Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Ciência da Computação

ELoS - Ensino de Lógica em Sistemas Gráficos

Rodrigo Oliveira Ferrari

JUIZ DE FORA
DEZEMBRO, 2023

ELoS - Ensino de Lógica em Sistemas Gráficos

Rodrigo Oliveira Ferrari

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Ciência da Computação

Orientador: Rodrigo Luis de Souza da Silva

JUIZ DE FORA

DEZEMBRO, 2023

ELoS - Ensino de Lógica em Sistemas Gráficos

Rodrigo Oliveira Ferrari

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Rodrigo Luis de Souza da Silva
Doutor em Engenharia

Igor de Oliveira Knop
Doutor em Modelagem Computacional

Luiz Maurílio da Silva Maciel
Doutor em Engenharia de Sistemas e Computação

JUIZ DE FORA
12 DE DEZEMBRO, 2023

Aos meus pais.

Resumo

Neste trabalho é apresentado o ELoS (Ensino de Lógica em Sistemas Gráficos), um sistema destinado a introduzir os conceitos fundamentais de lógica de programação de forma lúdica e interativa. Utilizando uma abordagem semelhante a videogames, o sistema apresenta níveis e fases desafiadoras, exigindo que os alunos completem tarefas por meio da digitação de comandos em uma linguagem de programação própria. Desenvolvido para navegadores *web*, o ELoS foi implementado utilizando tecnologias como HTML, CSS, JavaScript e ThreeJS. Este estudo introduziu o ELoS em 14 turmas da disciplina DC5199 (Algoritmos - Prática) na Universidade Federal de Juiz de Fora, envolvendo estudantes de diversos cursos na área de exatas. Uma avaliação inicial coletou dados sobre o perfil dos participantes, incluindo idade, tempo de conclusão e experiência prévia em programação. Os resultados revelaram uma participação diversificada, e a maioria dos participantes encontrava-se nos estágios iniciais de suas trajetórias acadêmicas. O sistema atendeu tanto a alunos iniciantes quanto àqueles com alguma experiência prévia em programação, oferecendo uma oportunidade de aprendizado para diferentes perfis de estudantes. Com base nessas informações, é possível afirmar que o ELoS demonstrou potencial para auxiliar no ensino e aprendizagem de programação.

Palavras-chave: software educacional, lógica de programação, computação gráfica.

Abstract

This work presents ELoS (Teaching Logic in Graphic Systems), a system designed to introduce the fundamental concepts of programming logic in a playful and interactive manner. Employing an approach similar to video games, the system features challenging levels and phases, requiring students to complete tasks by entering commands in its own programming language. Developed for web browsers, ELoS was implemented using technologies such as HTML, CSS, JavaScript, and ThreeJS. This study introduced ELoS to 14 classes in the course DC5199 (Algorithms - Practice) at the Federal University of Juiz de Fora, involving students from various courses in the exact sciences. An initial evaluation collected data on participants' profiles, including age, completion time, and previous programming experience. The results revealed a diverse participation, with the majority of participants in the early stages of their academic journeys. The system catered to both beginner and those with some prior programming experience, providing a learning opportunity for different student profiles. Based on this information, it can be affirmed that ELoS has demonstrated potential to assist in the teaching and learning of programming.

Keywords: educational software, programming logic, computer graphics.

Agradecimentos

À minha mãe, Simone, e meu pai, João, que sempre estiveram ao meu lado, oferecendo amor, incentivo e compreensão, mesmo nos momentos mais desafiadores. Sem a presença constante de vocês, esta jornada não teria sido possível. Vocês são a base sólida sobre a qual construí não apenas este trabalho, mas todo o meu percurso acadêmico.

Ao meu orientador, Rodrigo, pela orientação, paciência e companheirismo ao longo destes últimos anos.

Aos amigos que fiz ao longo desta jornada universitária, pelas risadas compartilhadas nos momentos de descontração, pelo apoio nos desafios acadêmicos e pela companhia que tornou cada etapa mais leve.

Este trabalho não é apenas o resultado de esforço individual, mas um esforço conjunto de todos que contribuíram para o meu desenvolvimento pessoal, acadêmico e profissional durante minha graduação, e por isso, agradeço a todos que, de alguma forma, fizeram parte desta trajetória.

"Persistence is the shortest path to success".

Charles Chaplin

Conteúdo

Lista de Figuras

Lista de Tabelas

1 Introdução

1.1 Apresentação do Tema

O aprendizado de lógica de programação pode ser uma atividade desafiadora, especialmente no momento em que os estudantes são introduzidos ao assunto. Em um exemplo geral sobre dificuldades encontradas por alunos novatos tem-se a transposição de uma tarefa pensada em linguagem natural em uma forma lógica e algorítmica para uma linguagem de programação. Outro caso são as situações em que pode ser difícil de observar a execução de um código. Nesta situação, o aluno, que ainda não está muito familiarizado com as técnicas de depuração de código, não consegue visualizar o que está acontecendo com sua programação e acaba prejudicado na compreensão dos conceitos aprendidos e na identificação de erros que venha a cometer em seus exercícios práticos.

A superação dessas dificuldades é o objetivo de muitos *softwares* educacionais voltados ao ensino de programação. Como apontado no mapeamento sistemático feito por Marcolino e Barbosa (2015), na literatura há diversas ferramentas com o intuito de apoiar o ensino convencional presencial, estender o estudo para além da sala de aula, permitir o aluno observar diversos conceitos através de recursos visuais, entre outros. O mapeamento também aponta questões a serem melhor exploradas em relação a avaliação de eficácia dos *softwares* citados, e também uma predominância em abordagem específicas para determinadas linguagens de programação em detrimento de fundamentos gerais da programação. Ainda assim, o apoio do estudo por *softwares* educacionais ainda pode ser uma alternativa viável para melhorar a qualidade do ensino de lógica de programação.

1.2 Contextualização

Dentre as diversas abordagens para o desenvolvimento de um *software* educacional, as que buscam tornar o aprendizado mais lúdico e divertido podem ser uma boa alternativa para promover um primeiro contato positivo para pessoas de diversos níveis de ensino à

lógica de programação. Como apresentado na análise realizada por Kuz e Ariste (2022), *softwares* educacionais em ambientes lúdicos, como Scratch, Lightbot, entre outros, procuram apresentar os fundamentos da lógica de programação abstraindo as questões mais complexas que geralmente permeiam o ensino deste assunto, como sintaxe e gramática de linguagens de programação convencionais, e também proporcionam uma maneira mais descontraída de se aprender. *Softwares* educacionais lúdicos também tem a característica de exigir pouco ou nenhum conhecimento prévio de seus usuários, o que pode ser essencial para uma maior inclusão ao ensino de computação, já que *softwares* assim podem expandir o aprendizado para além do ensino técnico ou superior. Apesar de *softwares* educacionais lúdicos não serem capazes de apoiar o estudante muito além dos conceitos básicos em comparação a abordagens mais sérias de ensino de lógica de programação, eles se apresentam como uma boa "porta de entrada" para o conteúdo, em especial para despertar o interesse em estudos mais aprofundados de quem está experimentando seu primeiro contato.

1.3 Descrição do Problema

Ainda observando a análise feita por Kuz e Ariste (2022), é possível notar que um aspecto comum em ferramentas educacionais lúdicas consiste na utilização de linguagens de programação em blocos. A princípio este tipo de linguagem cumpre bem o papel introdutório de um estudante na computação, mas o problema se encontra quando esse estudante precisa avançar seus estudos. Para isso ele precisa passar a aplicar seus conhecimentos em linguagens de programação escritas.

Na discussão feita por Moors, Luxton-Reilly e Denny (2018) são apontadas as barreiras que linguagens de programação em blocos causam ao aluno ao tentar transicionar seus conhecimentos para linguagens de programação escritas. Uma dessas barreiras é a falta de confiança em escrever em linguagens com sintaxes complicadas, já que uma das características das linguagens de programação em blocos é eliminar a possibilidade de erros de sintaxe. Outro problema está relacionado à ilusão que linguagens em bloco proporcionam em criar coisas muito complexas de maneira simples em relação ao que é feito com linguagens escritas. Isso faz com um professor que tenha interesse em aplicar

softwares que utilizam esse tipo de linguagem em suas aulas, precise analisar com cuidado como evitar que seus alunos criem dependência por ela e se sintam desestimulados a avançar com os estudos.

1.4 Justificativa

Como elucidado na Seção 1.3, linguagens de programação em blocos tentam contornar a dificuldade que novatos na computação tem em lidar com sintaxe em linguagem de programação através da completa remoção de um erro de sintaxe e uma excessiva simplificação sobre o que é programar linhas de código em uma aplicação qualquer. Isso pode representar a criação de um degrau menor na introdução aos estudos de lógica de programação, porém a curva de dificuldade pode se acentuar ao tomar o próximo passo. Para evitar esse tipo de problema no desenvolvimento de um *software* educacional lúdico, ao invés de eliminar a possibilidade do aluno cometer erros e adiar as dificuldades que ele vai enfrentar em seus estudos, uma abordagem lúdica que ainda utilize linguagem de programação escrita pode ser adotada. Essa abordagem, se utilizada para apresentar os conceitos e possíveis dificuldades que o estudante poderá ter de maneira gradual, pode ser uma solução eficiente em oferecer um primeiro contato com os estudos de lógica de programação. Além disso, ela pode ainda familiarizar o aluno com os próximos passos a serem seguidos em seus aprendizados.

1.5 Objetivos

Este projeto tem como objetivo geral apresentar um *software* educacional de apoio ao ensino de lógica de programação que utilize como recurso tecnológico a computação gráfica para proporcionar o primeiro contato com programação de forma lúdica e divertida. O foco deste projeto são alunos de ensino fundamental e médio. O sistema foi desenvolvido para ser executado em navegadores *web*, facilitando assim o acesso através de diversos dispositivos, como *desktops*, *smartphones* e *tablets*, necessitando obrigatoriamente apenas de conexão à Internet, garantindo assim uma melhor experiência para o uso tanto em sala de aula quanto em estudos individuais. A abordagem adotada para o ensino utiliza

linguagem de programação escrita, de forma a emular a experiência de programar em uma linguagem de programação real e introduzir gradualmente os recursos gerais de lógica de programação, como atribuição de comandos e estruturas condicionais e de repetição.

