UNIVERSIDADE DE PASSO FUNDO

Carmen Vera Scorsatto Brezolin

CONTRIBUIÇÕES DA FERRAMENTA GRÁFICA *BLOCKLY* NO PROCESSO ENSINO-APRENDIZAGEM NA DISCIPLINA DE ALGORITMOS

Passo Fundo

2016

Carmen Vera Scorsatto Brezolin

CONTRIBUIÇÕES DA FERRAMENTA GRÁFICA *BLOCKLY* NO PROCESSO ENSINO-APRENDIZAGEM NA DISCIPLINA DE ALGORITMOS

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Educação da Universidade de Passo Fundo, como requisito parcial para obtenção do título de Mestre em Educação, sob orientação da Profa. Dra. Neiva Ignês Grando.

Passo Fundo

2016

Dedico este trabalho aos meus pais, com quem sempre posso contar em todos os momentos da vida, a meu esposo João e às minhas filhas Helena e Gabriela que estiveram sempre ao meu lado durante essa caminhada.

Agradeço

À minha orientadora Profa. Dra. Neiva Ignês Grando, por sua dedicação e disponibilidade, pois seus ensinamentos foram fundamentais e me conduziram durante o desenvolvimento da pesquisa.
Aos alunos pela disponibilidade para contribuir e ao Ifsul que concedeu os recursos financeiros para viabilizar a pesquisa.

RESUMO

Algoritmos é uma disciplina na qual são abordados os conceitos fundamentais de programação de computadores, considerados essenciais para os alunos que desejam seguir um curso de Tecnologia da Informação. Nessa disciplina, exige-se do aluno uma elevada capacidade de raciocínio lógico e de abstração, demandando do professor um comprometimento com o processo de aprendizagem e disponibilidade para elaborar propostas pedagógicas que auxiliem os alunos a superar as dificuldades que possam surgir. Nesse contexto, esta pesquisa propôs-se a investigar as potencialidades do uso de uma ferramenta gráfica (Blockly) como estratégia de ensino e de aprendizagem na disciplina de Algoritmos. Participaram do estudo 26 alunos e a professora (a própria pesquisadora) da disciplina de Algoritmos, do Curso Superior em Sistemas para a Internet do Instituto Federal de Educação, Ciência e Tecnologia Sul Rio-Grandense/Ifsul, do campus de Passo Fundo/RS, no segundo semestre de 2014. A pesquisa caracterizou-se por uma abordagem qualitativa, sendo que seus dados foram constituídos de registros das atividades realizadas pelos alunos com o uso da ferramenta gráfica, também, das atividades realizadas em sala de aula e das observações da pesquisadora-professora. Observou-se, por meio da aplicação da ferramenta, que, durante esse processo, o papel do software utilizado foi mudando paulatinamente e, além de ferramenta para aplicação e ampliação dos conhecimentos abordados na disciplina, tornou-se um instrumento de diagnóstico. Tal instrumento permitiu identificar dificuldades conceituais da construção de algoritmos, os quais, até então, a professora-pesquisadora não havia observado. Assim, a investigação possibilitou a compreensão da ferramenta como um instrumento mediador utilizado para estimular os alunos a elaborar os planos de ações e a resolver os desafios, ao mesmo tempo em que deu pistas à professora sobre como atuar na Zona de Desenvolvimento Proximal daqueles sujeitos. Além da Teoria Histórico-Cultural, fundamentaram o estudo noções da Didática da Matemática, a Teoria das Situações Didáticas (Brousseau), a teoria dos registros de representações semióticas (Duval) e conceitos tais como transposição didática (Chevallard), resolução de problemas (Polya) e representações mentais (Duval). Com base nos resultados da pesquisa, apresentam-se ponderações importantes que ultrapassam a aplicação de um software, baseadas na observação do "comportamento", das "atitudes" e dos procedimentos adotados pelos alunos e pela professora no decorrer do processo investigativo. Durante o desenvolvimento da pesquisa, foram identificados dificuldades da própria professora-pesquisadora em relação a sua prática e a sua habilidade de sentir-se parte do processo ensino-aprendizagem de Algoritmos. Constatou-se que o uso de um software com o intuito de auxiliar no processo de ensino-aprendizagem de algoritmos constitui os "saberes" necessários para ensinar.

