceu com o grupo anterior. Porém, apesar de obtermos muitas respostas concordantes com as afirmações com sentimento negativo em relação ao MVP, a maioria dos estudantes concordou que gostaria de explorar mais a simulação e que seus amigos poderiam aprender a usá-la muito rápido, mostrando o potencial da ferramenta. A partir dessas respostas, verificamos a necessidade de melhorar a forma de introduzir os conceitos de lógica de programação para aqueles que nunca tiveram contato com computação.

Ademais, calculamos as pontuações SUS, na escala de 0 a 100, para as respostas dos dois grupos de alunos. Para os alunos que já tiveram contato anterior com programação, obtivemos uma média de *score* de 71.6, indicando uma boa usabilidade para o MVP, segundo Bangor *et al.* (2009). Para aqueles que não tinham contato anterior com computação, a média de pontuação calculada foi de 53.9, o que demonstra usabilidade razoável para a ferramenta de acordo com os autores.

Opiniões e sugestões sobre a simulação

Em relação as questões dissertativas sobre opiniões e sugestões da simulação, apresentadas ao final do formulário de usabilidade, quando questionado o que mais gostaram na ferramenta, no geral, as crianças relataram que gostaram da experiência. Dividimos

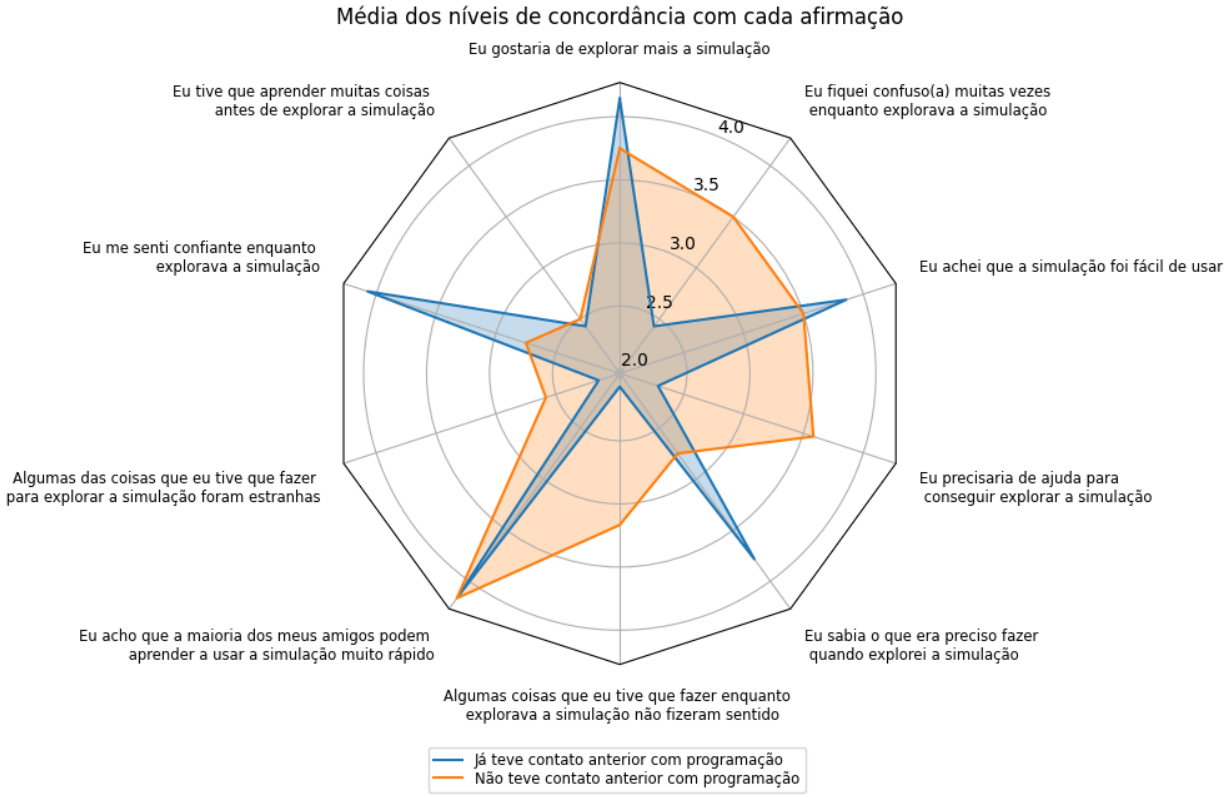

Figura 6.3: *Média dos níveis de concordância com cada afirmação dos alunos que já tiveram contato anterior com programação e dos que não tiveram. Cada raio do gráfico representa a média para uma afirmação do formulário.*

	Já tiveram contato anterior com programação	Não tiveram contato anterior com programação
	82.5	55
	75	50
	100	47.5
	65	47.5
	75	65
	77.5	30
	65	87.5
	67.5	50
	60	57.5
	62.5	40
	72.5	50
	37.5	67.5
	97.5	-
	65	-
média	71.6	53.9

Tabela 6.1: *Pontuação SUS calculada para cada aluno, com as médias para alunos que já tiveram contato anterior com programação e para os que não tiveram.*

as respostas em quatro categorias baseadas nos temas que se destacaram no *feedback*: recursos específicos, interatividade, experiência de aprendizado e experiência geral. A Figura 6.4 apresenta os relatos dos estudantes divididos nessas categorias.

