e-Xacta

ISSN: 1984-3151

UMA ABORDAGEM CONSTRUTIVISTA NO ENSINO DE ALGORITMOS E LÓGICA DE PROGRAMAÇÃO COM O AUXÍLIO DE UMA FERRAMENTA GAMIFICADA

A CONSTRUCTIVIST APPROACH TO SUPPORT THE TEACHING AND LEARNING PROCESSES OF ALGORITHMS AND PROGRAMMING LOGIC, USING A GAMIFIED TOOL

Rodrigo Perlin[1], Ricardo Tombesi Macedo[2], Sidnei Renato Silveira[3]

1 Graduando em Sistemas de Informação. Universidade Federal de Santa Maria - UFSM - Campus Frederico Westphalen/RS. rodrigo_perlin@hotmail.com.

2 Doutor em Ciência da Computação pela Universidade Federal do Paraná. 2016. Professor Adjunto do Departamento de Tecnologia da Informação da UFSM – Campus Frederico Westphalen/RS. rmacedo1987@gmail.com.

3 Doutor em Ciência da Computação pela Universidade Federal do Rio Grande do Sul. 2006. Professor Adjunto do Departamento de Tecnologia da Informação da UFSM – Campus Frederico Westphalen/RS. sidneirenato.silveira@gmail.com.

Recebido em 20/03/2019; Publicado em 23/06/2019

RESUMO: Ao analisar os esforços para apoiar os processos de ensino e de aprendizagem de algoritmos e lógica de programação, encontram-se estudos envolvendo a aplicação de diferentes ferramentas, tais como o Scratch e o Algo+. Além disso, existem trabalhos que propõem uma reorganização dos conteúdos e a aplicação de metodologias de ensino inovadoras. Nesse contexto, este artigo propõe uma abordagem para apoiar os processos de ensino e de aprendizagem de algoritmos e lógica de programação baseada na teoria construtivista, utilizando a ferramenta P.e.p.y, a qual implementa o conceito de gamificação. Para validar essa proposta, bem como a ferramenta desenvolvida, foi realizado um estudo de caso. A aplicação dos instrumentos no início e no final do estudo de caso, apontam percentuais elevados de compreensão dos conceitos de lógica de programação e da linguagem de programação Python. Os resultados apontam que a aplicação da ferramenta auxiliou os alunos a desenvolver o pensamento computacional, uma área que vem sendo estimulada pela SBC (Sociedade Brasileira de Computação) e que a abordagem proposta estimula os processos de ensino e de aprendizagem por meio da ferramenta P.e.p.y.

PALAVRAS-CHAVE: Aprendizado de Lógica de Programação. Abordagem Construtivista. Ferramenta P.e.p.y.

ABSTRACT: By analyzing efforts to support learning process applied to logic and programming courses, there are studies involving the employment of different tools, such as Scratch and Algo+. Besides, there are works proposing the content reorganization and the employment of innovative teaching methodologies. In this context, this paper proposes an approach to support algorithms learning process based on constructivist theory through the use P.e.p.y tool, which implements the gamification concept. In order to validate this proposal, as well as the implemented tool, it was performed a case study. The instruments application in the beginning and in the end of the case study indicates elevated perceptual of comprehension of logic and Python programming languages concepts. Results indicate that the tool application supported students to develop the computational thinking, a field highly stimulated by the SBC, the Brazilian Computing Society, and that the proposed approach stimulates the learning processes through the P.e.p.y tool employment.

KEYWORDS: Learning Processes of Programming Logic. Constructivist Theory. P.e.p.y Tool.

---

1 INTRODUÇÃO

Atualmente, para formar profissionais qualificados na área de informática, torna-se indispensável uma base sólida de conhecimentos acerca de algoritmos e de lógica de programação. O domínio dos conteúdos pertencentes à disciplina de algoritmos serve como base para a prática da programação de computadores, a qual tem como finalidade automatizar processos e as atividades para facilitar/otimizar o dia-a-dia de pessoas e empresas (GUSMÃO, 2011). As projeções da empresa *VisionMobile*, compiladas pela Universidade de *Richmond*, estimam que, em 2020, existirá uma demanda de quinze vezes mais profissionais especializados em programação relacionada com a Internet das Coisas (EL PAIS, 2016). Além disso, a disciplina de algoritmos consiste no eixo central para todas as áreas relacionadas com a computação (SCAICO, 2013). A SBC (Sociedade Brasileira de Computação) também destaca que o ensino de conceitos básicos de computação nas escolas é fundamental, permitindo que as crianças e adolescentes construam o pensamento computacional, importante na sociedade do conhecimento (SBC, 2017; SBC, 2018).

O pensamento computacional se refere à capacidade de sistematizar, representar, analisar e resolver problemas por meio da construção de *algoritmos*. Apesar de ser um termo recente, vem sendo considerado como um dos pilares fundamentais do intelecto humano, junto com a leitura, a escrita e a aritmética, pois, como estas, serve para descrever, explicar e modelar o universo e seus processos complexos. O pensamento computacional envolve abstrações e técnicas diferentes das aprendidas na Matemática, necessárias para a descrição e análise de informações (dados) e processos (SBC, 2017).

Entretanto, um problema frequente nos cursos de computação consiste no alto nível de evasão. Grande parte dessa desistência é ocasionada pelo nível de dificuldade no aprendizado de algoritmos (HOED, 2017). Embora a demanda por profissionais da área de Tecnologia da Informação (TI) cresça cada vez mais, pois cada vez mais setores diferentes buscam soluções de seus problemas com o apoio da TI, existe uma diminuição na procura por cursos superiores na área (CABRAL *et al.*, 2007; COMPUTAÇÃO BRASIL, 2007).

Na literatura, são apresentadas diversas estratégias para aprimorar os métodos tradicionais de ensino de algoritmos, sendo que, dentre elas, Amaral, Medina e Tarouco (2016) propõem o emprego de uma concepção de reorganização das matérias de forma mais intuitiva ao aluno. Oliveira, Rodrigues e Queiroga (2016) utilizaram o *Scratch* para auxílio no aprendizado de lógica da programação; Amaral e colaboradores (2017) utilizaram o *Algo+* como um instrumento motivador para alunos e professores. Porém o projeto aqui apresentado aborda o ensino de algoritmos e lógica de programação baseando-se na teoria construtivista, auxiliada por uma ferramenta desenvolvida para apoiar os processos de ensino e de aprendizagem da linguagem de programação *Python*.