Palavras-chave: Ensino de Matemática. Algoritmos. Ferramenta gráfica. *Blockly*. Ensino Superior.

ABSTRACT

Algorithms is a discipline in which fundamental concepts of computer programming are addressed, they are considered essential for students who wish to follow the Course of Information Technology. In this discipline, high capacity for logical reasoning and abstraction is required from students, demanding from the teacher a commitment to the learning process and a certain availability to develop educational proposals that help students overcome difficulties that may arise. In this context, this research proposes to investigate the potential of the use of a graphical tool (Blockly) as a teaching and learning strategy in the algorithms subject. 26 academics participated in this study and also the teacher (the researcher) of Algorithms discipline of the Internet Systems graduation course from Federal Institute of Education, Science, and Technology of Rio Grande do Sul/IFSul, Passo Fundo in the second half of 2014. The research had a qualitative approach and its data were consisted of records from the activities performed by the students using the graphical tool, activities in the classroom and the observations performed by the researcher-teacher. It was observed, through the application of the tool, that during this process, the role of the software used has been changed gradually from a tool for application and expansion of knowledge to a diagnostic tool. This instrument allowed to identified conceptual difficulties on constructing algorithms of contents that were already covered in the course, and, until then, were not noticed by the teacher. Thus, the research enabled the understanding of the tool as a mediator instrument used to encourage students to design action plans and solve challenges at the same time as it gave clues to the teacher about how to act on Proximal Development Zone of those individuals. Besides the historical-cultural theory, Didactic notions of mathematics, the Theory of Didactic Situations (Brousseau), the theory of records of semiotic representations (Duval) and concepts such as didactic transposition (Chevallard), problem solving (Polya) and mental representations (Duval) substantiated the study. Based on the survey results, important considerations that go beyond the application of a software are presented, based on observing the "behavior", the "attitudes" and the procedures adopted by the students and the teacher during the investigative process. During the development of this research, important elements emerged that made possible to identify difficulties of the researcher-teacher regarding its practice and its ability to feel part of the teaching-learning algorithms process. It was noted that the use of software with the purpose of assist in the teaching-learning of algorithms process constitute the "knowledge" needed to teach.

Key-words: Mathematics Teaching. Algorithms. Graphical Tool Blockly. Higher Education.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AL: Algoritmos

EAD: Educação a Distância

IFSUL: Instituto Federal de Educação, Ciência e Tecnologia Sul Rio-grandense

LP: Linguagem de programação

MIT: *Massachussets Institute of Technology*

TSPI: Tecnologia em Sistemas para Internet

UTFPR: Universidade Tecnológica Federal do Paraná

XML: *Extensible Markup Language*

ZDP: Zona de Desenvolvimento Proximal

SUMÁRIO

1 INTRODUÇÃO

Depois de oito anos trabalhando como programadora de computador iniciei minha carreira como professora, com a ingênua aspiração de que faria com que meus alunos gostassem da disciplina de Algoritmos se ela fosse apresentada da forma como aprendi a gostar na universidade e, também, de que, por meio da experiência no mercado de trabalho, teria subsídios suficientes para convencê-los do quão importante é aprender essa disciplina. Desde 2002, atuando como professora em cursos de Informática, foram anos de busca, de "altos e baixos", com o objetivo de entender o porquê de os alunos apresentarem tantas dificuldades com Algoritmos.