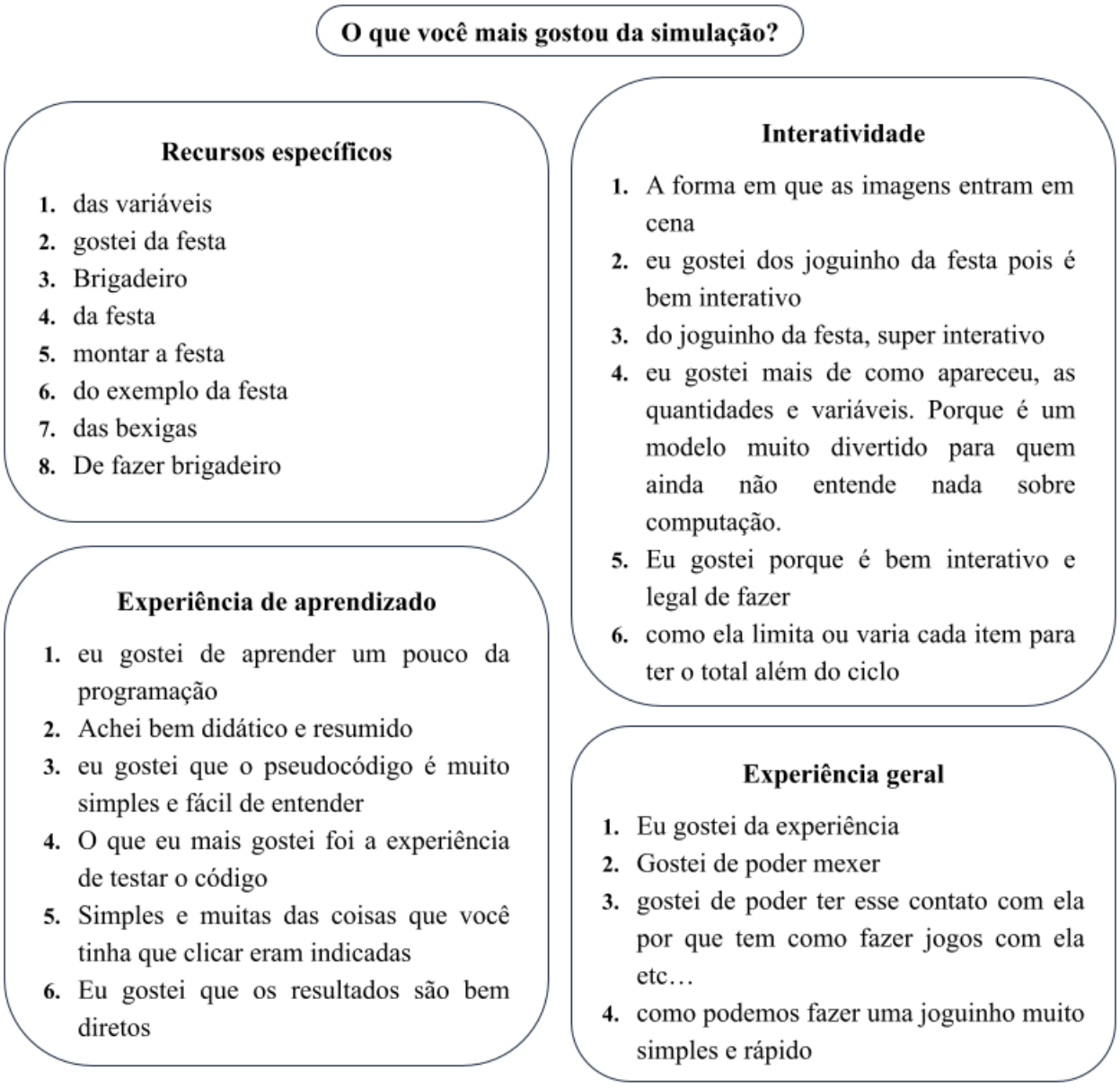

Figura 6.4: *Respostas dos alunos em relação à pergunta livre "O que você mais gostou da simulação?", divididas em categorias.*

Sobre o que menos gostaram da simulação, observamos outras quatro categorias em que se encaixavam os temas de respostas dos alunos: experiência positiva geral, confusão ou dificuldade, design ou problemas de interação, e conteúdo ou tarefas. A Figura 6.5 apresenta os relatos dos estudantes divididos nessas categorias.

Por fim, recebemos comentários referentes a apreciação e divertimento, e complexidade, além de sugestões para organizar os códigos, como mostra a Figura 6.6.