Neste contexto, este trabalho propõe a utilização de uma ferramenta (denominada *P.e.p.y*), visando auxiliar os processos de ensino e de aprendizagem de lógica de programação e algoritmos. Para o desenvolvimento dessa proposta foram seguidas algumas etapas: Apresentação dos conteúdos referentes à construção de algoritmos e lógica de programação; utilização de ferramentas; realização de desafios pelos alunos e o estudo da linguagem de programação *Python*. Esse processo está explicado, detalhadamente, na seção 4 deste artigo.

Os principais objetivos deste trabalho envolvem o desenvolvimento e aplicação da ferramenta *P.e.p.y*, bem como a validação da mesma, por meio de um

estudo de caso, visando identificar como o uso de uma ferramenta gamificada (*P.e.p.y*) pode aprimorar os processos de ensino e de aprendizagem da lógica de programação e do pensamento computacional.

Para validar a proposta foi realizado um estudo de caso com 119 alunos da Escola Estadual de Educação Básica Sepé Tiarajú, localizada na cidade de Frederico Westphalen - RS. Os alunos participantes estão cursando o primeiro, segundo e terceiro anos do Ensino Médio. A proposta teve, como método de avaliação, os desafios realizados pelos alunos, além da aplicação de um questionário. O questionário foi aplicado em dois momentos: No primeiro contato com os alunos e, posteriormente, após a realização dos desafios. As atividades práticas foram desenvolvidas no Laboratório de Informática da referida escola, conduzidas pelos autores desta proposta.

O restante do artigo está organizado como segue: A seção 2 apresenta os trabalhos relacionados. A seção 3 apresenta alguns conceitos da teoria construtivista. A seção 4 detalha a proposta deste projeto. A seção 5 apresenta a ferramenta gamificada *P.e.p.y*. A seção 6 expõe o delineamento do estudo de caso, bem como discute os resultados obtidos. Encerrando o artigo, são apresentadas as considerações finais e as referências empregadas.

2 TRABALHOS RELACIONADOS

Dentre algumas abordagens utilizadas para apoiar a redução das dificuldades inerentes aos processos de ensino e de aprendizagem de algoritmos e lógica de programação, foram estudados ambientes imersivos, paradigma de blocos de programação visual, uso da ferramenta *Scratch* e a ferramenta *Algo+*, para o apoio ao ensino de algoritmos e programação para alunos iniciantes.

Amaral, Medina e Tarouco (2016), propõem uma concepção diferenciada para os processos de ensino e de aprendizagem de algoritmos, caracterizada pela prática pedagógica baseada na integração de ambientes imersivos a paradigmas de programação. Uma avaliação foi conduzida com quatro turmas de acadêmicos das disciplinas de algoritmos e programação da Universidade Federal do Pampa. Os resultados apontam que os alunos obtiveram um percentual de 91% de aprovação. Além disso, houve a criação da capacidade de construção do pensamento computacional dos participantes do experimento. Embora existam resultados satisfatórios, o trabalho não se dedicou à criação de um sistema para auxiliar no aumento desse percentual de aprovação, como é o caso do trabalho desenvolvido e apresentado neste artigo.

Oliveira, Rodrigues e Queiroga (2016) utilizaram o *Scratch*, uma ferramenta intuitiva, tendo sua programação por meio de blocos visuais, o que torna sua compreensão fácil. Os autores utilizaram o *Scratch* em um estudo realizado com alunos ingressantes em cursos técnicos e tecnológicos da área de informática de diferentes instituições. A avaliação apresentada no estudo demonstra um índice de 85% de aprovação na turma de ensino superior obtendo, assim, um aumento na aprovação de 5% em relação à média de aprovações em anos anteriores. O uso do *Scratch*, além de propiciar um incremento nas notas dos alunos, também aumentou o grau de satisfação dos participantes, levando à maior permanência dos alunos em horário extra-aula e à participação destes em eventos. Embora tenha ocorrido um aumento na satisfação dos alunos, não houve uso de uma metodologia diferenciada de ensino.

Amaral e colaboradores (2017) apresentam um estudo que aborda o *Algo+* como um elemento inovador e motivador de professores e alunos. O *Algo+* é uma aplicação *web* que possibilita a livre escolha do

estudante entre dois métodos de ensino: O primeiro com apenas uma linguagem de programação tradicional e o segundo com diferentes paradigmas de programação durante a disciplina. Além disso, vale salientar a possibilidade que o professor tem de acompanhar a evolução de suas turmas. O experimento foi realizado com duas turmas da disciplina de algoritmos e programação. Os resultados apontam que não houve uma grande diferença entre as notas dos alunos da sala de aula tradicional em comparação aos discentes que utilizaram a ferramenta. No entanto, entre os participantes do experimento, houve uma grande aceitação na forma na qual foram organizados os conteúdos. Não houve, como no trabalho destacado anteriormente, a aplicação de uma metodologia de ensino diferenciada.

O trabalho apresentado neste artigo propõe uma abordagem construtivista no ensino de Programação, pois trata o aluno como sujeito ativo, criador da sua própria lógica (sujeito na construção do seu conhecimento), incluindo o uso de ferramentas de ensino de programação. Nenhum dos trabalhos relacionados destaca o uso de uma teoria construtivista em conjunto com ferramentas para apoiar os processos de ensino e de aprendizagem de algoritmos e lógica de programação.

3 TEORIA CONSTRUTIVISTA

Ainda buscando-se alternativas para apoiar os processos de ensino e de aprendizagem de algoritmos e lógica de programação, foi feito um estudo envolvendo Teorias de Aprendizagem. Nesse contexto, Vasconcelos, Praia e Almeida (2003) abordam a teoria cognitivo-construtivista, destacando o aluno no papel central do processo e não o professor. O aluno, nesse sentido, deve atuar como construtor do seu próprio conhecimento, sendo sujeito ativo, desenvolvendo sua criatividade e compreensão. Essa ideia fundamenta-se na teoria construtivista, de Piaget (CARRETERO, 2002; FRANCO, 2004; SILVEIRA *et al.*, 2019). Segundo Vasconcelos, Praia e Almeida (2003), a aprendizagem só ocorre quando a informação nova é ligada a conceitos existentes.