Um estudo experimental do *software*, consistindo de análises quantitativas sobre dados coletados, representa o primeiro objetivo específico do projeto. O estudo tem a finalidade de averiguar a eficácia do sistema em atingir o objetivo geral proposto. O segundo objetivo específico consiste em disponibilizar o projeto finalizado ao público como uma ferramenta de ensino real, não se restringindo apenas ao uso experimental. Por fim, o terceiro objetivo específico é fornecer materiais para análise e produção de estudos acadêmicos futuros que ampliem os resultados apresentados neste trabalho.

1.6 Metodologia

Este estudo possui caráter aplicado, visando a criação e análise de uma solução prática para o problema descrito na Seção 1.3. Para averiguar a eficácia da solução apresentada, as pesquisas de perfil exploratório contribuíram para compreender como é a recepção do Sistema ELoS nas salas de aula e a formulação de hipóteses acerca de seus acertos e falhas.

Primeiramente foi realizado um levantamento bibliográfico com a finalidade de identificar as principais ferramentas lúdicas educacionais para o ensino de lógica de programação, visando a definição da melhor plataforma para a aplicação em ambientes escolares e a escolha de uma temática interessante como elemento lúdico para este projeto. A pesquisa indicou que a plataforma ideal para o Sistema ELoS são navegadores *web*, e a temática lúdica do projeto deveria assemelhar-se às dinâmicas comumente utilizadas em jogos eletrônicos. Com base nessas decisões de projeto, as tecnologias de desenvolvimento foram selecionadas e deu-se início à elaboração de um primeiro protótipo.

Após a conclusão da primeira versão do projeto, foram realizadas análises quantitativas através da aplicação do protótipo em campo. Essas análises foram conduzidas com alunos ingressantes na área de Ciências Exatas, que cursaram a disciplina DC5199 (Algoritmos - Prática) da UFJF no primeiro semestre de 2023. Esses alunos testaram o primeiro nível implementado do sistema como a primeira atividade prática da disciplina. O experimento teve como objetivo avaliar o nível de dificuldade dos desafios proporcio-

nados pelo sistema, coletando dados de tempo de realização da atividade correlacionados com a experiência do aluno em lógica de programação. Os resultados dessa análise foram positivos, considerando o público-alvo selecionado e, de forma geral, indicaram uma boa recepção inicial do projeto.

2 Fundamentação Teórica

Neste capítulo são apresentados os conceitos e cenários que fundamentam as abordagens adotadas para este projeto, que consiste na comparação entre o ensino de lógica de programação através do uso de ambientes que utilizem linguagem de programação em blocos, e a utilização de linguagens de programação escritas para o ensino. A Seção 2.1 expõe uma breve definição do que consiste uma linguagem de programação em blocos, o motivo de elas serem comumente adotadas para o ensino introdutório na computação, além dos pontos positivos e negativos dessa adesão baseado em estudos da literatura. Na Seção 2.2 são mostrados cenários indicando que o contato com programação escrita no momento da introdução aos estudos é necessário para evitar futuras frustrações de um estudante. Contudo, deve-se também atentar para questões relacionadas a sintaxes complexas sobre linguagens escritas, visto que esse é um ponto crucial na boa compreensão dos conceitos iniciais de lógica de programação.

2.1 Linguagem de Programação em Blocos

Linguagem de Programação em Blocos é um tipo de linguagem que utiliza recursos visuais para substituir a atribuição de comandos através da escrita e busca aumentar o foco na lógica de programação ao invés da sintaxe. Para isso os comandos são dispostos ao usuário em forma de um quebra-cabeça, e para atribuir e conectar os comandos, basta apenas clicar e arrastá-los para seu devido lugar. A Figura 2.1 demonstra um exemplo de código simples feito no *software* educacional Scratch.

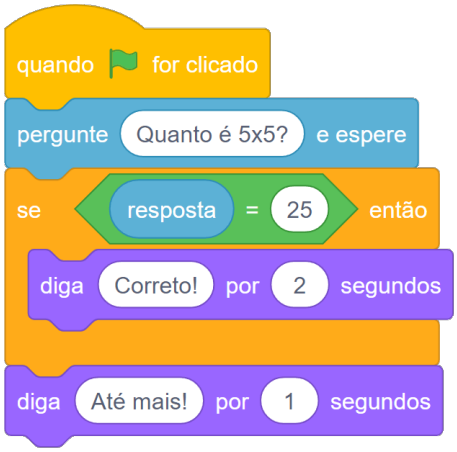


Figura 2.1: Um exemplo simples de um código implementado com linguagem de programação em blocos. ⟨https://idocode.com.br/blog/programacao/programacao-em-blocos/⟩ Adaptado.

Esse tipo de linguagem se tornou uma alternativa popular para ambientes educacionais que introduzem alunos na computação. Uma característica marcante para sua adoção está justamente na ausência de sintaxe e uma baixa exigência de estruturação para tornar o código mais compreensível, como é o caso da indentação nas linguagens de programação escritas. Outro ponto relevante é que *softwares* educacionais que utilizam linguagem em blocos dispõem todos os comandos que a linguagem oferece de forma mais fácil para quem programa (MOORS; LUXTON-REILLY; DENNY, 2018).

Embora linguagens de programação em blocos se mostrem como uma alternativa viável para reduzir as barreiras iniciais de estudantes ao aprendizado de lógica de programação, estudos demonstram que novos empecilhos apareceram para aqueles que tinham interesse em aprofundar seus estudos, especialmente ao iniciarem na programação com linguagens escritas. O contato mais tardio com a sintaxe de uma linguagem escrita se provou ser um fator que abala a confiança de um aluno em avançar com seu aprendizado (MOORS; LUXTON-REILLY; DENNY, 2018). Além disso, remover o desafio de lidar com uma linguagem de programação escrita nem sempre se provou motivador ao ensino. No estudo feito por DiSalvo (2014), jovens afro-americanos estudantes do ensino médio foram treinados e contratados como *game testers*. No treinamento, os estudantes foram expostos a um ambiente de programação em bloco chamado Alice e a uma linguagem de programação baseada em Python, chamada Jython. Ao entrevistar os jovens, foi apon-

tado uma grande preferência deles pela linguagem Jython e como o desafio de aprendê-la foi motivador em relação a utilizar o Alice, que alegaram que era muito mais lento e trabalhoso de utilizar durante o treinamento.

2.2 Linguagem de Programação Escrita

Como relatado na Seção 2.1, privar o aluno de ter contato com sintaxes de linguagens de programação escritas pode acarretar em dificuldades em estudos mais avançados. Um exemplo que segue a linha contrária dessa privação e mostra resultados positivos foi apresentado em Weintrop e Wilensky (2015). Embora eles utilizem uma abordagem que envolva linguagem de programação em blocos, o contato oferecido aos estudantes com a linguagem programação escrita foi praticamente constante. Na pesquisa, foram observados e avaliados os participantes de um curso de Ciência da Computação em uma escola de ensino médio. A ferramenta adotada no curso foi o Snap!, um ambiente de programação híbrida, que procura mesclar programação escrita e em blocos, sendo possível visualizar o que é programado em blocos traduzido para uma linguagem de programação escrita e até programar de maneira escrita as funcionalidades de um bloco. Os resultados da pesquisa foram positivos em relação ao uso da ferramenta e sua capacidade de promover a introdução ao ensino de programação e oferecer uma base para aqueles que pretendem continuar com os estudos.

Contudo, ao fazer uso de linguagens de programação escritas na introdução à lógica de programação, deve-se atentar bem na escolha da linguagem correta para melhor eficácia no ensino. Linguagens com estruturas e sintaxes mais complicadas para um primeiro contato podem se provar como um fator impeditivo no aprendizado. Um exemplo deste caso é mostrado na pesquisa feita por Mladenović, Boljat e Žanko (2018). Nessa análise, três turmas do ensino básico foram introduzidas a programação em três linguagens diferentes, uma delas sendo a linguagem em blocos do Scratch, e as demais sendo as linguagens escritas Logo e Python. O estudo foi feito em cima dos exercícios aplicados nessas turmas sobre a identificação de estruturas de repetição aninhadas, como ilustrado na Figura 2.2. Os resultados identificaram que as turmas que aprenderam com linguagens de programação escritas, em especial as que usaram a linguagem Python, tiveram mais

dificuldade no exercício do que as turmas que aprenderam com o Scratch. A hipótese levantada para esse resultado seria que linguagens fortemente orientadas a indentação, como o Python, são mais difíceis de serem compreendidas por iniciantes na programação.

```
arr = ["A","B","C"]

for i in range(3):
  print(arr[i])
  for i in range(3):
    print(i)
  print("End")
```

Figura 2.2: Um exemplo de loops aninhados em Python em uma estrutura complicada para iniciantes em lógica de programação.

2.3 Considerações Finais

Em resumo, este capítulo fundamenta a decisão do projeto em adotar a programação escrita como uma abordagem de um *software* lúdico de ensino de lógica de programação, tendo em vista que quanto mais tardio o contato do aluno com este tipo de linguagem, mais o seu desempenho em estudos mais avançados é prejudicado. Contudo, há aspectos da linguagem de programação em blocos que são válidos de serem adaptados em uma abordagem escrita, sendo o principal a capacidade de apresentar ao aluno de maneira facilitada o que ele é capaz de fazer com a linguagem. Além disso, é válido ressaltar que a sintaxe de uma linguagem escrita é muito importante na sua adoção para a educação. Uma sintaxe simples e que facilite a identificação dos conceitos pode ser a chave para uma boa introdução de um estudante a programação.