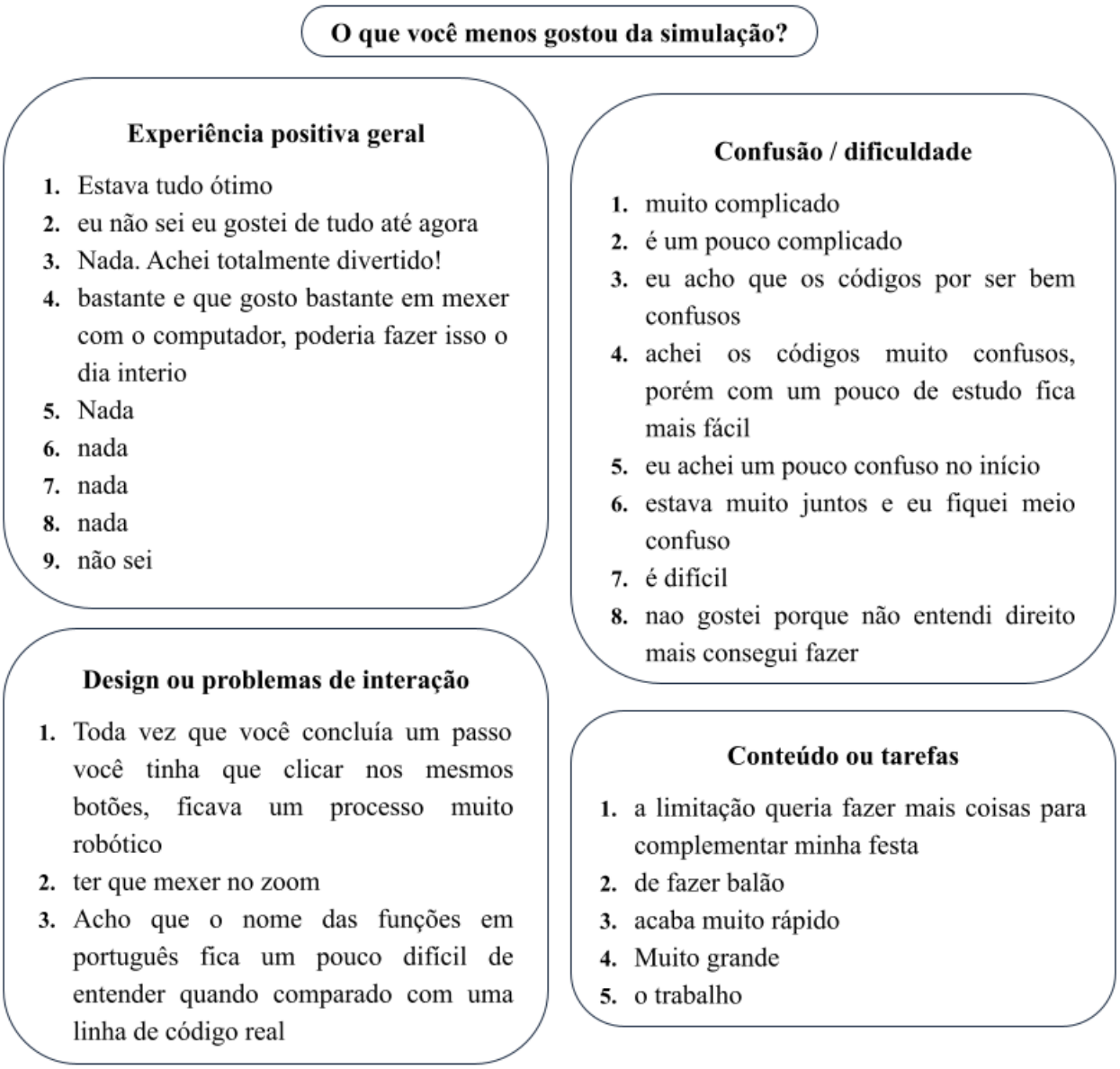

Figura 6.5: *Respostas dos alunos em relação à pergunta livre "O que você menos gostou da simulação?", divididas em categorias.*

6.2 Atividade de Aprendizado

O questionário com a atividades de aprendizado foi respondido por 13 alunos após o preenchimento do formulário anterior. Das 13 respostas, 6 foram de alunos que nunca tiveram contanto com programação e 7 de alunos que já tiveram. Ainda, dos 6 alunos, apenas 3 responderam a segunda parte da atividade, referente à simulação "Guardando os Brinquedos". Para analisar as respostas, separamos os resultados entre os dois grupos de estudantes.