Na abordagem construtivista (construtivismo ou construcionismo), o aluno é visto como construtor do seu conhecimento, mas que está inserido em uma sociedade, em uma determinada cultura que contribuirá na determinação do seu saber (VYGOTSKY, 2007). A construção do conhecimento, que possibilita a aprendizagem, permite que os alunos assimilem novos conhecimentos, a partir de conceitos já conhecidos. Essa construção envolve interação, estudo, experiência e erro. Nesse sentido, os processos de ensino e de aprendizagem não podem envolver meramente atividades repetitivas. O professor precisa estimular os alunos a desenvolverem sua criatividade e interagirem entre eles. As TDICs (Tecnologias Digitais da Informação e da Comunicação) podem ser aliadas nessa questão, pois diferentes ferramentas computacionais podem ser utilizadas neste contexto, tal como a ferramenta apresentada neste artigo (*P.e.p.y*).

> O construtivismo é ...a ideia que sustenta que o indivíduo (...) não é um mero produto do ambiente nem um simples resultado de suas disposições internas, mas sim, uma construção própria que vai se produzindo, dia a dia, como resultado da interação entre esses dois fatores. (...) segundo a posição construtivista, o conhecimento não é uma cópia da realidade, mas sim, uma construção do ser humano (CARRETERO, 2002, p. 10).

Essa afirmação deixa claro que o conhecimento é algo individual, ou seja, mesmo estudando sobre um mesmo conteúdo, cada um de nós construirá o conhecimento sobre este conteúdo de forma diferente, são saberes diferentes.

Segundo Piaget, o criador da teoria construtivista, o conhecimento não está no sujeito nem no objeto, mas ele se constrói na interação do sujeito com o objeto.

Na medida em que o sujeito interage com os objetos é que ele produz a capacidade de conhecer e produz o próprio conhecimento (BRENELLI, 2005; FRANCO, 2004; SILVEIRA *et al.*, 2019). A construção é realizada por meio de esquemas que cada pessoa já possui, ou seja, esquemas que foram construídos por meio da sua relação com o meio em que vive.

> Os esquemas (...) são as estruturas mentais ou cognitivas pelas quais os indivíduos intelectualmente organizam o meio. São estruturas que se modificam com o desenvolvimento mental e que se tornam cada vez mais refinadas (...) os processos responsáveis por essas mudanças são assimilação e acomodação (CAMPOS, 1996, p. 19, citado por SILVEIRA *et al.*, 2019).

Um esquema é um padrão de comportamento ou uma ação que se desenvolve com certa organização e que consiste em um modo de abordar a realidade e conhecê-la.

Segundo Piaget, as chaves principais do desenvolvimento são a própria ação do sujeito e o modelo pelo qual esta ação se converte em um processo de construção interna, isto é, de formação dentro da mente de uma estrutura em contínua expansão, que corresponde ao mundo exterior (SILVEIRA *et al.*, 2019).

> O papel do professor não pode ser nem de um 'expositor' nem de um 'facilitador', mas sim de um problematizador. Isto significa que o professor está ali para organizar as interações do aluno com o meio e problematizar as situações de modo a fazer o aluno, ele próprio, construir o conhecimento sobre o tema que está sendo abordado (FRANCO, 2004, p. 56).

Brenelli (2005) coloca que o professor precisa criar situações-problema que desencadeiem a atividade espontânea do sujeito, para que as estruturas cognitivas se desenvolvam. A atuação como problematizador faz com que o professor tenha um trabalho maior na preparação de suas aulas. Além disso, os alunos precisam estar motivados para atuarem como sujeitos ativos no processo de aprendizagem, não apenas como meros expectadores de aulas expositivas tradicionais.

Na abordagem construtivista, então, o professor deve produzir situações que favoreçam a compreensão dos alunos, de que existe um conflito entre sua ideia sobre um determinado fenômeno e a concepção cientificamente correta. Isso supõe a aplicação de uma metodologia educativa que apresente maiores dificuldades e complicações do que geralmente se quer reconhecer (CARRETERO, 2002). Essa metodologia não é a aula tradicional, expositiva. Isso não significa que não podemos e/ou não devemos utilizar a aula expositiva, mas que devemos utilizá-la cada vez menos.

> A aula expositiva pode ser resultado eminente do esforço de argumentação do professor, no qual mostra sua capacidade de elaboração e comunicação. Este tipo de aula precisa ser defendido e continua em alta. O problema é que predomina, de longe, a aula reprodutiva, fruto, geralmente, de professores mal preparados, desestimulados e cansados (...) (DEMO, 2005, p. 74).

O aluno aprende quando participa ativamente do processo de aprendizagem, isto é, quando constrói, modifica, diversifica e coordena progressivamente seus esquemas de conhecimento estabelecendo, deste modo, redes de significado que enriquecem seu conhecimento do mundo físico e social e potencializam seu crescimento pessoal. Isso significa que o professor deve centrar o processo na atividade construtiva do aluno, nas suas possibilidades de elaboração pessoal, promovendo a autonomia na aprendizagem, a partir da experiência e conhecimento dos alunos, recuperando-os para novas aprendizagens, planejando tarefas que lhes permitam aprender a pensar (observar, analisar, classificar, organizar, hierarquizar, questionar, elaborar hipóteses e comprová-las), ou seja, aprender a aprender. Na atual sociedade do conhecimento, aprender a aprender é um aspecto muito importante, pois todos os dias precisamos aprender novas ferramentas,

novas tecnologias, novos conhecimentos, etc., em um processo contínuo de desenvolvimento cognitivo (SILVEIRA *et al.*,2019).

O trabalho apresentado neste artigo propõe uma abordagem construtivista no ensino de algoritmos e lógica de programação, pois trata o aluno como sujeito ativo, criador da sua própria lógica, incluindo o uso de ferramentas de ensino de programação, deixando de utilizar apenas a sala de aula expositiva e as metodologias tradicionais de ensino.

4 Proposta

A abordagem proposta compreende quatro etapas, sendo elas: 1) Apresentação de conceitos; 2) Lógica de programação; 3) Pensamento computacional e 4) Prática da programação. A Figura 1 ilustra essas etapas.

Figura 1 - Etapas da proposta

Fonte - Próprio autor.

4.1 Primeira Etapa: Apresentação de Conceitos

A primeira etapa envolve uma explicação teórica dos conceitos básicos de lógica de programação aos alunos, focando na compreensão das funcionalidades de diferentes algoritmos. A apresentação é baseada em exemplos do dia-a-dia, relacionando os conceitos estudados, com algum já conhecido pelo aluno, baseando-se na ideia da assimilação da teoria construtivista. Além dessa apresentação, a explicação dos conceitos básicos de programação é realizada junto à resolução de questões no ambiente *P.e.p.y*, pelo professor e alunos. Para a resolução dos exercícios apresenta-se uma questão de cada turma já cadastrada no sistema, sendo: Turma 1: Conceitos de lógica de programação; Turma 2: Conceitos de algoritmos; Turma 3: Operadores aritméticos; Turma 4: Variáveis; Turma 5: Estruturas de condição e, Turma 6: Laços de repetição. As turmas correspondem às diferentes unidades (conteúdos abordados).