3 Trabalhos Relacionados

Este capítulo apresenta uma revisão da literatura sobre estudos relacionados tematicamente com a proposta do Sistema ELoS. Na Seção 3.1 serão apresentadas diversas soluções dos últimos cinco anos para o ensino introdutório de programação a alunos do ensino básico, variando desde testes de eficiência de apoio ao ensino de softwares educacionais, até a avaliação de cursos e *workshops*. As abordagens apresentadas variam desde aproximações mais sérias, com o objetivo de promover o ensino de maneira tradicional, até abordagens mais lúdicas e divertidas. Os estudos também utilizam diversos modelos de programação, desde abordagens visuais utilizando programação em blocos, até abordagens mais tradicionais que utilizam linguagens de programação escritas convencionais. Também são abordadas situações em que ambos os modelos foram utilizados. Ao final, todos os estudos foram sintetizados na Tabela 3.1 para fins comparativos. A seção 3.2 apresenta as considerações finais sobre os trabalhos citados.

3.1 Descrição dos Trabalhos

Foi proposto por Chochiang et al. (2019) o emprego da linguagem de programação visual denominada ArViz como uma ferramenta de suporte para estudantes do ensino médio no aprendizado de IoT (Internet das Coisas). Foi desenvolvido um curso abordando 11 tópicos introdutórios, destinado a uma turma composta por 28 alunos da Kathu Wittaya School, na qual apenas alguns estudantes possuíam conhecimento prévio sobre IoT. Para isso, foi utilizado o ambiente de programação em blocos da plataforma ArViz, juntamente com uma placa microcontroladora e sensores. Como método de avaliação, foi realizado um pré-teste e um pós-teste. Os testes demonstraram uma melhoria nas habilidades com um intervalo de confiança de 95%. Além disso, foi aplicada uma pesquisa de satisfação, na qual mais de 80% dos alunos demonstraram estarem satisfeitos com o curso. Portanto, pode-se concluir que o uso dessa plataforma no curso mostrou-se eficaz ao auxiliar os jovens no aprendizado de IoT.

O projeto apresentado em Seralidou e Douligeris (2021) teve como objetivo introduzir o ensino de programação por meio de jogos digitais em uma sala de aula. Nessa turma do ensino fundamental, localizada na Grécia, os alunos aprenderam conceitos de programação utilizando o Scratch, uma ferramenta online baseada em programação por blocos. O jogo desenvolvido, denominado "Objects game", permitiu que os estudantes relacionassem as ideias de um conteúdo complexo, como a programação orientada a objetos, de maneira interativa e didática. Após a conclusão do projeto, um questionário de satisfação foi aplicado aos alunos tendo como resultado uma avaliação bastante positiva.

No estudo realizado por Saito et al. (2019), teve-se como objetivo analisar os efeitos do ensino de programação em texto para crianças do ensino fundamental. Foi implementado um curso que utilizou a linguagem Python em conjunto com o Raspberry Pi. O curso foi estruturado em sete tópicos, que abordaram desde noções básicas até o desenvolvimento de um produto final, cuja apresentação ocorreu no último tópico. Durante esse período, 19 alunos foram preparados para criar um projeto relacionado a um problema real de programação. Para avaliação, foram utilizados três métodos: dois questionários respondidos pelas crianças (pré-teste e pós-teste) e rubricas preenchidas pelo professor. Os resultados demonstraram uma melhora significativa na compreensão da programação. É importante ressaltar que o estudo de caso enfrentou algumas dificuldades na obtenção de parâmetros de avaliação, uma vez que, com o método escolhido, não foi possível atribuir notas a muitos dos tópicos abordados no curso.

O estudo conduzido por Mattos e Ferreira (2021) propôs a realização de um *workshop* exclusivo para estudantes do terceiro ano do Ensino Médio, com o objetivo de introduzir o ensino de programação e influenciar as jovens a seguir uma carreira na área de computação. O curso foi ministrado em uma escola pertencente à Rede Estadual de Educação, na cidade de São Carlos. As aulas ocorreram em um laboratório de informática disponível na própria instituição. O programa foi organizado em 12 encontros, realizados aos sábados, onde foram desenvolvidas atividades de programação utilizando a ferramenta Scratch, além de outros ambientes de programação com objetivos semelhantes, como Alice, Game Maker, Kodu e Greenfoot. Também foram realizadas palestras com profissionais da área de Tecnologia da Informação. O conteúdo do curso foi estruturado de

acordo com a complexidade dos temas abordados. Para a coleta de dados, foram aplicados questionários, anotação de atividades e entrevistas. As alunas avaliaram o curso como excelente, destacando sua importância para desenvolver o raciocínio lógico e adquirir conhecimentos sobre Tecnologia da Informação. O único ponto negativo mencionado foi que a ferramenta Scratch foi considerada infantil demais para a faixa etária das participantes. No entanto, elas afirmaram que programar utilizando essa ferramenta foi agradável e que os conteúdos apresentados eram de fácil compreensão.

O curso de verão OST STEM, apresentado por Wang, Vemula e Frye (2020), teve como propósito avaliar o interesse das meninas pela programação antes e após a participação no curso. Nesse programa, a linguagem de programação adotada foi o Python, escolhida com o intuito de oferecer suporte tanto para tarefas simples quanto para as mais complexas. Os conceitos foram introduzidos por meio de cinco projetos baseados em jogos. Com a participação de 48 meninas, o curso foi concluído por 27 delas, resultando em uma melhora nas habilidades, quando comparado o pré-teste com o pós-teste. No entanto, considerando que o objetivo do curso era despertar o interesse pelas carreiras na área, as entrevistas revelaram que as alunas ainda não se sentiam completamente motivadas a ingressar nesse campo, principalmente devido à insegurança em relação às habilidades necessárias.

No estudo realizado por Perin, Silva e Valentim (2022), foi conduzida uma pesquisa de opinião com o objetivo de investigar como a programação em blocos está sendo abordada por professores do Ensino Médio e quais ferramentas eles utilizam para o ensino dessa modalidade de programação. Foi elaborado um questionário online, utilizando a plataforma Google Forms, composto por 10 perguntas de múltipla escolha e duas perguntas foram abertas. Os resultados da pesquisa foram divididos em três partes: dados de caracterização dos docentes; ferramentas de programação em blocos utilizadas para apoiar os processos de ensino e aprendizagem; e dificuldades encontradas no ensino e aprendizagem da programação em blocos. Algumas das dificuldades identificadas no processo de ensino da programação em blocos incluíram a escassez de materiais adequados, o conhecimento limitado sobre as ferramentas de programação em blocos e as dificuldades institucionais em incorporar novas tecnologias no cotidiano escolar. Além disso,

foram observadas dificuldades relacionadas ao pensamento computacional, compreensão do conceito de variáveis, interpretação de problemas, acesso à Internet de qualidade, desenvolvimento de habilidades lógicas e a falta de computadores nas residências dos alunos. Alguns professores também mencionaram aspectos positivos da programação em blocos, como a disponibilidade de plataformas gratuitas, o ensino intuitivo e atrativo proporcionado por essa abordagem e a adequação para fins educacionais. Como projeto futuro, foi sugerido o desenvolvimento de um Sistema de Informação que recomende ferramentas de programação em blocos mais adequadas ao contexto de cada professor, a fim de aprimorar a experiência e a interação desses profissionais.

A plataforma LV4LP foi apresentada como uma ferramenta de aprendizagem assistida para alunos do ensino médio, conforme mencionado por Lin, Yeh e Tan (2022). Seu principal objetivo consiste em ensinar por meio de palestras tradicionais e aulas com programação ao vivo. No estudo realizado, participaram 38 alunos que já haviam cursado pelo menos um semestre de programação fundamental anteriormente. Durante as aulas ministradas na plataforma, divididas em conceitos de programação e exibição de código em tempo real, os alunos tiveram a oportunidade de testar suas habilidades por meio de questões e foram incentivados a realizar anotações. Como resultado, os alunos assistiram a 17.408 vídeos e fizeram 632 anotações. Adicionalmente, foram aplicados um pré-teste e um pós-teste, nos quais foi possível identificar uma melhora nas habilidades dos alunos, especialmente no que se refere à sintaxe.

No artigo de Weintrop e Wilensky (2019), foram estudados os efeitos do ensino introdutório exclusivo por meio de programação baseada em blocos na transição para o ensino de linguagens de programação escritas convencionais. O estudo acompanhou um curso introdutório de programação com duração de 15 semanas para alunos do ensino médio. Foram analisadas duas turmas desse curso, sendo que cada turma passou as primeiras cinco semanas sendo introduzida aos conceitos de lógica de programação por meio de duas versões diferentes do mesmo ambiente educacional: Pencil Code Block, que utiliza a programação em blocos, e Pencil Code Text, que utiliza a programação em texto, mas mantém uma sintaxe relativamente similar à versão em blocos. Nas 10 semanas seguintes, os estudos continuaram com a linguagem de programação Java. Para a avaliação, foram

considerados testes que avaliaram aspectos como aprendizado conceitual e mudanças nas práticas de programação. Os resultados indicaram que, embora a turma que utilizou a linguagem de programação em blocos tenha apresentado um melhor aprendizado inicial, ambas as turmas não apresentaram diferenças no aprendizado durante o processo de transição para o aprendizado com Java.

Tabela 3.1: Comparativo entre trabalhos relacionados.