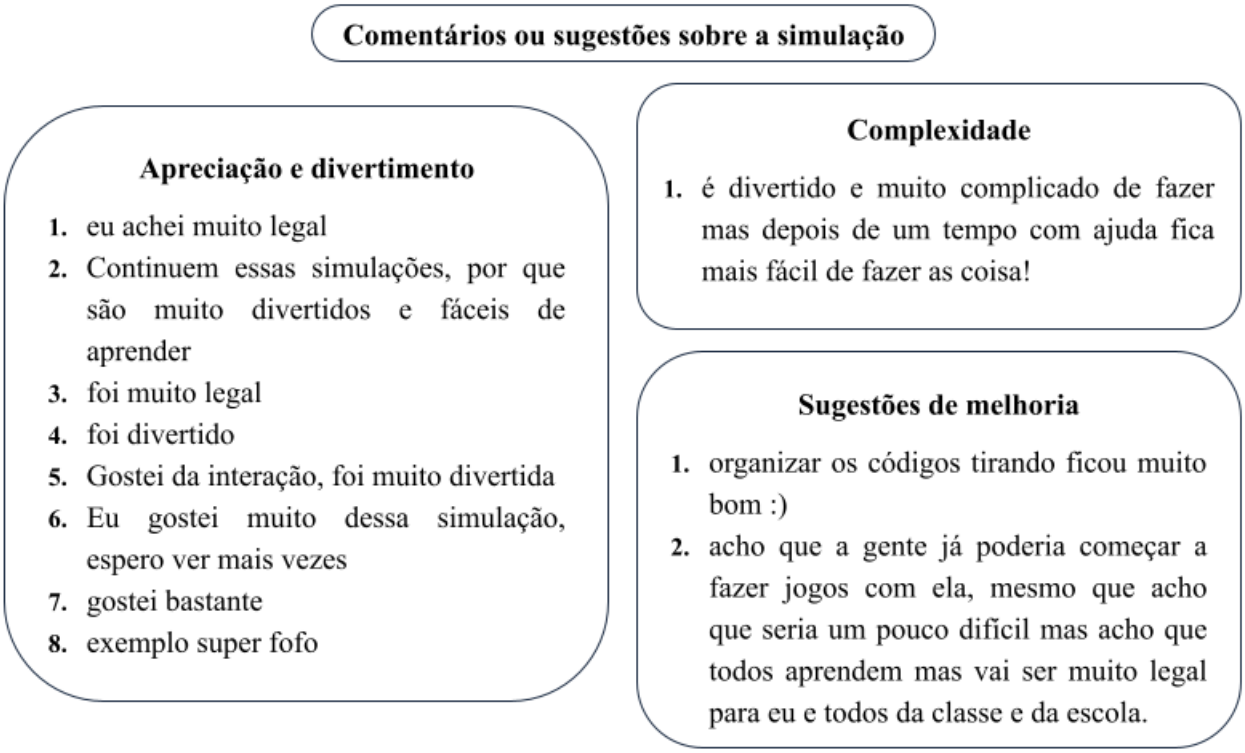

Figura 6.6: *Respostas dos alunos em relação à pergunta livre sobre comentários e sugestões da simulação, divididas em categorias.*

6.2.1 Perguntas sobre a simulação "Planejando a Festa"

A Figura 6.7 apresenta as respostas dos alunos às questões relacionadas à simulação "Planejando a Festa". A pergunta 1 tinha como resposta o conceito de 'condicional'. Considerando os alunos que já tinham um contato anterior com programação, 42.9% acertou esta questão, outros 42.9% escolheram o conceito de 'variáveis' e 14.3%, o de 'saída'. Dentre os demais alunos, que nunca tiveram contato com computação, 33.3% escolheu a resposta certa, enquanto 50% respondeu 'variáveis' e 14.3% escolheu 'saída'. Podemos observar que houve confusão entre o conceito de 'condicional' e o de 'variáveis' e 'saída' pelos dois grupos de estudantes. Entretanto, no trecho de pseudocódigo apresentado nessa questão (Figura 5.2a) temos a repetição dos dois conceitos que foram confundidos como resposta na execução dentro dos condicionais, o que poderia explicar o engano.

No caso da questão 2, que tinha como resposta 'variáveis', observamos que houve uma confusão entre esse conceito e o de 'entrada', que também foi notada no geral. Nessa pergunta, 57.1% dos alunos com conhecimento prévio de computação acertaram a resposta enquanto 42.9% escolheram 'entrada'. Dos alunos que não tinham conhecimento prévio, tivemos um número menor de acertos (33.3%) e o restante também respondeu 'entrada'.

Já na pergunta 3, a resposta correta era 'laço de repetição'. Das crianças que tiveram contato anterior com programação, 42.9% acertaram a questão e os demais responderam 'saída' (42.9%) e 'operadores' (14.3%). Já entre os alunos que não tiveram contato anterior com computação, 33.3% escolheram a resposta correta, enquanto 33.3% escolheram o