4.2 Segunda Etapa: Lógica de Programação

A segunda etapa se caracteriza pela utilização da ferramenta *Scratch*, que é uma ferramenta voltada para apoiar os processos de ensino e de aprendizagem que envolvem os primeiros passos de lógica de programação. O aluno desenvolve a lógica de programação por meio de blocos, proporcionando uma experiência mais prazerosa e simples aos iniciantes em programação (DIAS; SERRÃO, 2014; SCRATCH, 2019). Estudar os primeiros passos de lógica de programação e construir pequenos programas são os objetivos dessa etapa. Espera-se que o aluno compreenda, assim, a estrutura e a lógica de programação, para que, quando for exigido, saiba resolver problemas mais complexos em outra linguagem de programação.

4.3 Terceira Etapa: Pensamento Computacional

A terceira etapa se caracteriza pelo início da apresentação de desafios aos alunos. Os desafios realizados nessa etapa são propostos em uma linguagem simples, de fácil entendimento, conhecida como *Portugol*, o que faz com que a jornada de aprendizagem da programação se torne mais construtiva (RODRIGUES, 2010). Utilizando *Portugol*

é possível escrever algoritmos que utilizam a sintaxe mais próxima das linguagens de programação, mas as instruções são escritas em português (tais como ler, escrever, etc). Porém ao chegar à quarta etapa os desafios devem ser desenvolvidos na linguagem de programação *Python*, sendo realizados no ambiente *P.e.p.y*. O ambiente permite que o professor cadastre questões e os alunos as solucionem programando, em linhas de código, dentro do próprio sistema.

O objetivo dos desafios é o de incentivar os alunos para que consigam construir sua própria lógica de pensamento computacional (PC), que é o conjunto de habilidades utilizado para resolver problemas. Esses desafios são uma forma de incentivar os alunos a utilizarem conceitos assimilados nos exercícios anteriores, resolvendo desafios mais complexos, focando em uma abordagem baseada na compreensão.

4.4 Quarta Etapa: Prática da Programação

A quarta etapa é caracterizada pelo estudo da linguagem de programação *Python*, utilizando a lógica de programação estudada nas etapas anteriores. O objetivo dessa etapa envolve a compreensão da sintaxe da linguagem de programação *Python* e suas funcionalidades. Esse processo, no primeiro momento, envolve a apresentação da linguagem e suas funcionalidades. Embora o entendimento de uma linguagem de programação seja um processo complicado ao aluno, esse procedimento é facilitado pelas etapas anteriores.

Além disso, existe um fator diferenciado, explorado nessa etapa. O *P.e.p.y*, além de focar no aprendizado da linguagem *Python*, que é uma linguagem baseada no paradigma orientado a objetos, é um ambiente gamificado e proporciona aos alunos compreender lógicas usadas em jogos de computador ou de celulares, que são *softwares* que realmente fazem parte do cotidiano. Isso possibilita que os processos de ensino e de aprendizagem sejam mais intuitivos, proporcionando interação do aluno dentro e fora da sala de aula (ROSSUM; DRAKE, 2013).

5 A Ferramenta Gamificada P.e.p.y

A ferramenta desenvolvida busca apoiar os processos de ensino e de aprendizagem de algoritmos e lógica de programação, visando proporcionar um ambiente de fácil compreensão aos usuários. Na área do aluno, em específico, buscou-se criar um *layout* gamificado para tornar o ensino mais lúdico, fácil e motivador, possibilitando a associação de conteúdos (SILVA *et al.*, 2015).

O *P.e.p.y* tem, como interface inicial, uma área de identificação, pois ele possui dois tipos de usuários (professores e alunos). Após a escolha do tipo de usuário, o sistema conduzirá o usuário para a área de *login* em que o mesmo se enquadra, que solicitará nome e senha para verificar a existência do cadastro no sistema e, caso as informações estejam corretas, abrirá a interface correspondente.

Para os conceitos da primeira etapa, foram desenvolvidos um tutorial com vídeos sobre conceitos de programação e o “resolver questão”. As duas opções estão disponíveis na área do aluno, sendo a primeira no menu “ajuda” e a segunda dentro do menu “questões”. Nesse espaço o aluno pode escolher entre os exercícios cadastrados pelo docente em uma turma ou somente um, a partir do seu código de identificação. O discente, após a escolha, programando em linhas de código, dentro da própria ferramenta, resolverá o exercício e o enviará para correção. Ainda existem outras três funções auxiliares dentro do menu “questões”: excluir resposta, ver questões resolvidas e resolver desafios.

Pensando nos conceitos da segunda etapa, criou-se a área do professor possuindo o menu “questões” com várias funcionalidades (Figura 2): Cadastro de questões, alteração de questões, excluir questões, ver suas questões e cadastro de desafios. A função mais importante nessa etapa se caracteriza no “cadastro de questões”, que possibilita o cadastro de exercícios, informando a turma em que o mesmo deve ser aplicado, o valor correto da saída do exercício ou da lógica procurada na resposta, o valor da pontuação que o aluno ganhará caso acerte a resposta e a escolha do estilo da correção, que é fundamental no momento da avaliação da resposta do aluno, que acontece após ele realizar a questão. A opção de cadastro de desafios funciona da mesma forma que a opção de cadastro de questões.

Figura 2 - Área do *P.e.p.y* para Cadastro de Questões

Fonte - Próprio autor.

Implementando os conceitos da terceira etapa, foi desenvolvida a opção “resolver desafios”, na área do aluno, que fica dentro do menu “questões”. Nesse espaço o aluno pode resolver desafios cadastrados pelo professor. O discente, após a escolha do desafio, programando em linhas de código, dentro da própria ferramenta, resolverá o exercício e o enviará para correção. A Figura 3 apresenta a interface da ferramenta *P.e.p.y* utilizada para resolver os desafios.

Figura 3 - Área do *P.e.p.y* para resolver desafios

Fonte - Próprio autor.