Trabalho	**Tema**	**Público-Alvo**	**Tipo de Linguagem**	**Avaliação**
Chochiang et al. (2019)	Programação IoT	Alunos do EM	Blocos	Pesquisa de Satisfação
Seralidou e Douligeris (2021)	Plataforma Lúdica	Alunos do EF	Blocos	Pesquisa de Satisfação
Saito et al. (2019)	Curso de Programação	Alunos do EF	Escrita	Testes de Conhecimento
Mattos e Ferreira (2021)	Workshop	Alunos do EM	Blocos e Escrita	Pesquisa de Satisfação
Wang, Vemula e Frye (2020)	Curso de Programação	Alunas do EM	Escrita	Testes de Conhecimento
Perin, Silva e Valentim (2022)	Ambientes de Programação	Professores do EM	Blocos	Pesquisa de Opinião
Lin, Yeh e Tan (2022)	Plataforma de Ensino	Alunos do EM	Escrita	Testes de Conhecimento
Weintrop e Wilensky (2019)	Transição de Bloco para Texto	Alunos do EM	Blocos e Escrita	Testes de Conhecimento

Público-Alvo: *EF* - Ensino Fundamental, *EM* - Ensino Médio.

3.2 Considerações Finais

Os estudos e projetos mencionados exemplificam várias abordagens e estratégias utilizadas para o ensino introdutório de programação em diferentes contextos educacionais. De maneira geral, todos esses trabalhos demonstram ter impactado positivamente o primeiro contato de crianças e jovens do ensino básico com a lógica de programação, de acordo com seus resultados. No entanto, é importante considerar algumas ressalvas a esse respeito. Parte dos estudos apresentados baseou-se exclusivamente em pesquisas de satisfação para sustentar seus resultados, o que pode impedir uma conclusão sólida sobre a efetividade das soluções estudadas, uma vez que a única evidência apresentada consiste na opinião dos respondentes das pesquisas. Os projetos que adotaram uma avaliação mais tradicional do conhecimento se destacaram por fornecer uma perspectiva estatística para corroborar seus resultados. Por fim, nos estudos que utilizam da programação em blocos, nota-se a ausência de um momento em que os estudantes avaliados em cada estudo pudessem ter contato com uma linguagem de programação real para observar a aplicação dos conhecimentos adquiridos, exceto nos estudos realizados por Mattos e Ferreira (2021) e Weintrop e Wilensky (2019), nos quais atividades em ambientes que utilizam linguagens de programação de propósito geral foram realizadas durante os cursos analisados.

O Sistema ELoS relaciona-se a esses trabalhos como um acréscimo à literatura

sobre ambientes introdutórios de lógica de programação. Sua singularidade reside na proposta de utilizar uma linguagem de programação escrita própria e simplificada para auxiliar na futura aplicação dos conhecimentos adquiridos. Além disso, a avaliação desse projeto será focada em uma análise quantitativa, a fim de inferir estatisticamente a qualidade em relação ao cumprimento dos objetivos propostos pelo projeto.

4 Sistema ELoS

Este capítulo apresenta de forma detalhada as funcionalidades do sistema desenvolvido neste trabalho, além de detalhes de sua implementação e uso de tecnologias. Na Seção 4.1, são comentadas as tecnologias utilizadas no desenvolvimento e as justificativas para seu uso. A Seção 4.3 detalha a implementação de uma das principais funcionalidades do sistema: a interpretação do código escrito pelo usuário, de modo que suas ações se reflitam na área gráfica. Por fim, a Seção 4.2 contém a descrição de todas as características e funcionalidades do projeto desenvolvido.

4.1 Tecnologias Utilizadas

Com o objetivo de proporcionar maior acessibilidade, o ELoS foi concebido para ser executado em navegadores *web*, possibilitando assim que o sistema seja compatível com uma variedade de dispositivos com acesso à Internet, tais como computadores, *tablets* e *smartphones*. Para o *design*, foram utilizadas as linguagens de marcação e estilo HTML e CSS, com o auxílio do *framework* de interfaces *web* Bootstrap[1]. A utilização desse *framework* foi fundamental para a adaptabilidade do sistema a diversas telas das plataformas que ele pode suportar. Embora o sistema tenha como plataformas principais de uso *desktops* e *notebooks*, é possível utilizá-lo em dispositivos móveis com poucas limitações.

O sistema foi desenvolvido integralmente em JavaScript e a parte gráfica foi implementada por meio da biblioteca ThreeJS[2]. Além da geração da interface 3D, a programação em JavaScript no sistema envolve a geração dinâmica de efeitos e elementos gerais das páginas, bem como a interpretação do código escrito pelo usuário.

O Sistema ELoS encontra-se disponível em https://avrgroup.github.io/ELoS/ e seu código fonte pode ser baixado em https://github.com/AVRGroup/ELoS.

[1] ⟨https://getbootstrap.com/docs/5.0/getting-started/introduction/⟩
[2] ⟨https://threejs.org⟩

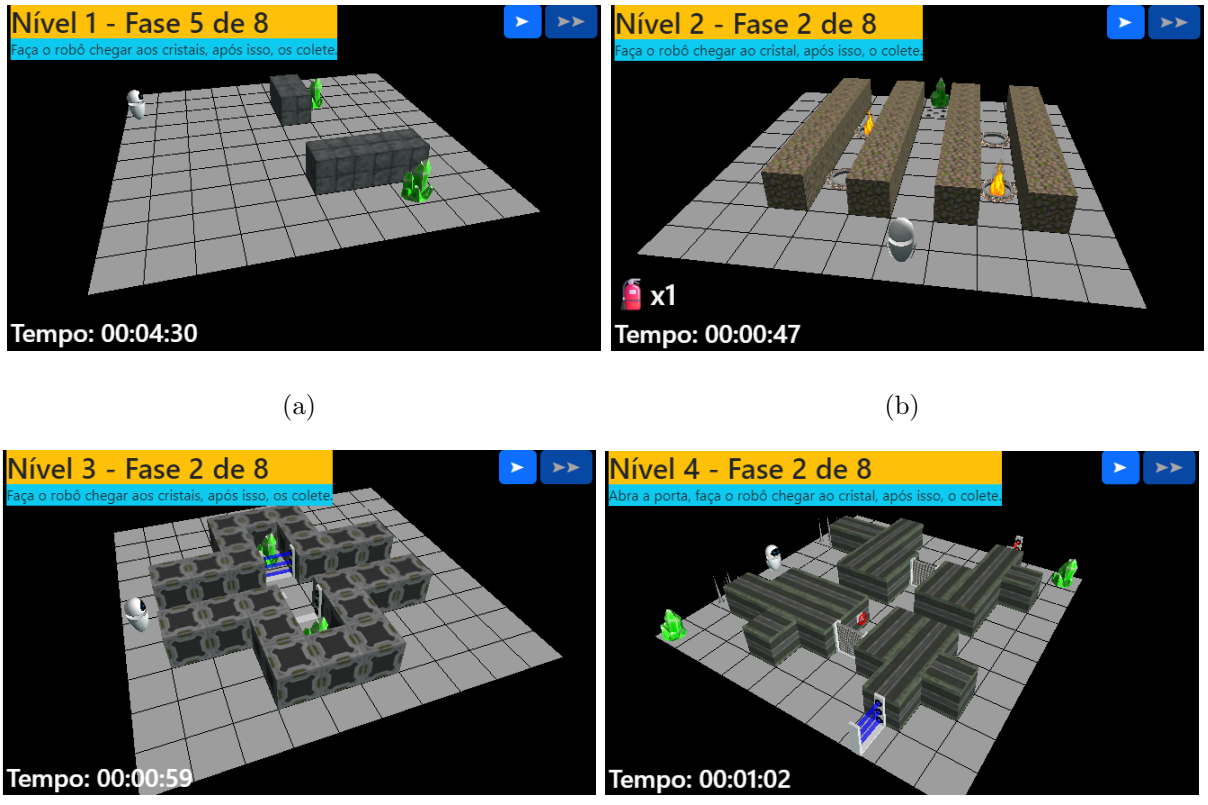
4.2 Apresentação do Sistema

Para introduzir os alunos à lógica de programação de maneira lúdica e divertida, o Sistema ELoS foi concebido para apresentar os conceitos básicos de lógica de programação de forma semelhante a videogames. Cada tópico de ensino representa um nível do jogo. O Nível 1 representa o conceito de sequência simples, o Nível 2 aborda estruturas condicionais simples, o Nível 3 estruturas condicionais com alternativa dupla e o Nível 4 apresenta o conceito de estruturas de repetição. Cada nível possui oito fases, que consistem em desafios apresentados em um sistema gráfico. A dificuldade dos desafios aumenta progressivamente. Para resolver os desafios, o usuário deve digitar os comandos necessários em uma caixa de texto. Os comandos foram estruturados de forma a se assemelharem a uma linguagem de programação genérica. Em todos os níveis, o usuário tem fácil acesso a todos os comandos disponíveis. Ao executar os comandos, que serão explicitados mais adiante neste capítulo, o sistema gráfico reage ao que foi programado, fornecendo *feedbacks* em forma de texto em um console.

O sistema foi desenvolvido para ser executado em navegadores *web*. Ao acessar o sistema, o usuário é direcionado para a página inicial (Figura 4.1), onde é possível escolher o nível a ser completado e assistir a vídeos de apresentação do projeto e breves tutoriais sobre como utilizar os recursos de cada nível do sistema para completar os desafios. Tanto a página inicial quanto os níveis possuem suporte para as línguas portuguesa e inglesa.

Figura 4.1: Página inicial do Sistema ELoS

Para o *design* dos níveis, a página foi dividida em duas partes. A primeira metade consiste na parte gráfica, apresentada na Figura 4.2, onde no canto superior esquerdo é apresentado o nível e a fase atual e abaixo são fornecidos os objetivos a serem alcançados no desafio proposto. No canto superior direito há dois botões que ajustam a velocidade de execução dos comandos. Ao completar uma fase, uma mensagem de sucesso é exibida logo abaixo dos objetivos, juntamente com um botão para avançar para a próxima fase. Ao concluir todas as fases do nível, clicar nesse botão redireciona o usuário para a página inicial.



(a) (b)

(c) (d)

Figura 4.2: Parte gráfica do Sistema ELoS. Em (a) há um exemplo de uma fase do Nível 1, em (b) um exemplo do Nível 2, em (c) do Nível 3 e em (d) um exemplo de fase do Nível 4.