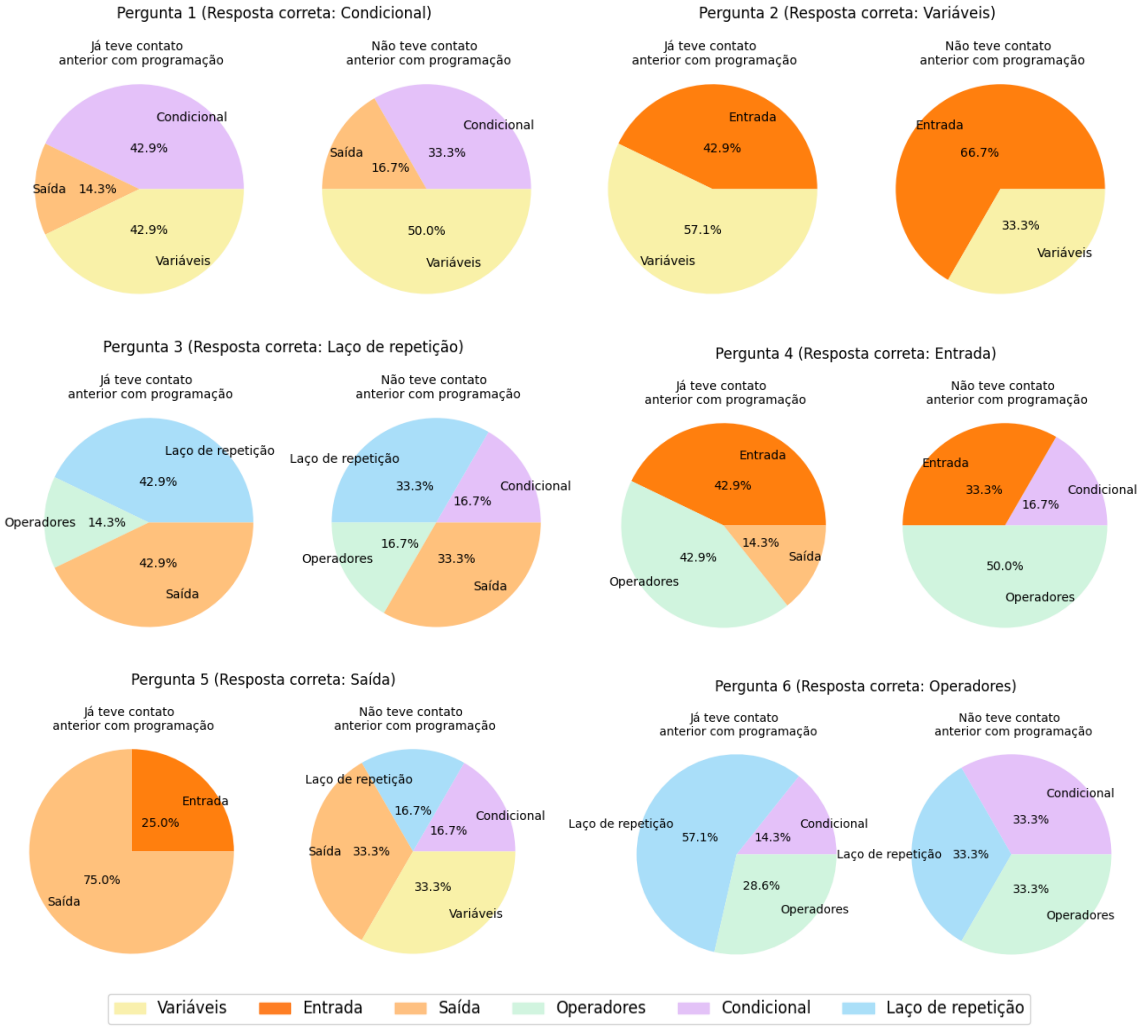

Figura 6.7: *Porcentagens das respostas dos alunos às perguntas da atividade de aprendizado relacionada à simulação "Planejando a Festa". Para cada pergunta, apresentamos as respostas dos alunos que já tiveram contanto anterior com computação à esquerda e as dos que não tiveram à direita.*

conceito de 'saída', 16.7% de 'operadores' e outros 16.7% de 'condicional'. Nesta questão tivemos muitas respostas divergentes, assim como na pergunta 1. O trecho de pseudocódigo apresentado nela (Figura 5.2c) trazia além do laço de repetição, o conceito de operadores representado na condição do laço e dentro da execução dele, o que poderia justificar a escolha desta resposta por alguns alunos. Além disso, acreditamos também que a ação de preparar um item, a qual ocorre dentro do laço, possa ter sido confundida com o conceito de 'saída', respondido por outros. Entretanto, a escolha do conceito de 'condicional' por alguns estudantes indica a falta de compreensão deles.

Nas três última questões, também observamos que não houve entendimento da maior parte dos alunos em relação aos conceitos de lógica de programação. A quarta pergunta do questionário tinha como resposta o conceito de 'entrada'. Nela, observamos que 42.9% dos alunos com conhecimento em programação acertaram a questão. Outras resposta foram 'operadores' (42.9%) e 'saída' (14.3%). Dos alunos sem conhecimento prévio em computação, 33.3% escolheram a resposta certa, enquanto 50% responderam 'operadores'

e 16.7% ‘condicional’.

Na pergunta 5, que tinha como resposta o conceito de ‘saída’, a maioria das crianças com conhecimento anterior de programação acertou (75%) e apenas 25% confundiram com de ‘entrada’. Dos alunos sem conhecimento anterior 33.3% responderam corretamente e os demais alunos responderam ‘variáveis’ (33.3%), ‘condicional’ (16.7%) e ‘laço de repetição’ (16.7%).

Por fim, na questão 6 a resposta correta era ‘operadores’ e poucos alunos acertaram o conceito apresentado. Entre as crianças que já tiveram contato com computação, apenas 28.6% acertou a resposta. A maior parte das respostas foi ‘laço de repetição’ (57.1%) e os demais responderam ‘condicional’ (14.3%). Entre os estudantes que nunca tiveram contato com programação, as respostas foram igualmente escolhidas entre ‘operadores’, ‘laço de repetição’ e ‘condicional’.

6.2.2 Perguntas sobre a simulação “Guardando os Brinquedos”

A Figura 6.8 apresenta as respostas dos alunos às questões relacionadas à simulação “Guardando os Brinquedos”, com a qual não houve interação. As crianças apenas visualizaram a imagem de um estado da simulação e do pseudocódigo completo. No geral, podemos observar que as crianças com conhecimento prévio em programação fizeram mais confusões em relação aos conceitos de programação abordados do que os demais alunos.

A primeira pergunta tinha como resposta ‘condicional’ e, dos alunos que já tiveram contato com computação, 85.7% acertaram, enquanto 14.3% responderam ‘variáveis’. Entre aqueles que não conheciam programação, 66.7% escolheram a resposta correta e 33.3% responderam o conceito de ‘saída’. Nesta questão, novamente, observamos que o trecho de pseudocódigo apresentado (Figura 5.4a) continha também os demais conceitos escolhidos como resposta por alguns estudantes, o que poderia explicar a confusão.

Já na questão 2, cuja resposta era ‘saída’, houve problemas de compreensão dos conceitos por parte da maioria das crianças. Entre aquelas que já conheciam programação, apenas 28.6% acertaram a resposta, enquanto 42.9% escolheram ‘operadores’ e 28.6% escolheram ‘entrada’. Entre os alunos que não conheciam computação, 33.3% deles acertaram e 66.7% escolheram ‘entrada’.

Na pergunta 3, em que a resposta correta era ‘laço de repetição’, as duas crianças que não tinham contato anterior com computação e responderam a questão, acertaram. Dentre os demais alunos, que já tiveram contato, 28.6% acertaram a resposta e o restante escolheu os conceitos de ‘saída’ (28.6%), ‘condicional’ (28.6%) e ‘entrada’ (14.3%). Porém, ressaltamos que todos os conceitos citados estavam presentes no trecho de pseudocódigo apresentado para eles na pergunta (Figura 5.4c).