A correção dos desafios e exercícios ocorre no próprio ambiente *P.e.p.y*, de forma automática, mas ela é executada conforme a escolha feita pelo professor no momento do cadastro da questão. Se o estilo escolhido pelo docente for “resultado final”, a resposta de saída do que foi interpretado do código do aluno será comparada com o que o professor colocou no campo “valor da comparação”. Se o estilo desejado pelo professor for “lógico”, o ambiente verifica se ocorreu interpretação do código digitado pelo aluno e, se ela ocorrer, irá procurar a lógica inserida no campo "valor da comparação", verificando se a lógica está correta.

A interpretação do código acontece no próprio ambiente *P.e.p.y*, sendo possível pela utilização do *PyPy.js* (PYPYJS.ORG, 2019). O *Pypy.js* permite a utilização do interpretador *PyPy Python* (PYPY.ORG 2019), criando um arquivo .js (*JavaScript*) que pode ser executado na *web*.

Além das opções ligadas à manutenção das questões e desafios, existem outras funções. A opção “fórum” possibilita ao professor visualizar e responder dúvidas dos discentes, o que torna mais fácil a comunicação; a opção “consulta desempenho”, em que o docente pode escolher uma turma específica ou aluno para mostrar dados de exercícios realizados e, caso for de seu interesse, alterar a correção da tarefa. Essas duas opções encontram-se no menu “consulta”.

O menu “serviço”, que é comum em ambas as áreas (aluno e professor) disponibiliza três funções: A “troca de senha”, o “*feedback*”, que é um espaço para o usuário selecionar como foi a sua experiência durante a utilização do ambiente e a função “imprimir”, que possibilita ser impresso qualquer dado de aluno, turma ou questão.

Incrementando os conceitos da quarta etapa, foram desenvolvidos elementos para gamificar o sistema, tais como a pontuação e níveis. A pontuação, além de somar e mostrar os pontos atuais, também mostra quantos pontos ainda faltam para o aluno alcançar o próximo nível. O sistema conta com seis níveis, conforme a pontuação obtida pelo usuário. Cada nível possui um *layout* diferente do outro. Esse conceito foi inspirado em jogos de computador e permite tornar o aprendizado da linguagem *Python*, mais interativo pelo fato de gamificar o ambiente. A Figura 4 apresenta a interface inicial do aluno na ferramenta *P.e.p.y* com diferentes pontuações e níveis.

Figura 4 - Área do *P.e.p.y* em diferentes níveis

Fonte - Próprio autor.

6 Estudo de Caso

Essa seção apresenta o cenário no qual foi realizada a validação do ambiente desenvolvido e da proposta construtivista (por meio de um estudo de caso), além da discussão dos resultados obtidos. Segundo Yin (2001), os estudos de caso são uma metodologia de pesquisa adequada quando se colocam questões do tipo “como” e “por que”. Tais indagações fazem parte do objetivo geral deste trabalho, pois pretende-se

identificar como o uso de uma ferramenta gamificada (*P.e.p.y*) pode aprimorar os processos de ensino e de aprendizagem da lógica de programação e do pensamento computacional.

6.1 Cenário de Validação

A ferramenta desenvolvida (*P.e.p.y*), seguindo a abordagem construtivista, conforme etapas detalhadas nas seções anteriores, foi aplicada com alunos do primeiro, segundo e terceiro ano da Escola Estadual de Educação Básica Sepé Tiarajú, na cidade de Frederico Westphalen - RS. Ao todo, 119 alunos participaram do estudo. A proposta teve, como método de avaliação, os desafios realizados na quarta etapa e um questionário objetivo aplicado no primeiro contato com os alunos e, posteriormente, ao final da quarta etapa.

Os encontros foram realizados na própria escola, no período noturno. Foram realizados 6 encontros, entre maio e outubro de 2018. Os encontros iniciais tiveram uma apresentação do projeto aos alunos, questionando-os sobre quem teria interesse em participar. Nos encontros seguintes aconteceu uma apresentação dos conceitos de algoritmos e lógica de programação aos discentes. Utilizaram-se as ferramentas *Scratch* e *VisuAlg* para estimular os discentes a praticar e resolver tarefas. O *VisuAlg* foi utilizado por ser uma ferramenta para programação em que o código é escrito em *Portugol* (VISUALG3, 2019). Além do uso do *VisuAlg*, em um dos encontros houve a realização de exercícios de programação na linguagem *Python*. As tarefas foram cadastradas na ferramenta *P.e.p.y* e os alunos resolviam conforme conseguiam, sem ajuda do professor. Após os discentes concluírem as tarefas, estas foram enviadas para correção automática no ambiente desenvolvido.

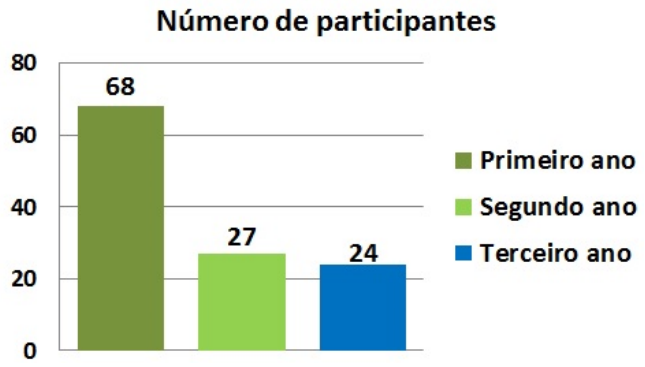
Analisando os resultados obtidos, verifica-se que existem indícios que demonstram a efetividade da proposta defendida, considerando-se o estudo de caso realizado. As comparações foram realizadas com base nos dados coletados com a aplicação do primeiro questionário (aplicado no primeiro contato com os alunos) e os dados coletados por meio do questionário aplicado ao final da quarta etapa, bem como o número de acertos dos desafios.

6.2 Resultados Obtidos

Com base nas perguntas do primeiro questionário, aplicado no primeiro contato com os alunos, foram identificados o gênero dos participantes, o ano de ensino, se o discente teria interesse em seguir seus estudos na área de tecnologia, o período em que os participantes ficavam conectados à Internet, se os alunos tinham alguma noção de linguagem de programação e, por último, foi apresentado um algoritmo escrito em *Portugol*, sendo solicitado o entendimento sobre o seu funcionamento. Inicialmente, para a apresentação dos resultados obtidos, os alunos foram divididos em 3 grupos, de acordo com o ano/série (primeiro, segundo e terceiro ano).