Desde o ano de 2008, quando ingressei no Instituto Federal Sul-Rio-Grandense (Ifsul) do campus Passo Fundo, abdiquei de trabalhar como programadora e dediquei-me, exclusivamente, à carreira de professora. Com isso começaram a surgir muitas inquietações acerca dessa nova fase da vida, as quais me trouxeram ao curso de Mestrado em Educação, no qual, por meio desta pesquisa, trago questionamentos sobre o ensino e a aprendizagem da principal disciplina que ministro, a Algoritmos (AL).

O Ifsul campus Passo Fundo é um dos quatorze campi vinculados à sede de Pelotas, o qual iniciou suas atividades em outubro de 2008. Atualmente, esse campus oferta os cursos de nível pós-médio, de Técnico em Edificações, Técnico em Informática e Técnico em Mecânica e de nível superior, além dos cursos de Tecnologia em Sistemas para a Internet, Engenharia Mecânica e Engenharia Civil.

A disciplina de AL é um dos pilares do Curso Superior de Tecnologia em Sistemas para a Internet (TSPI). Nessa disciplina, o aluno deverá estudar os conceitos necessários para desenvolver as bases do raciocínio lógico, voltado ao contexto das linguagens de programação, o que é fundamental para seu êxito tanto na disciplina, quanto no curso como um todo. É nessa disciplina que os alunos do curso (TSPI) são introduzidos aos princípios da lógica e da programação de computadores, áreas que estabelecem o embasamento dos principais conceitos sob os quais a Ciência da Computação está fundada.

O objetivo geral da disciplina de Algoritmos, segundo seu Plano de Ensino, é "Desenvolver o raciocínio lógico, estrutura da lógica formal e conceitos de Linguagem de Programação". Além de estimular o raciocínio lógico dos alunos, a disciplina se propõe a capacitá-los a realizar a transposição dos problemas do mundo real para o ambiente computacional. Para realizar a tradução desses problemas para a uma linguagem que possa ser compreendida e processada por computadores, utilizam-se Algoritmos.

Embora os alunos já tenham noções matemáticas oriundas da educação básica, encontram muitas dificuldades para a sua representação no ambiente computacional. Nesse sentido, observamos que, historicamente, nessa disciplina acontece um alto índice de evasão e reprovação. Entre 2010 e 2014, foram ao todo quatorze turmas de Algoritmos, nos turnos da manhã e noite, a média de reprovação nessas foi de 44,03%. Um número preocupante, que despertou a consciência de que é preciso pesquisar outras formas, encontrar maneiras eficazes de estimular a aprendizagem dos alunos e, principalmente, de entender as dificuldades desse processo.

Setti (2009) menciona como um dado preocupante obtido com seu estudo de caso, o fato de que mais de 60% dos alunos não conseguiram propor uma solução algorítmica para determinado problema e aproximadamente 30% também não conseguiram elaborar uma solução matemática. Isso mostra, segundo o autor, o despreparo desses alunos em relação à resolução de problemas, embora tenham passado por um rigoroso processo de seleção, em uma relação de aproximadamente dez candidatos por vaga.

Do mesmo modo, Miranda (2004) descreve que Algoritmo, embora seja uma disciplina inicial, proporciona um grande grau de dificuldade para os alunos, porque esses não conseguem se adaptar à forma do pensamento do "passo a passo". O autor ainda menciona a dificuldade que o professor encontra durante as aulas de avaliar qual é a real dificuldade apresentada pelos alunos, apontando que, normalmente, alguns aprendizes não expõem de forma verbal os problemas encontrados, os quais somente se tornam claros durante a aplicação de uma prova, ou mesmo de exercícios válidos como nota.

Convencionalmente, o aprendizado de algoritmos é desenvolvido a partir da transcrição dos problemas propostos para o pseudocódigo, também denominado, por Forbellone e Eberspächer (2005), de português estruturado e ainda por Ascencio e Campos (2012) de Portugol. Tais problemas podem ser escritos diretamente em uma linguagem de programação estruturada (como C++, por exemplo), o que impõe um desafio a mais para o aluno, o de compreender a estrutura e a sintaxe da linguagem utilizada.