Já a pergunta 4 tinha como resposta o conceito de ‘operadores’. Entre os alunos que conheciam computação, apenas 14.3% acertaram a questão, e a maioria (42.9%) respondeu ‘variáveis’. Outros escolheram ‘laço de repetição’ (28.6%) e ‘condicional’ (14.3%). Entre os alunos que não tinham conhecimento, a maioria (66.7%) acertou a resposta e 33.3% respondeu ‘laço de repetição’. Nessa questão, o trecho de pseudocódigo apresentado (Figura 5.4d) trazia uma operação sendo atribuída a uma variável. Assim, a confusão com o conceito

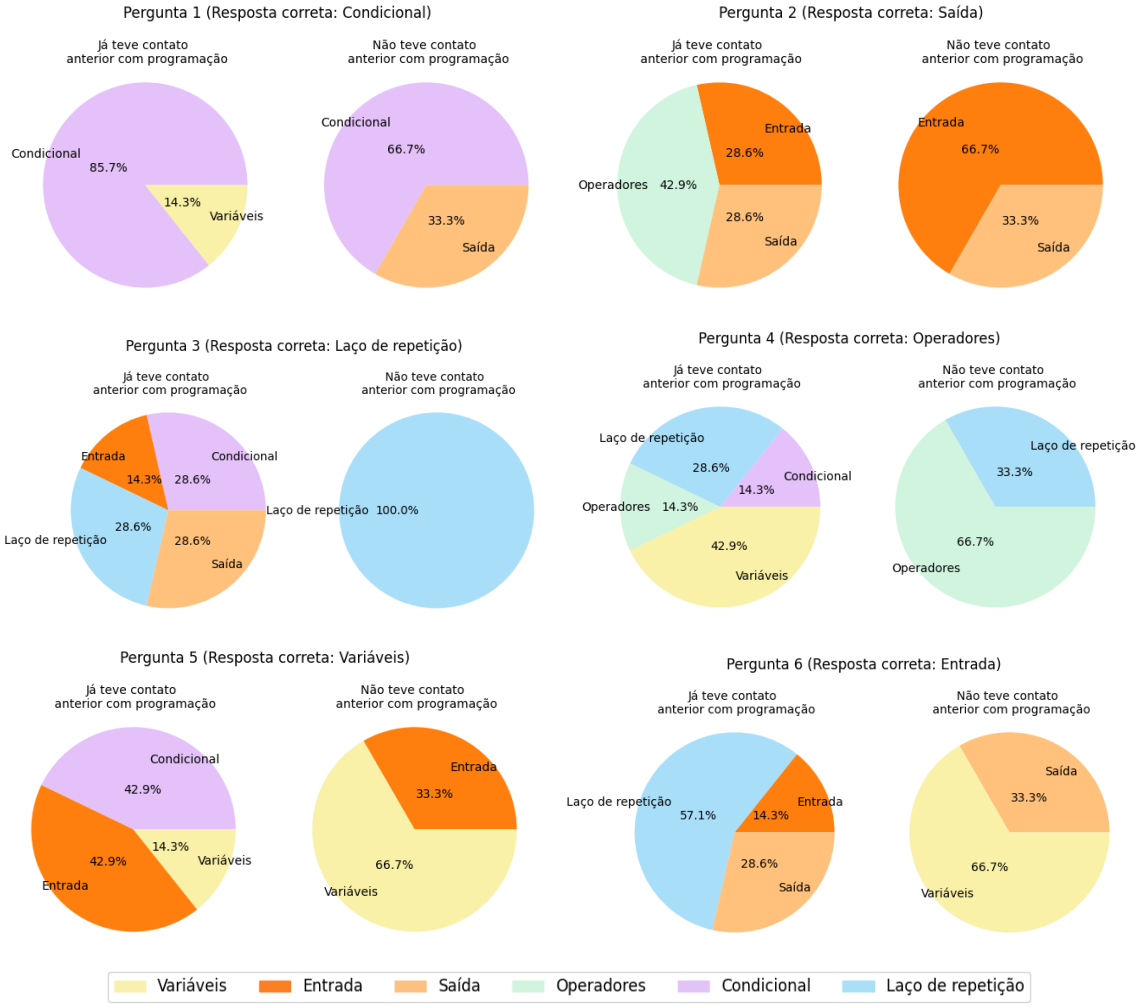

Figura 6.8: *Porcentagens das respostas dos alunos às perguntas da atividade de aprendizado relacionada à simulação "Guardando os Brinquedos". Para cada pergunta, apresentamos as respostas dos alunos que já tiveram contanto anterior com computação à esquerda e as dos que não tiveram à direita.*

de 'variáveis' pode ter ocorrido por isso. Entretanto, a escolha das demais respostas mostrou, novamente, a falta de compreensão dos conceitos de programação por parte das crianças.

Ademais, nas duas últimas questões, também observamos que não houve entendimento dos alunos em relação aos conceitos. Na questão 5, em que a resposta era 'variáveis', dos alunos que tiveram contato anterior com programação, apenas 14.3% acertaram, enquanto 42.9% responderam 'entrada' e 42.9% responderam 'condicional'. Dos estudantes sem conhecimento prévio, 66.7% respondeu 'variáveis' e 33.3% 'entrada'.

Finalmente, na pergunta 6, a resposta correta era 'entrada'. Entre as crianças que conheciam computação, apenas 14.3% acertou a resposta, 57.1% apontaram de forma errada o conceito de 'laço de repetição' e 28.6% o de 'saída'. Dentre os três alunos que nunca tiveram contato com programação, nenhum acertou essa questão, 66.7% deles confundiu o conceito com o de 'variáveis' e 33.3% escolheu 'saída'.