O estudo de caso contou com 119 discentes participantes com idades entre 15 e 22 anos. 68 estudantes cursavam 1º ano do Ensino Médio, 27 frequentavam o 2º ano e 24 estavam no 3º ano, como mostra o gráfico da Figura 5. Um detalhe importante a ser destacado, foi a participação da maioria dos estudantes do Ensino Médio da escola no referido estudo de caso. Solicitou-se à Escola uma lista com todos os alunos matriculados no 1º ano do Ensino Médio (102 alunos), 2º ano do Ensino Médio (50) e 3º ano (48), totalizando 200 discentes. Sendo assim, os 119 participantes do projeto representam 59,5% do total de estudantes do Ensino Médio da Escola.

Figura 5 - Participantes por Ano/Série



Fonte - Próprio autor.

Analisando os dados coletados no primeiro questionário, referentes ao primeiro grupo (1º ano do Ensino Médio), verificou-se que metade do grupo era do gênero masculino e a outra metade do gênero feminino. Porém no questionário aplicado ao final da quarta etapa, houve uma diminuição de 21,4% dos participantes do gênero feminino. Com relação ao interesse em seguir os estudos (graduação) na área de tecnologia, no primeiro questionário 35,5% respondeu “Sim”, 50% “Talvez” e 14,5% “Não”. No entanto, no questionário aplicado na quarta etapa, o percentual de respostas “Sim” foi 57,1% e 42,9% de respostas “Talvez”. Com relação à noção de uma linguagem de programação, no primeiro questionário o grupo tinha apenas 3,45% que responderam “Sim”, 55,9% responderam “Não” e 40,7% responderam “Pouco”. Vê-se que um percentual muito pequeno tinha conhecimento sobre linguagens de programação. Analisando-se os resultados sobre o entendimento de um exemplo de algoritmo, no primeiro questionário apenas 6,7% dos participantes responderam “Ótimo”, seguidos de 50% que responderam “Bom”. Na aplicação do segundo questionário, as respostas “Ótimo” alcançaram 28,6% e “Bom”, 57,1%.

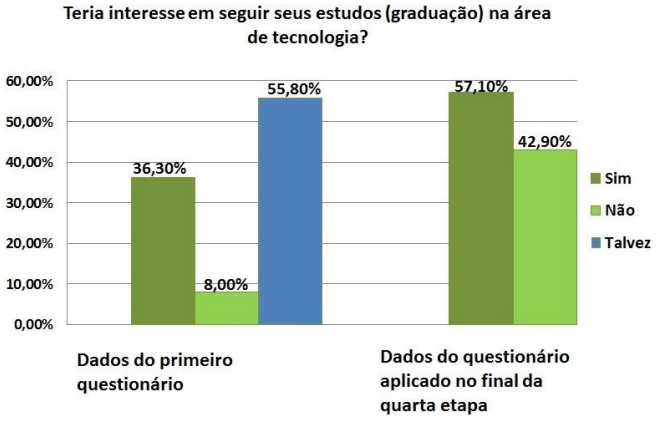
Com relação aos dados coletados do segundo grupo (2º ano do Ensino Médio), no início do estudo de caso os participantes totalizavam 66,7% do gênero feminino e 33,7% do gênero masculino. Na quarta etapa o percentual era de 50% para cada gênero. Com relação a continuarem seus estudos na área de tecnologia, os percentuais foram de 50% para “Sim” e 50% para “Talvez”, nas duas aplicações do questionário. A noção de uma linguagem de programação para o 2º grupo também era muito pequena, pois apenas 4,2% dos participantes responderam “Sim”. Analisando os resultados da pergunta relativa ao entendimento de um algoritmo (questão 7 do instrumento) na primeira aplicação, 4,2% dos participantes responderam “Ótimo”, percentual que aumentou para 25% na aplicação do segundo questionário.

O terceiro grupo (alunos do 3º ano) eram 45,8% do gênero feminino e 54,2% do gênero masculino no início do estudo de caso. Na quarta etapa, restaram apenas participantes do gênero masculino. Com relação a continuar os estudos na área de tecnologia, na primeira aplicação do questionário as respostas ficaram divididas em “Sim” (50%) e “Talvez” (50%). Na quarta etapa, o percentual de respostas “Sim” aumentou para 66,7%. Com relação às noções de programação, inicialmente o grupo apontou “Pouco” (54,2%) e “Não” (45,8%). Na quarta etapa o resultado foi positivo, pois 66% responderam “Sim”. Com base no exemplo de algoritmo demonstrado no questionário, na primeira aplicação, a maioria respondeu “Péssimo” e “Regular”. Na quarta etapa os resultados foram positivos, sendo 33,3% de respostas “Ótimo” e 66,7% de respostas “Bom”.

A partir da tabulação e análise dos dados coletados, foi possível concluir que o estudo de caso estimulou os alunos a seguirem uma graduação na área de Tecnologia da Informação (Figura 6). No primeiro questionário, a opção “Talvez” era com o maior percentual de 55,8%, depois o “Sim” com 36,3 % e “Não” com 8%. Os resultados do questionário aplicado, ao final da quarta etapa, foram muito significativos visto que a opção “Sim” teve 57,1% e, logo a seguir, a opção “Talvez” com 42,9%. Cabe

destacar que, nesta avaliação, nenhum aluno marcou a opção “não”.

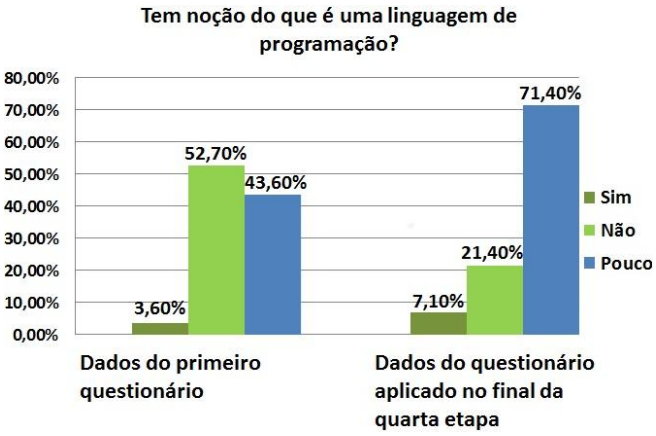
Figura 6 - Porcentagens de interesse em seguir seus estudos na área de tecnologia



Fonte - Próprio autor.