Um dos grandes obstáculos encontrados ao ministrar a disciplina de Algoritmos é identificar qual é a real dificuldade dos alunos, pois os próprios discentes não conseguem identificá-las, dizendo em geral apenas: "Não entendi nada". Nessa situação, torna-se complicado ajudá-los. Essa problemática implica reflexões acerca das estratégias de ensino até então utilizadas na disciplina.

A inquietação acerca das dificuldades apresentadas pelos alunos e os seus altos índices de reprovação e evasão, obrigaram-me a repensar a prática adotada até então.

Acredito que profissionais não podem permanecer omissos a responsabilidade de repensar sobre a eficácia dos métodos, do agir dentro da sala de aula e dos resultados de cada semestre. Trazendo Gauthier et al. para refletir: "Não se pode responsabilizar um advogado por ter perdido uma causa, nem um médico por não ter conseguido manter seu paciente com vida se eles deram provas de terem usado todos os meios necessários para "vencer"" (2013, p. 124).

Na turma anterior ao início desta pesquisa (1M1/2014-1), o percentual de alunos aprovados foi de 36,4%, ou seja, somente oito passaram em uma turma de 22 alunos, cinco desistiram ainda nas primeiras avaliações e nove reprovaram por não atingir a média necessária (6,0). A soma dos percentuais de reprovação e evasão ultrapassam 60%, índices que deixam clara a necessidade de estudo e de comprometimento, a qual é, também, citada por Gauthier et al., para os quais "A responsabilidade ética para com os alunos provoca no professor a necessidade de conhecer os melhores meios para instruir e educar, ou seja, ele não pode ignorar o que a pesquisa diz a respeito das melhores práticas" (2013, p. 399).

Em busca de respostas para as inquietações advindas da prática docente, iniciei a busca por ferramentas gráficas que contribuíssem para os processos de ensino e de aprendizagem de algoritmos. Observando pesquisas que abordavam o tema de aprendizagem de algoritmos e uso de ferramentas gráficas, verifiquei que, na maioria, houve certa tendência em direcionar o foco para aspectos relacionados ao impacto que essas causariam com relação aos interesses e frustrações dos alunos. Poucas investigações se preocuparam com o processo de seu uso, com o comportamento dos envolvidos, ou, ainda, abordaram e analisaram o papel do professor nesse processo.

Com base nas razões explicitadas, ou seja, nas dificuldades do processo de ensino-aprendizagem na disciplina de Algoritmos e na falta de pesquisas que pudessem orientar esse processo, defini, juntamente com a orientadora, o tema de pesquisa: A relação entre o estudo de Algoritmos e o uso de ferramentas gráficas.

Perpassando o tema, definimos como pergunta principal: Em que medida o uso de uma ferramenta gráfica de programação pode qualificar os processos de ensino e de aprendizagem na disciplina de Algoritmos?

Por meio desse questionamento, a presente pesquisa objetiva investigar se a ferramenta gráfica *Blockly* contribui para o ensino-aprendizagem de algoritmos, no Curso Superior de Tecnologia em Sistemas para a Internet (TSPI).

O texto desta dissertação está organizado em quatro capítulos, além da Introdução e das Considerações finais. No primeiro, contextualizamos a pesquisa pela definição do local

e caracterização dos sujeitos participantes, sendo, em seguida, apresentados os aspectos metodológicos gerais da pesquisa com a finalidade de justificar seu caráter qualitativo.