A partir desses resultados, observamos que o uso das simulações não cumpriu a finalidade de ensinar os conceitos de lógica de programação para as crianças. Entretanto, como

observado, algumas confusões dos conceitos de programação pelos alunos podem ter sido causadas pela formulação das questões da atividade de aprendizado. A presença de muitos conceitos, que inclusive se repetiam, nos trechos de pseudocódigo apresentados em algumas questões pode ter gerado ambiguidade nas respostas. Ainda assim, aqueles conceitos que foram apresentados de forma isolada, como o trecho de 'variáveis', não poderia ter sido confundido com o de 'condicional', por exemplo, sendo completamente desrelacionado. Isso também indica a falta completa de entendimento do conteúdo apresentado.

Ademais, notamos que os alunos sem conhecimento prévio em programação tiveram um maior número de acertos nas questões relacionadas à simulação "Guardando os Brinquedos", com a qual eles não interagiram. Isso poderia sugerir a necessidade de alterações e melhorias nas interações implementadas na simulação "Planejando a Festa". Por fim, acreditamos que, no geral, os resultados indicaram a necessidade de diversas reformulações para o MVP, implicando em uma nova iteração nos ciclos de design e engenharia de DSR.

Capítulo 7

Considerações Finais

7.1 Limitações e Trabalhos Futuros

7.2 Conclusões

Referências

[Araújo *et al.* 2021] Evando Santos Araújo, João Lucas Barbosa do Nascimento, Jucilene Carvalho Silva e Caiala Fonseca Andrade Bim. "O uso de simuladores virtuais educacionais e as possibilidades do phet para a aprendizagem de física no ensino fundamental". *Revista de Ensino de Ciências e Matemática* 12.3 (2021), pp. 1–25 (citado na pg. 4).

[Bangor *et al.* 2008] Aaron Bangor, Philip T Kortum e James T Miller. "An empirical evaluation of the system usability scale". *Intl. Journal of Human–Computer Interaction* 24.6 (2008), pp. 574–594 (citado na pg. 13).

[Bangor *et al.* 2009] Aaron Bangor, Philip T Kortum e James T Miller. "Determining what individual sus scores mean: adding an adjective rating scale". *Journal of usability studies* 4.3 (2009), pp. 114–123 (citado nas pgs. 13, 33).

[Blatt *et al.* 2017] Lucas Blatt, Valdecir Becker e Alexandre Ferreira. "Mapeamento sistemático sobre metodologias e ferramentas de apoio para o ensino de programação". In: *Workshop de Informática na Escola (WIE)*. SBC. 2017, pp. 815–824 (citado na pg. 3).

[Bordini *et al.* 2016] Adriana Bordini *et al.* "Computação na educação básica no brasil: o estado da arte". *Revista de Informática Teórica e Aplicada* 23.2 (2016), pp. 210–238 (citado na pg. 2).

[BRASIL 2021] BRASIL. *Normas sobre Computação na Educação Básica - Complemento à BNCC*. Parecer Normativo, n. 23001.001050/2019-18. Ministério da Educação. Conselho Nacional de Educação. Brasília, DF, abr. de 2021. 49 pp. (citado na pg. 2).

[Brezolin e Silveira 2021] Carmen Vera Scorsatto Brezolin e Milene Selbach Silveira. "Panorama brasileiro de uso de ferramentas para desenvolvimento do pensamento computacional e ensino de programação". In: *Workshop sobre Educação em Computação (WEI)*. SBC. 2021, pp. 398–407 (citado na pg. 4).

[Brooke *et al.* 1996] John Brooke *et al.* "Sus-a quick and dirty usability scale". *Usability evaluation in industry* 189.194 (1996), pp. 4–7 (citado nas pgs. 13, 14).

[Caldwell *et al.* 2008] Ben Caldwell *et al.* "Web content accessibility guidelines (wcag) 2.0". *WWW Consortium (W3C)* 290.1-34 (2008), pp. 5–12 (citado na pg. 23).

[Cravo e Espartosa 2021] Andreia Regina Cravo e Karina Dias Espartosa. "Avaliação de simulações interativas em ciências da plataforma on-line "phet" por meio de parâmetros de avaliação e de oficinas com futuros docentes". *Revista de Ensino de Biologia da SBEnBio* (2021), pp. 658–679 (citado na pg. 4).

[Dexheimer *et al.* 2017] Judith W Dexheimer *et al.* "Usability evaluation of the smart application for youth with mtbi". *International journal of medical informatics* 97 (2017), pp. 163–170 (citado na pg. 14).

[Dresch *et al.* 2015] Aline Dresch, Daniel Pacheco Lacerda e José Antônio Valle Antunes Jr. *Design science research*. Springer, 2015 (citado nas pgs. 9, 10).

[Falkner *et al.* 2019] Katrina Falkner *et al.* "An international benchmark study of k-12 computer science education in schools". In: *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education*. 2019, pp. 257–258 (citado na pg. 1).

[Fernandes 2012] Andrino Fernandes. "Animações e simulações para apoio ao ensino da lógica de programação". *Revista Técnico-Científica do IFSC* (2012), pp. 266–266.

[França e Amaral 2013] Rozelma Soares de França e Haroldo José Costa do Amaral. "Ensino de computação na educação básica no brasil: um mapeamento sistemático". In: *Anais do XXI Workshop sobre Educação em Computação*. SBC. 2013, pp. 426–431 (citado na pg. 2).