Além disso, o instrumento aplicado também demonstrou que os alunos compreenderam noções de linguagens de programação (Figura 7). O aluno poderia escolher entre três opções: Sim, Pouco e Não. A porcentagem da opção “Não” reduziu, de 52,7% no primeiro questionário para 21,4% no questionário aplicado ao final da quarta etapa. Além disso, houve crescimento da opção “Sim” passando de 3,6% para 7,1% representando quase o dobro (Figura 7).

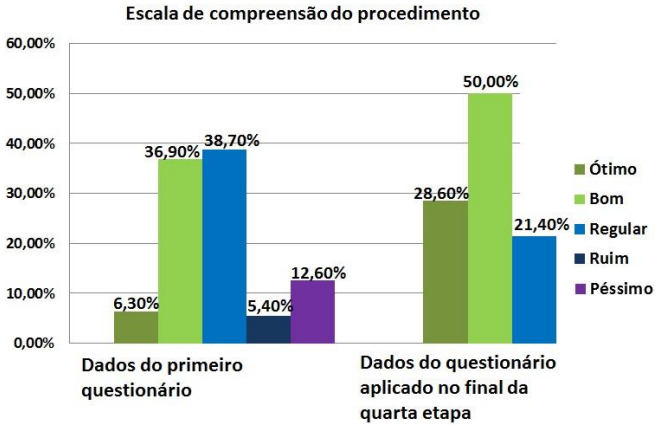
Figura 7 -. Porcentagens referentes à noção de uma linguagem de programação



Fonte - Próprio autor.

Também foi possível verificar o aumento no processo de compreensão dos alunos já que, no primeiro questionário, a opção “Regular” era com o maior percentual de 38,7%, depois o “Bom” com 36,9 %, “Péssimo” com 12.6%, “Ótimo” com 6.3% e “Ruim” com 5.4%. Os resultados do questionário aplicado ao final da quarta etapa, foram otimistas, pois a opção “Bom” obteve 50,0%, a opção “Ótimo” 28,6% (quase seis vezes o valor da primeira avaliação) e, com 21,4%, ficou a opção “Regular”. As opções ruim e péssimo não foram assinaladas por nenhum dos alunos (Figura 8).

Figura 8 - Porcentagens de compreensão do aluno sobre o algoritmo na linguagem de programação *Portugol*

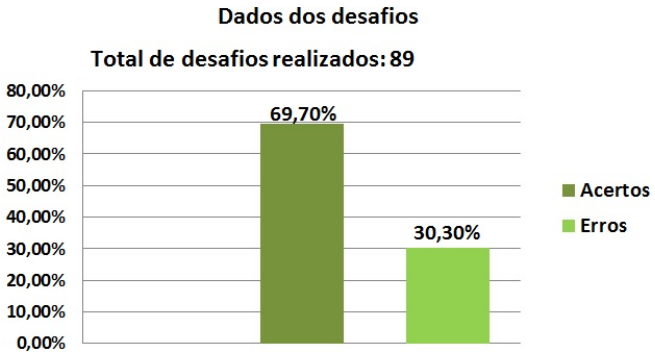


Fonte - Próprio autor.

Outra contribuição importante deste estudo de caso foram os desafios realizados em grupos (aproximadamente quatro alunos por grupo) na ferramenta *P.e.p.y*. Foram cinco desafios cadastrados no sistema. Cada grupo realizou, inicialmente, dois desafios escolhidos pelo docente (um com pontuação 5 e outro com pontuação 7) e, ao final, um desafio bônus (com pontuação 10). Os alunos obtiveram uma porcentagem de acerto de 69,7% (Figura 9). Um detalhe importante sobre os desafios, é que estes foram realizados sobre vários assuntos, abordando conceitos de operadores aritméticos, variáveis,

estruturas de condição, laços de repetição. Além disso, cerca de 89 tarefas foram realizadas sem auxílio de ferramentas (fora do ambiente *P.e.p.y*).

Figura 9 - Porcentagem de acertos e erros dos desafios



Fonte - Próprio autor.

7 CONSIDERAÇÕES FINAIS

O aprendizado de algoritmos e de lógica e programação são de extrema importância para qualquer curso de computação. Acredita-se que a utilização de uma metodologia para apoiar os processos de ensino e de aprendizagem de algoritmos e lógica de programação ajudará na motivação do aluno a seguir na área de computação, como também, na redução das dificuldades apresentadas pelos alunos. Além disso, também contribuirá na diminuição da evasão dos cursos de informática, pois os temas estão diretamente relacionados.

Além disso, a SBC vem destacando a importância do estudo de programação desde o ensino fundamental, tendo elaborado propostas para a inclusão de estudos em Ciência da Computação na proposta da nova BNCC (Base Nacional Curricular Comum). Nesse sentido, a ferramenta *P.e.p.y* contribui, de forma significativa, para apoiar o estudo de programação na Educação Básica (SBC, 2017; SBC, 2018).

Outros diferenciais da proposta apresentada envolvem a abordagem construtivista, para que os alunos se tornem sujeitos ativos na construção do conhecimento e no uso de gamificação. Os resultados obtidos por meio do estudo de caso demonstram que houve ganhos efetivos nos processos de ensino e de aprendizagem, bem como no incentivo para que os alunos continuem seus estudos na área de Tecnologia da Informação.

Com relação aos impactos sociais do projeto desenvolvido, acredita-se que atuar em prol da formação dos discentes da Educação Básica impactará, positivamente, no aprimoramento da capacidade de raciocínio lógico e da aprendizagem destes alunos, bem como ampliará a importância de se desenvolver o pensamento computacional desde o ensino fundamental, algo que vem sendo debatido e amplamente difundido pela SBC (SBC, 2017; SBC, 2018).

Por limitações de tempo, neste estudo de caso só trabalhamos com o ensino da linguagem de programação *Python*, não sendo explorada a possibilidade de ensinar outras linguagens de programação, tais como *Java*, PHP ou C. Essas linguagens serão inseridas na ferramenta, futuramente. Em relação a melhorias na ferramenta desenvolvida, uma sugestão possível seria implantar mais níveis e apresentar ao professor qual foi o erro no código do aluno ao realizar a questão.

Para trabalhos futuros, seria interessante, também, o desenvolvimento de um aplicativo *mobile* para que os alunos tenham mais facilidade no uso da ferramenta. Outra possibilidade é realizar um estudo de caso da aplicação da ferramenta *P.e.p.y* com alunos de cursos superiores de Informática.