A segunda metade apresenta a área em que o usuário pode interagir (Figura 4.3). Na parte superior, há uma caixa que pode ser expandida para exibir todos os comandos que o usuário pode digitar para completar a fase. É possível clicar nos comandos apresentados, e ao fazer isso, um *template* do comando selecionado é adicionado na caixa de texto. No centro, há uma caixa de texto onde o usuário pode digitar os comandos. Essa

caixa de texto segue o *design* e as mecânicas padrão de vários editores de linguagens de programação, com coluna de marcação de linhas, tabulação automática e indicadores de início e fim de blocos, entre outros recursos. Ao terminar de escrever o código, o usuário pode utilizar o botão "Executar" para observar a execução do código na parte gráfica. Durante a execução, os comandos que estão sendo executados no momento recebem uma marcação amarela para melhor relacionar os comandos utilizados na área de código com as ações realizadas na área gráfica. Caso o usuário queira interromper a execução e voltar ao ponto inicial da fase, basta pressionar o botão "Reset". O uso do botão "Executar" também retorna a fase ao seu estado inicial antes de executar o código. Por fim, na parte inferior, há um console que fornece *feedbacks* em texto sobre a execução do código digitado, apresentando mensagens indicando quando o código é inválido para execução, quando ocorre algo inesperado durante a execução ou quando ações são realizadas com sucesso. Há a possibilidade de apagar os textos gerados neste console através do botão "Limpar Console".

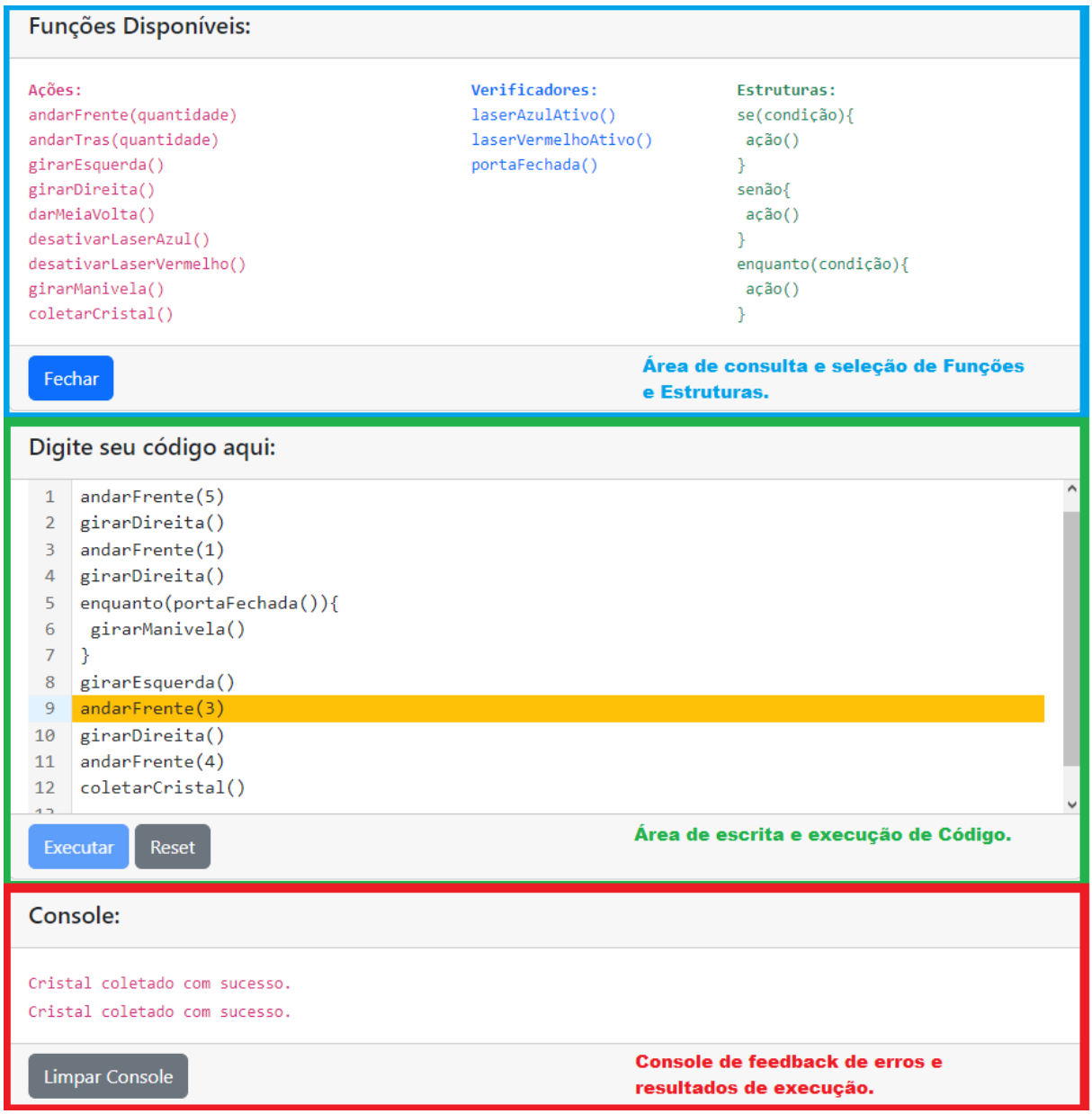


Figura 4.3: Área de interação do usuário.

4.3 Análise e Execução do Código do Usuário

A interpretação do código ocorre quando o usuário clica no botão "Executar" da interface. Neste momento, uma *string* contendo o código digitado passa por uma função que irá analisar linha a linha o que foi digitado e verificará o padrão dos comandos em conformidade com o que é esperado ser escrito. A análise do padrão é realizada com o auxílio do construtor de expressões regulares do Javascript, o *RegExp*. Quando uma linha se ajusta

ao padrão esperado pela linguagem, ela é "reescrita" de forma a permitir a interpretação pelas funções internas do sistema e é concatenada em uma nova *string* que representará a versão a ser interpretada em JavaScript.

Algumas estruturas do código requerem uma análise adicional, além de seguir o padrão proposto pelas expressões regulares. Um exemplo disso são as estruturas condicionais e de repetição, para as quais é feita uma verificação se o que é colocado como condição de início é válido e se este código se encaixa nas funções chamadas de "Verificadores" no sistema. Além disso, no caso de uma estrutura apresentar um bloco de execução, o código do usuário é percorrido para verificar se esse bloco foi fechado corretamente.

Uma última funcionalidade deste analisador é avaliar a necessidade de chamadas de funções chamadas de "*badLuck*". Em determinados níveis do sistema, como aqueles que envolvem o uso de condicionais, era inicialmente possível completar as fases sem utilizar as estruturas abordadas no nível, desde que o usuário acertasse o momento certo para pressionar o botão "Executar". Para evitar isso, o *parser* analisa o contexto no qual o usuário realiza um comando sem usar as estruturas esperadas e chama essa função para readequar a fase, trocando sutilmente os estados de alguns obstáculos no cenário, tornando impossível completar o nível sem usar as estruturas abordadas.

Se o usuário não cometeu nenhum erro sintático, uma *string* contendo o código do usuário reescrito em JavaScript é retornada para que ela seja avaliada pelas funções internas de interpretação de código da linguagem. O diagrama mostrado na Figura 4.4 ilustra um exemplo de como o código do usuário fica após ser convertido para a versão a ser executada pelo sistema. Caso o usuário tenha cometido algum erro sintático, a análise do código é interrompida quando ocorre um erro, e esse erro é exibido no "Console" do sistema para que o usuário possa corrigi-lo.

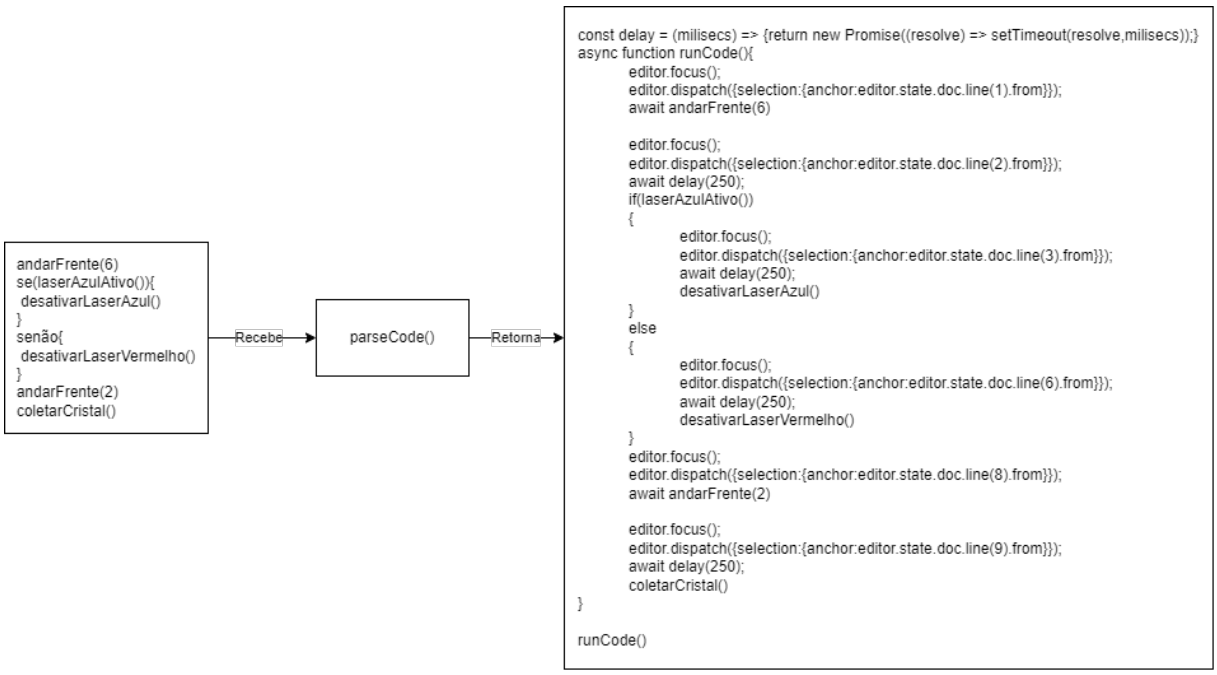


Figura 4.4: Um exemplo de como o código escrito pelo usuário é transformado em um código que o sistema pode executar de forma a produzir todos os efeitos necessários na interface do nível.