[Gomes *et al.* 2017] Vitor Gomes, Renata Pontes, Carlos Camelo, Gilvonaldo Cavalcanti e Mirko Perkusich. "Ensino de programação para crianças e adolescentes: um estudo exploratório". In: *Anais dos Workshops do Congresso Brasileiro de Informática na Educação*. Vol. 6. 1. 2017, p. 490 (citado na pg. 3).

[Hevner *et al.* 2004] Alan R Hevner, Salvatore T March, Jinsoo Park e Sudha Ram. "Design science in information systems research". *MIS quarterly* (2004), pp. 75–105 (citado na pg. 9).

[Hubwieser *et al.* 2015] Peter Hubwieser *et al.* "A global snapshot of computer science education in k-12 schools". In: *Proceedings of the 2015 ITiCSE on working group reports*. ACM Digital Library, 2015, pp. 65–83 (citado na pg. 1).

[Khatri *et al.* 2013] Raina Khatri, Charles Henderson, Renee Cole e Jeffrey Froyd. "Over one hundred million simulations delivered: a case study of the phet interactive simulations". In: *Physics Education Research Conference*. Citeseer. 2013, pp. 205–208 (citado na pg. 4).

[Price *et al.* 2018] Argenta M Price, Katherine K Perkins, NG Holmes e Carl E Wieman. "How and why do high school teachers use phet interactive simulations". *Learning* 33 (2018), p. 37 (citado na pg. 4).

[Putnam *et al.* 2020] Cynthia Putnam, Melisa Puthenmadom, Marjorie Ann Cuerdo, Wanshu Wang e Nathaniel Paul. "Adaptation of the system usability scale for user testing with children". In: *Extended abstracts of the 2020 CHI conference on human factors in computing systems*. 2020, pp. 1–7 (citado nas pgs. viii, ix, 14, 15).

[Ramos *et al.* 2020] Marcos Coelho Ramos, Kitawann Tayrone de Sousa Nunes Cardoso e Maria do Carmo Silva Carvalho. "O ensino de ciências com o uso da ferramenta digital simulador phet por meio da estratégia investigativa nos anos finais do ensino fundamental ii". *Anais CIET: Horizonte* (2020) (citado na pg. 4).

[Ribeiro *et al.* 2019] Leila Ribeiro *et al. Diretrizes da sociedade brasileira de computação para o ensino de computação na educação básica*. Rel. técn. 2019 (citado nas pgs. viii, 3, 4).

[Runeson *et al.* 2020] Per Runeson, Emelie Engström e Margaret-Anne Storey. "The design science paradigm as a frame for empirical software engineering". *Contemporary empirical methods in software engineering* (2020), pp. 127–147 (citado nas pgs. 9, 10).

[Sánchez-Morales *et al.* 2020] A Sánchez-Morales, J.A Durand-Rivera e C.L Martínez-González. "Usability evaluation of a tangible user interface and serious game for identification of cognitive deficiencies in preschool children". *International Journal of Advanced Computer Science and Applications* 11.6 (2020) (citado na pg. 14).

[A. d. S. d. Santos *et al.* 2021] Aline de SM dos Santos, Wellington G Pereira e Rozelma Soares de França. "Como ensinar ciência da computação para crianças? tendências e lacunas de pesquisa na área". In: *Anais do XXIX Workshop sobre Educação em Computação*. SBC. 2021, pp. 298–307 (citado na pg. 2).

[E. O. Santos e Silva 2020] Elisângela Oliveira Santos e Ivanderson Pereira da Silva. "Revisão acerca do tema simulações computacionais no ensino de química (2008–2017)". *Debates em Educação* 12.27 (2020), pp. 841–855 (citado na pg. 4).

[P. S. Santos *et al.* 2018] Priscila SC Santos, Luis Gustavo J Araujo e Roberto A Bittencourt. "A mapping study of computational thinking and programming in brazilian k-12 education". In: *2018 IEEE Frontiers in Education Conference (FIE)*. IEEE. 2018, pp. 1–8 (citado na pg. 2).

[Souza *et al.* 2021] Franciely Alves de Souza, Taciana Pontual Falcão e Rafael Ferreira Mello. "O ensino de programação na educação básica: uma revisão da literatura". *Anais do XXXII Simpósio Brasileiro de Informática na Educação* (2021), pp. 1265–1275 (citado na pg. 4).

[Tasfia *et al.* 2023] Sheikh Tasfia, Muhammad Nazrul Islam, Syeda Ajbina Nusrat e Nusrat Jahan. "Evaluating usability of ar-based learning applications for children using sus and heuristic evaluation". In: *Proceedings of the Fourth International Conference on Trends in Computational and Cognitive Engineering: TCCE 2022*. Springer. 2023, pp. 87–98 (citado na pg. 14).

[Wieringa 2014] Roel J Wieringa. *Design science methodology for information systems and software engineering*. Springer, 2014 (citado nas pgs. 9, 10).

[Wing 2006] Jeannette M Wing. "Computational thinking". *Communications of the ACM* 49.3 (2006), pp. 33–35 (citado na pg. 1).

[Wrońska *et al.* 2015] Natalia Wrońska, Begonya Garcia-Zapirain e Amaia Mendez-Zorrilla. "An ipad-based tool for improving the skills of children with attention deficit disorder". *International journal of environmental research and public health* 12.6 (2015), pp. 6261–6280 (citado na pg. 14).

[Zhu e Wang 2023] Meina Zhu e Cheng Wang. "Core competencies of k-12 computer science education from the perspectives of college faculties and k-12 teachers." *International Journal of Computer Science Education in Schools* 6.2 (2023), n2 (citado na pg. 2).