REFERÊNCIAS

AMARAL, É.; MEDINA, R. D.; TAROUCO, L. M. R. T. Processo de Ensino e Aprendizagem de Algoritmos Integrando Ambientes Imersivos e o Paradigma de Blocos de Programação Visual. **V Congresso Brasileiro de Informática na Educação.** Anais, 2016. Disponível em: <http://www.br-ie.org/pub/index.php/wcbie/article/view/6905>. Acesso em setembro, 2018.

AMARAL, É. *et al.* ALGO+ Uma ferramenta para o apoio ao ensino de Algoritmos e Programação para alunos iniciantes. **SBIE – Simpósio Brasileiro de Informática na Educação,** Anais, 2017. Disponível em: <http://www.br-ie.org/pub/index.php/sbie/article/view/7699>. Acesso em agosto, 2018.

BRENELLI, R. P. **O Jogo como Espaço para Pensar:** a construção de noções lógicas e aritméticas. Campinas, São Paulo: Papirus, 2005.

CABRAL, M. I. C. *et al.* Perfil dos Cursos de Computação e Informática no Brasil. In: **XXVII Congresso da SBC - XV WEI**, Rio de Janeiro, 2017.

CARRETERO, M. **Construtivismo e Educação.** Porto Alegre: Artes Médicas, 2002.

COMPUTAÇÃO BRASIL **Mercado de Trabalho em Computação:** Oportunidades e Desafios. In: Computação Brasil, Porto Alegre: Sociedade Brasileira de Computação. Edição n. 25; Março/Abril/Maio - Ano VIII, 2007.

DEMO, P. **Universidade, Aprendizagem e Avaliação:** horizontes reconstrutivos. 2. ed. Porto Alegre: Mediação, 2005.

DIAS, K. L.; SERRÃO, M. L. A linguagem Scratch no ensino de Programação: Um relato de experiência com alunos iniciantes do curso de licenciatura em computação. **CSBC – Congresso da Sociedade Brasileira de Computação - XXII Workshop sobre Educação em Computação** – Anais, 2014. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wei/2014/0017.pdf>. Acesso em agosto, 2018.

EL PAIS, E. **Emprego:** Onze novas profissões que darão muito que falar, 2016. Disponível em: <https://brasil.elpais.com/brasil/2016/10/26/tecnologia/1477502097_89975 1.html>. Acesso em: outubro, 2018.

FRANCO, S. R. K. **O Construtivismo e a Educação**. Porto Alegre: Mediação, 2004.

GUSMÃO. C. Jportugol: uma ferramenta de auxílio à aprendizagem de algoritmos. **Brazilian Educational Technology: Research and Learning**, 2, 2011. Disponível em: <http://inseer.ibict.br/betrl/index.php/betrl/article/view/89>. Acesso em setembro, 2018.

HOED, R. M. **Análise da evasão em cursos superiores:** o caso da evasão em cursos superiores da área de computação. Brasília: Programa de Pós - Graduação em Computação Aplicada - UnB. (Dissertação de Mestrado), 2017. Disponível em: <http://repositorio.unb.br/handle/10482/22575>. Acesso em: outubro, 2018.

OLIVEIRA, M.; RODRIGUES, L.; QUEIROGA, A. Material didático lúdico: uso da ferramenta Scratch para auxílio no aprendizado de lógica da Programação. **V Congresso Brasileiro de Informática na Educação – WIE Workshop de Informática na Escola** – Anais, 2016. Disponível em: <http://www.br-ie.org/pub/index.php/wie/article/view/6842>. Acesso em agosto, 2018.

PYPY.ORG **Bem vindo ao PyPy**, 2019. Disponível em: <https://pypy.org/>. Acesso em: março, 2019.

PYPYJS.ORG **PyPy.js**, 2019. Disponível em: <https://pypyjs.org/>. Acesso em: março, 2019.

RODRIGUES, A. **Manual do VisuAlg** Instituto Federal de Educação, Ciência e Tecnologia, Ceará - Campus Iguatu, 2010. Disponível em: <http://www.inf.ufsc.br/~bosco.sobral/ensino/ine5201/Visualg2_manual.pdf>. Acesso em setembro, 2018.

ROSSUM, V. G., DRAKE, F. L. **The Python Language Reference**, 2013. Disponível em: <http://docs.python.org/2/reference/>. Acesso em julho, 2018.

SBC. Sociedade Brasileira de Computação. **Diretrizes para o Ensino de Computação Básica**. Documento Interno da Comissão de Educação Básica da SBC, 2018.

SBC. Sociedade Brasileira de Computação **Referenciais de Formação em Computação:** Educação Básica, 2017. Disponível em: <http://www.sbc.org.br/documentos-da-sbc/send/131-curriculos-de-referencia/1166-referenciais-de-formacao-em-computacao-educacao-basica-julho-2017>. Acesso em maio, 2019.

SCAICO, A. A.. Ensino de Programação no Ensino Médio: Uma abordagem Orientada ao Design com a Linguagem Scratch. **Revista Brasileira de Informática na Educação**, 21(2), 92, 2013. Disponível em: <http://www.br-ie.org/pub/index.php/wie/article/viewFile/2112/1878>. Acesso em setembro, 2018.

SCRATCH BRASIL.NET.BR. **Você conhece o Scratch?**, 2019. Disponível em: <http://www.scratchbrasil.net.br/index.php/sobre-o-scratch.html>. Acesso em junho, 2019.

SILVA P. A. *et al.*. **Gamificação para melhoria do engajamento no ensino médio integrado**, 2015. Disponível em: <http://www.sbgames.org/sbgames2015/anaispdf/cultura-full/146993.pdf>. Acesso em: 09 setembro 2018.

SILVEIRA, S. R. *et al.*. **Metodologia do Ensino e Aprendizagem em Informática.** Santa Maria: NTE/UAB/UFSM, 2019.

VASCONCELOS, C.; PRAIA, J. F.; ALMEIDA, L. S. **Theory of learning and the teaching-learning of sciences-from instruction to apprenticeship**, 2003. Disponível em: <http://pepsic.bvsalud.org/scielo.php?script=sci_arttext&pid=S1413-85572003000100002>. Acesso em julho, 2018.

VISUALG3.COM.BR **O que seria o VisualG?**, 2019. Disponível em: <"http://visualg3.com.br/>. Acesso em março, 2019.

VYGOTSKY, L. **A Formação Social da Mente.** São Paulo: Martins Fontes, 2007.

YIN, R. K. **Estudo de Caso:** planejamento e métodos. 2. ed. Porto Alegre: Bookman, 2001.