5 Avaliação do Sistema

O presente capítulo detalha todo o processo de análise do uso do sistema quando submetido a um ambiente estudantil, como parte de uma atividade que complementa o ensino introdutório de lógica de programação. A Seção 5.1 apresenta todos os processos de planejamento, execução, caracterização dos participantes e análise dos dados que foram produzidos pelo estudo realizado, além de identificar possíveis ameaças à validade do estudo. Por fim, a Seção 5.2 encerra o capítulo resumindo tudo o que foi apresentado no estudo e discutindo possíveis inferências sobre os resultados.

5.1 Estudo Experimental

Com o intuito de avaliar a viabilidade do ELoS como uma ferramenta de ensino de programação para iniciantes, foi conduzido um estudo experimental. O objetivo principal desta pesquisa é analisar o funcionamento do ELoS e sua eficácia no processo de aprendizagem dos estudantes que estão dando os primeiros passos na programação. A abordagem consistiu em testes com alunos ingressantes do ensino superior, permitindo que eles participassem do processo de aprendizagem por meio do sistema. Essa metodologia visa envolver os alunos de forma mais engajada e motivadora, proporcionando uma experiência de aprendizado divertida e imersiva. Os resultados desses testes contribuirão para a obtenção de *insights* mais precisos sobre a eficácia do ELoS como uma ferramenta de ensino de programação, levando em consideração a interação dos alunos com os jogos e o impacto em sua aprendizagem.

5.1.1 Definição e Planejamento

Durante as aulas introdutórias de programação na Universidade Federal de Juiz de Fora, o ELoS foi divulgado e implementado como parte do currículo. Para coletar dados dos alunos, foi aplicado um questionário que poderia ser respondido a qualquer momento ou durante a conclusão das 8 fases do Nível 1. Por meio desse questionário, foi possível

obter informações como a fase completada, nome, idade, tempo de conclusão, gênero e experiência prévia do aluno com programação. Esses dados foram essenciais para análise e avaliação do desempenho dos alunos, bem como para a obtenção de informações relevantes sobre o uso do ELoS como ferramenta de ensino.

5.1.2 Execução do Estudo

Inicialmente, o sistema foi introduzido em 14 turmas da disciplina DC5199 (Algoritmos - Prática) da instituição, envolvendo diversos cursos da área de exatas. Dos alunos participantes, foram recebidas um total de 263 respostas ao questionário das fases. No entanto, após validação e análise, constatamos que apenas 229 respostas foram consideradas válidas e utilizadas como base de dados para o estudo experimental. As 34 entradas removidas consistiam de respostas mal preenchidas e respostas repetidas originadas de um mesmo participante.

Ao concluir a fase 8, que é a última fase do Nível 1, os participantes tiveram a oportunidade de preencher o questionário. Além disso, o questionário também estava disponível para aqueles que decidissem desistir em qualquer fase, todavia neste experimento todos os participantes finalizaram o Nível 1. Por meio dele, foi possível coletar dados dos participantes e dividir as respostas em grupos com base nas informações recebidas como fase completada, idade, tempo de conclusão, gênero e experiência prévia do aluno com programação. Essa segmentação permitiu uma análise mais detalhada e uma compreensão dos resultados, considerando diferentes variáveis que podem influenciar o desempenho e a percepção dos alunos em relação ao ELoS.

5.1.3 Caracterização dos Participantes

Os participantes dos estudo experimental são recém ingressos da UFJF no nível de graduandos. Todos os participantes estão cursando a disciplina DC5199 (Algoritmos - Prática).

As Tabelas 5.1, 5.2 e 5.3 apresentam a distribuição dos participantes. Entre os 229 participantes, 68 são do gênero feminino e 161 do gênero masculino, conforme demonstrado na Tabela 5.1.

Na Tabela 5.2, observa-se que a faixa etária predominante dos alunos que cursam

Tabela 5.1: Distribuição dos participantes de acordo com o gênero.

Gênero	**Número de Pessoas**
Feminino	68
Masculino	161
Total	229

a disciplina e participaram do questionário é entre 17 e 18 anos. Por outro lado, a menor faixa etária está representada entre os alunos de 22 e 23 anos.

Tabela 5.2: Distribuição dos participantes de acordo com a idade.

Idade	**Número de Pessoas**
17 a 18 anos	162
19 a 21 anos	53
22 a 23 anos	6
Acima de 23 anos	8
Total	229

Por último, em relação à experiência prévia com programação, observamos que a maioria dos participantes teve seu primeiro contato com a disciplina, conforme indicado na Tabela 5.3. O próximo valor mais significativo corresponde aos alunos que já tiveram experiência programando pequenos projetos.

Tabela 5.3: Distribuição dos participantes de acordo com a experiência.

Experiência	**Número de Pessoas**
Este foi meu primeiro contato	126
Tive uma pequena experiência no colégio	41
Já programei pequenos projetos	56
Sou um programador experiente	6
Total	229

5.1.4 Ameaças à Validade

Durante o planejamento deste estudo experimental, foram tomadas medidas para reduzir possíveis influências ou restrições que poderiam afetar a confiabilidade dos resultados obtidos. No entanto, não é possível garantir que tais influências não tenham impactado os resultados. Portanto, a seguir são descritas as influências identificadas neste contexto

de estudo experimental.

Uma das influências diz respeito à validade interna do estudo, o que envolve a possibilidade de variáveis não controladas afetarem os resultados. Por exemplo, fatores externos, como outras atividades extracurriculares dos alunos realizadas em paralelo ao experimento, podem interferir em sua participação e desempenho no ELoS, o que poderia afetar os resultados do estudo.

Além disso, não há controle sobre a forma como os participantes interagem com o ELoS fora do ambiente controlado do estudo. Os alunos podem buscar informações adicionais ou apoio de terceiros, o que pode influenciar os resultados e afetar a validade dos dados coletados.

O tamanho da amostra é limitado, o que não é ideal do ponto de vista estatístico. Portanto, os resultados do estudo experimental não são conclusivos, apenas indicativos. No entanto, o número de participantes acima de 30 é considerado alto, o que aumenta a validade estatística das conclusões obtidas.

5.1.5 Avaliação Quantitativa

O objetivo deste estudo é avaliar os dados obtidos através de um formulário respondido pelos próprios usuários ao final do Nível 1. O formulário solicitava dados como nome, idade, gênero e experiência com a programação, sendo a última uma questão objetiva, composta por quatro opções de resposta, sendo elas: este foi meu primeiro contato; tive uma pequena experiência no colégio; já programei pequenos projetos; sou um programador experiente.

Nota-se, através da Tabela 5.3, que a maior parte dos participantes tiveram sua primeira experiência com programação ao utilizar o ELoS. Esses participantes tiveram a maior média de tempo para completar o Nível 1, conforme ilustra a Figura 5.1. Os resultados da Figura 5.1 mostram que usuários que nunca programaram apresentam mais dificuldades em realizar as tarefas propostas em relação a usuários que já tiveram contato com a programação. Nota-se também, através das barras de erro de desvio padrão da Figura 5.1 que o conjunto de dados dos participantes que possuem o maior nível de experiência estão mais próximos da média de seu conjunto. Isso significa que os partici-

pantes desse grupo completaram o Nível 1 do ELoS em tempos semelhantes. É possível perceber também que mesmo sendo o segundo maior conjunto de dados da Tabela 5.3, os participantes que programaram pequenos projetos possuem a maior barra de erro. Isso mostra que esses dados se encontram mais afastados da média do conjunto, ou seja, nesse grupo, o tempo para completar o Nível 1 destoa consideravelmente de um participante para outro.

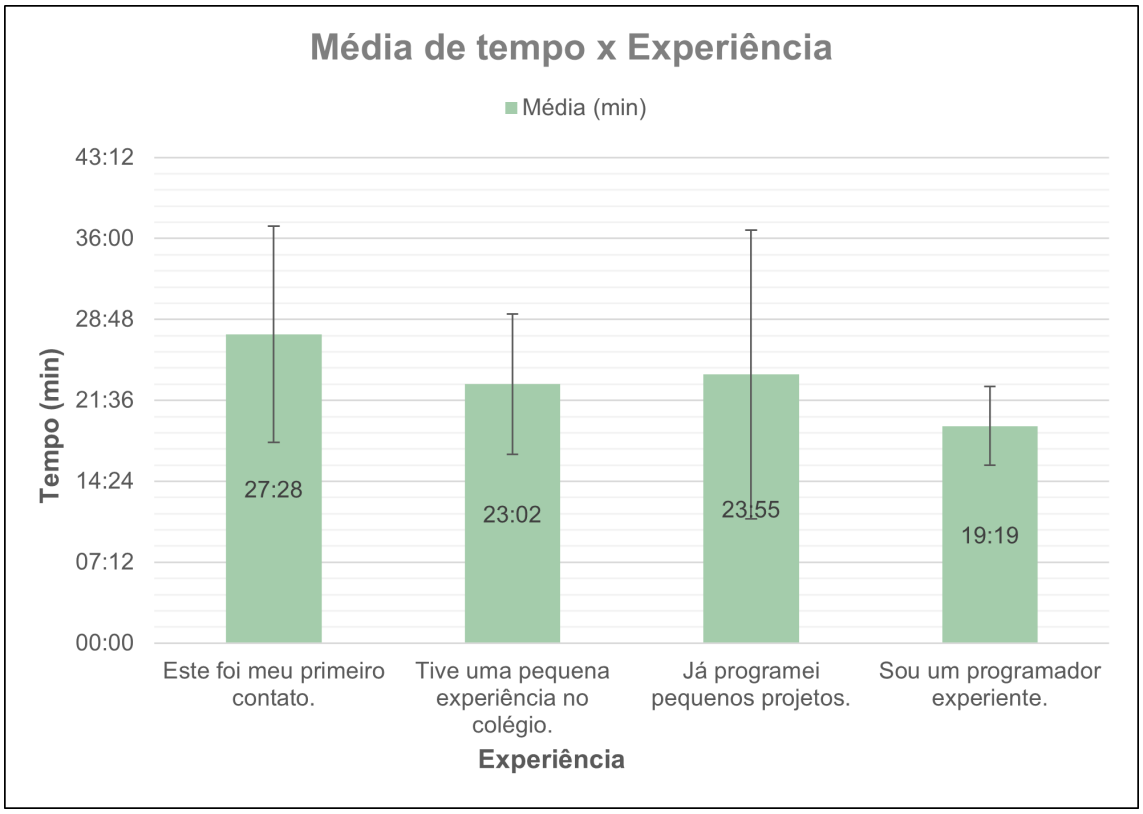


Figura 5.1: Gráfico da média de tempo pela experiência. Quanto menor o tempo, melhor.

A faixa etária predominante é de 17 a 18 anos, como é possível observar na Tabela 5.2. Conforme a Figura 5.2, a maior média de tempo para completar o nível 1 foi dos participantes acima de 23 anos e a menor média foi dos participantes de faixa etária de 22 à 23 anos. Para entender esses resultados, foi feito um gráfico relacionando a experiência com a faixa etária, ilustrado na Figura 5.3. É possível perceber que a grande maioria dos participantes com a faixa etária de 22 à 23 anos já programaram pequenos projetos ou são programadores experientes, já a grande maioria dos participantes acima de 23 anos tiveram seu primeiro contato com a programação com o ELoS.

Através das barras de erro de desvio padrão da Figura 5.2, é possível perceber

que os dados de tempo dos participantes com a faixa etária de 22 a 23 anos estão próximos da média do conjunto. Isso se dá pelo fato de 5 dos 6 participantes terem experiência com programação, logo, não houve grande diferença nos dados de tempo que esses participantes levaram para completar o nível 1. Ainda que um dos participantes do grupo em questão tenha iniciado sua experiência com programação ao utilizar o ELoS, o tempo gasto por ele para completar o nível 1 não fez grande diferença no cálculo da média, já que representa apenas 16,7% dos dados do grupo. Nota-se também, através da Figura 5.3, que os participantes de faixa etária de 17 a 18 anos possuem diferentes graus de experiência com programação, o que aumenta a diferença no tempo para conclusão do nível 1 de um participante para outro; por consequência, a barra de desvio padrão desse grupo é a maior, como observado na Figura 5.2. Para interpretar as barras de desvio padrão da Figura 5.2 foi feita a exclusão de um dado referente a amostra de faixa etária acima de 23 anos. O dado excluído foi considerado um *outlier*, por apresentar um grande afastamento dos demais dados, o que poderia afetar negativamente todo o resultado da análise.

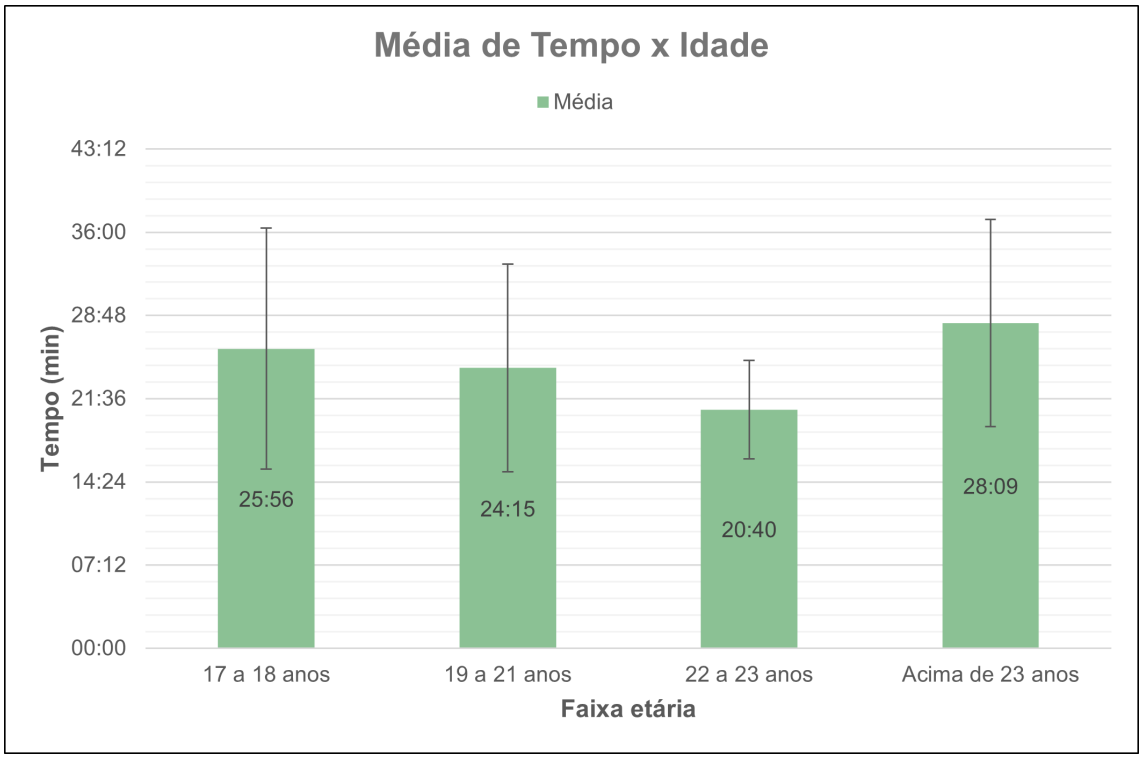


Figura 5.2: Gráfico da média de tempo pela idade.

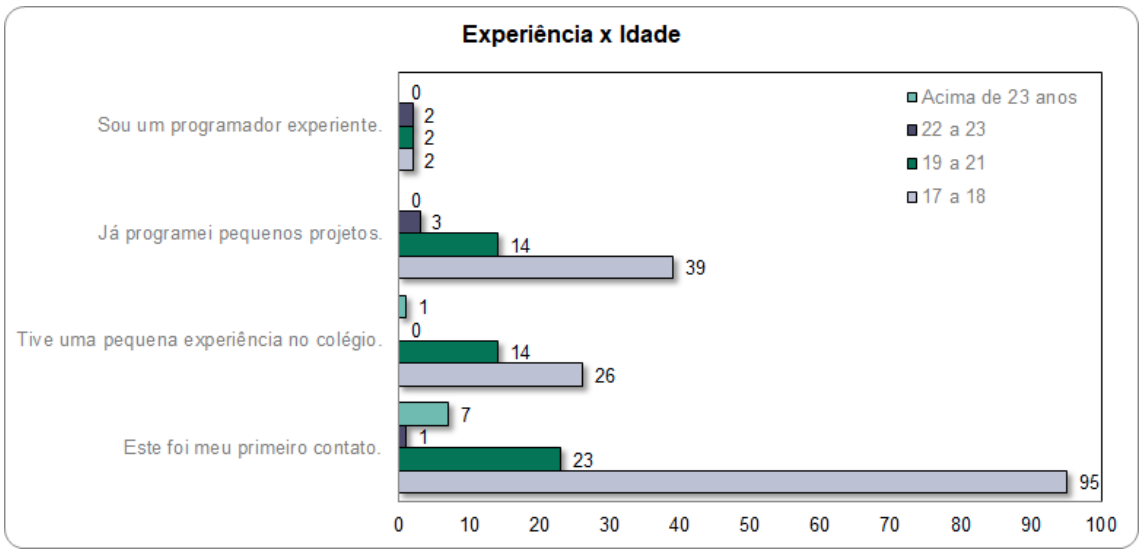


Figura 5.3: Gráfico da experiência pela idade

5.1.6 Correlação entre as Variáveis

Coeficientes de correlação são medidas estatísticas postas para examinar a relação entre duas variáveis. Eles desempenham um papel fundamental na análise estatística, permitindo-nos quantificar a força e a direção dessa relação e identificar padrões e tendências nos dados.

O coeficiente de correlação frequentemente usado é o coeficiente de correlação de Pearson, também chamado de correlação linear. Este coeficiente avalia a relação linear entre duas variáveis contínuas, variando de -1 a 1. Um valor de -1 indica uma correlação negativa perfeita, ou seja, à medida que uma variável aumenta, a outra diminui linearmente, enquanto um valor de 1 indica uma correlação positiva perfeita, logo, quando uma variável aumenta, a outra também aumenta linearmente. Um valor de 0 indica que não há correlação linear entre as variáveis. Esse coeficiente é útil quando as variáveis têm uma distribuição aproximadamente normal e existe uma relação linear entre elas.

Outro coeficiente de correlação amplamente utilizado é o coeficiente de correlação de Spearman. Ao contrário do coeficiente de Pearson, o coeficiente de Spearman avalia a relação monotônica entre as variáveis, ou seja, se a relação entre elas é crescente ou decrescente, se é linear ou não. Este coeficiente é baseado na ordem dos valores das variáveis e é particularmente útil quando os dados não seguem uma distribuição normal

ou quando existem *outliers* que podem afetar a análise.

Os coeficientes de correlação têm muitas utilizações em diversos campos. No sistema de saúde, eles são usados para avaliar a relação entre variáveis clínicas, como índice de massa corporal, e o risco de doenças cardíacas. Na economia, eles são usados para analisar a relação entre variáveis financeiras, como taxa de desemprego e inflação. Na psicologia, eles são usados para entender a relação entre variáveis psicológicas, como autoestima e bem-estar emocional. Neste estudo, eles foram utilizados para avaliar uma possível relação entre as variáveis idade, tempo e experiência.

Para fazer a análise dos dados, foi criada uma matriz de correlações, ilustrada na Tabela 5.4. Os valores com asterisco (*), referentes às correlações, significam que as medidas são estatisticamente significativas, pois os valores de p-value são menores que 0,05. Dessa forma, é possível fazer uma análise sobre esses resultados.

Tabela 5.4: Matriz de Correlações.

		Idade	**Tempo**	**Experiência**
Idade	R de Pearson	--		
	p-value	--		
	Rho de Spearman	--		
	p-value	--		
Tempo	R de Pearson	0,099	--	
	p-value	0,136	--	
	Rho de Spearman	-0,048	--	
	p-value	0,467	--	
Experiência	R de Pearson	0,022	-0,192*	--
	p-value	0,739	0,004	--
	Rho de Spearman	0,096	-0,293*	--
	p-value	0,148	0,001	--

Nota-se através da Tabela 5.4 que as únicas variáveis que possuem algum tipo de correlação são as variáveis experiência e tempo. Como os valores são negativos em ambos os coeficientes, à medida que uma variável aumenta, a outra diminui, indicando que um usuário mais experiente completaria o nível 1 do ELoS em menos tempo. Entretanto, mesmo que as medidas sejam significativas, as intensidades das correlações entre as variáveis são consideradas fracas, pois não passam de 0,3. Desse modo, não podemos afirmar que a experiência de um usuário influenciou no seu tempo para completar o nível 1 do sistema.

5.2 Discussão

O sistema foi implementado inicialmente em 14 turmas da disciplina DC5199 (Algoritmos - Prática) na Universidade Federal de Juiz de Fora. Essas turmas consistem em estudantes de diversos cursos da área de exatas, o que resulta em uma participação diversificada.

No total, foram coletadas 263 respostas no questionário de fases, mas apenas 229 foram validadas e utilizadas como base de dados para análise do estudo experimental. Essas informações possibilitaram a divisão dos participantes em grupos com base em critérios como idade, tempo de conclusão, gênero e experiência prévia em programação.

Ao examinarmos a faixa etária dos participantes, constatamos que a maioria está entre 17 e 18 anos, indicando que o ELoS foi utilizado por alunos em estágios iniciais de suas trajetórias acadêmicas.

No que se refere à experiência prévia dos participantes em programação, verificamos que a maioria teve o primeiro contato com o ELoS, enquanto alguns relataram ter realizado pequenos projetos de programação anteriormente. Isso sugere que o sistema atendeu tanto a alunos iniciantes quanto àqueles com alguma experiência prévia, oferecendo uma oportunidade de aprendizado para diversos perfis de estudantes.

É importante ressaltar que, ao planejar o estudo experimental, foram identificadas possíveis ameaças à validade dos resultados, como a falta de controle sobre as circunstâncias individuais nas quais cada participante realizou o estudo e a influência de variáveis não controladas nos resultados.

Com base nessas informações e considerando as características e resultados do ELoS, podemos afirmar que o sistema demonstrou potencial para auxiliar o ensino e aprendizagem de programação para iniciantes, atendendo a diferentes necessidades e perfis de estudantes. No entanto, são necessárias pesquisas e análises adicionais para obter conclusões mais abrangentes e compreender melhor o impacto e a eficácia do ELoS no contexto educacional.

6 Conclusão

O projeto ELoS tem como objetivo desenvolver um ambiente lúdico introdutório para o ensino de lógica de programação para crianças e jovens. A plataforma escolhida para o projeto foi a *web*, enquanto a temática lúdica baseou-se em dinâmicas de jogos eletrônicos.

A partir dessas escolhas, o desenvolvimento de um primeiro protótipo foi iniciado. O *design* do sistema utiliza HTML e CSS, com o suporte do *framework* Bootstrap para garantir adaptabilidade a diferentes telas. Embora *desktops* e *notebooks* sejam as plataformas principais, é possível usar o sistema em dispositivos móveis com poucas limitações. O sistema foi programado integralmente em JavaScript, abrangendo a geração de efeitos dinâmicos, elementos das páginas, interpretação do código do usuário, além da geração da interface 3D através da bibloteca gráfica ThreeJS.

Posteriormente, foi realizado um experimento com alunos de cursos da área de exatas recém-ingressos na UFJF, que testaram o primeiro nível do sistema como atividade prática da disciplina DC5199 (Algoritmos - Prática). O objetivo era avaliar o nível de dificuldade dos desafios oferecidos pelo sistema, correlacionando o tempo de realização da atividade com a experiência dos alunos em lógica de programação e demais dados de perfil, como gênero e idade. Ao analisar os dados coletados do experimento, observou-se a correlação de que os estudantes que tinham menos experiência com programação levaram mais tempo para concluir o Nível 1 do sistema. Outras correlações não apresentaram resultados que pudessem inferir alguma conclusão.

De forma geral os resultados indicaram uma boa recepção inicial do projeto, que demonstrou potencial para auxiliar o ensino e aprendizagem de programação para iniciantes. Essas informações, aliadas aos *feedbacks* coletados, servirão de base para a continuidade do desenvolvimento deste projeto em trabalhos futuros.

Sobre este projeto foi produzido um artigo que foi apresentado no XII Congresso Brasileiro de Informática na Educação (CBIE) e será publicado nos Anais do Simpósio Brasileiro de Informática na Educação 2023 (SBIE).

6.1 Trabalhos Futuros

Este é um trabalho inicial acerca do desenvolvimento e análise de um software educacional lúdico para apoio do ensino de lógica de programação. Sugerem-se, portanto, como trabalhos futuros, o contínuo aprimoramento do ELoS com base nos *feedbacks* dos alunos e em análises mais detalhadas de sua eficácia. Além disso, propõe-se a expansão do sistema para outras áreas do conhecimento e aprofundamento em estudos sobre o impacto do ELoS no desempenho dos alunos.

Outra sugestão é a realização de experimentos em escolas que possuam laboratórios de informática, direcionados a crianças e jovens do ensino básico que tenham pouco ou nenhum conhecimento prévio em lógica de programação. Por fim, com a implementação do projeto nas escolas, pretende-se proporcionar momentos, com o auxílio de professores da área de computação, nos quais os alunos possam relacionar o conteúdo introduzido pelo ELoS com o que é aplicado em um contexto real.

Bibliografia

CHOCHIANG, K.; CHAOWANAWATEE, K.; SILANON, K.; KLIANGSUWAN, T. Arviz: An iot teaching tool for high school students. In: IEEE. *2019 23rd International Computer Science and Engineering Conference (ICSEC)*. 2019. p. 87–91.

DISALVO, B. Graphical qualities of educational technology: Using drag-and-drop and text-based programs for introductory computer science. *IEEE computer graphics and applications*, IEEE, v. 34, n. 6, p. 12–15, 2014.

KUZ, A.; ARISTE, M. C. Análise e revisão de softwares educacionais para a aprendizagem da programação em ambientes lúdicos. *Tecné, Episteme y Didaxis: TED*, n. 52, p. 117–136, 2022.

LIN, Y.-T.; YEH, M. K.-C.; TAN, S.-R. Teaching programming by revealing thinking process: Watching experts' live coding videos with reflection annotations. *IEEE Transactions on Education*, IEEE, v. 65, n. 4, p. 617–627, 2022.

MARCOLINO, A.; BARBOSA, E. F. Softwares educacionais para o ensino de programaçao: Um mapeamento sistemático. In: *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. 2015. v. 26, n. 1, p. 190.

MATTOS, F. de; FERREIRA, V. Despertando o interesse em meninas pela computação com o ensino de programação em um ambiente apoiador. *Revista Psicologia e Transdisciplinaridade*, v. 1, n. 1, p. 39–58, 2021.

MLADENOVIĆ, M.; BOLJAT, I.; ŽANKO, Ž. Comparing loops misconceptions in block-based and text-based programming languages at the k-12 level. *Education and Information Technologies*, Springer, v. 23, p. 1483–1500, 2018.

MOORS, L.; LUXTON-REILLY, A.; DENNY, P. Transitioning from block-based to text-based programming languages. In: IEEE. *2018 International Conference on Learning and Teaching in Computing and Engineering (LaTICE)*. 2018. p. 57–64.

PERIN, A. P. J.; SILVA, D. E.; VALENTIM, N. M. C. Investigating the teaching of block programming in high school. In: *XVIII Brazilian Symposium on Information Systems*. 2022. p. 1–9.

SAITO, D.; WASHIZAKI, H.; FUKAZAWA, Y.; YOSHIDA, T.; KANEKO, I.; KAMO, H. Learning effects in programming learning using python and raspberry pi: Case study with elementary school students. In: IEEE. *2019 IEEE International Conference on Engineering, Technology and Education (TALE)*. 2019. p. 1–8.

SERALIDOU, E.; DOULIGERIS, C. Motivating students in distance programming learning using games. In: IEEE. *2021 6th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)*. 2021. p. 1–7.

WANG, C.; VEMULA, S.; FRYE, M. Out-of-school time stem: Teach programming using python for high school girls. In: IEEE. *2020 IEEE Integrated STEM Education Conference (ISEC)*. 2020. p. 1–6.

WEINTROP, D.; WILENSKY, U. To block or not to block, that is the question: students' perceptions of blocks-based programming. In: *Proceedings of the 14th international conference on interaction design and children*. 2015. p. 199–208.

WEINTROP, D.; WILENSKY, U. Transitioning from introductory block-based and text-based environments to professional programming languages in high school computer science classrooms. *Computers & Education*, v. 142, p. 103646, 2019.