O segundo capítulo compreende os fundamentos para a pesquisa, inicialmente, expomos as definições, as formas de representação e os conceitos básicos para a construção de algoritmos. Apresentamos também as ferramentas gráficas analisadas para o ensino de algoritmos e a ferramenta *Blockly* selecionada e utilizada nesta pesquisa. Ainda nesse capítulo, apresentamos a revisão bibliográfica e as teorias que embasam a investigação. Primeiramente, reflexões sobre algumas noções da didática da matemática, assim como da teoria das situações didáticas e alguns de seus principais conceitos. Abordamos a resolução de problemas por meio de uma abordagem cognitiva, elencamos alguns autores que articulam a temática do desenvolvimento geral da capacidade de raciocínio dos alunos e o desenvolvimento das suas representações mentais como a teoria dos registros e representações e semióticas de Duval. Buscamos contribuições da psicologia sobre o desenvolvimento do pensamento por meio do conceito das funções psicológicas superiores ou dos processos mentais superiores de Vygotsky. Para finalizar o segundo capítulo, apontamos resultados de pesquisas que abordaram temas de aprendizagem de algoritmos e uso de ferramentas gráficas.

No terceiro capítulo, são discutidos aspectos relacionados à disciplina de Algoritmos referentes à metodologia utilizada nas aulas até então e as dificuldades descritas pelos alunos com essa. O último capítulo compreende a descrição do processo de investigação realizado com a aplicação da ferramenta na turma escolhida na disciplina de Algoritmos. Inicialmente, apresentamos os padrões das atividades utilizadas na ferramenta, bem como os conceitos que essas envolvem e o processo de utilização do *Blockly* atividades na disciplina. Ainda nesse capítulo, expomos a descrição e a análise do material gerado pela pesquisa, primeiramente com análise do desempenho geral da turma, com conseguinte análise do processo de utilização da ferramenta pelos alunos de forma individualizada.

Por fim, são exibidas as considerações finais da pesquisa. Por meio da aplicação da ferramenta surgiram reflexões que ultrapassam a aplicação de um *software*, pela observação do "comportamento", das atitudes e procedimentos adotados pelos alunos e pela professora durante esse processo. Nossas percepções indicam que o uso de um *software* com o intuito de auxiliar no processo de ensino e aprendizagem, mesmo que na área técnica de algoritmos, para construção de *software* apontam para um indispensável

entendimento sobre responsabilidades e “saberes” necessários para o exercício da docência.

2 METODOLOGIA DA PESQUISA

Este capítulo abrange o contexto da pesquisa e a caracterização genérica dos sujeitos envolvidos, além dos aspectos metodológicos gerais, com a descrição dos procedimentos utilizados.

2.1 Instituição e sujeitos envolvidos

O Instituto Federal Sul-Rio-Grandense (IFSUL), caracterizado pela verticalização do ensino, oferta educação profissional e tecnológica em diferentes níveis e modalidades de ensino, tem como missão "Implementar processos educativos, públicos e gratuitos, de ensino, pesquisa e extensão que possibilitem a formação integral mediante o conhecimento humanístico, científico e tecnológico e que ampliem as possibilidades de inclusão e desenvolvimento social " (IFSUL, 2014a, p. 26).

No Ifsul campus Passo Fundo, o curso superior de Tecnologia em Sistemas para Internet (TSPI) tem duração de seis semestres, em sua grade curricular apresenta disciplinas que contemplam as áreas de desenvolvimento de *software*, manutenção de *hardware*, sistemas operacionais, redes de computadores e gestão. Cada semestre do curso é dividido em duas etapas, o aluno que não obtiver êxito nas etapas oferecidas terá oportunidade de reavaliação e recuperação dos conteúdos no final do semestre.

O objetivo geral do curso (TSPI), segundo o seu Projeto Pedagógico é:

> Proporcionar ao aluno uma formação tecnológica na área de Informática que o permita atuar no planejamento, análise, desenvolvimento, avaliação e utilização de tecnologias emergentes empregadas em aplicações para a Web, sítios e portais para Internet e intranets, visando suprir as necessidades do mundo do trabalho. Objetiva-se também uma formação humanística e integral para que além de tecnólogos, os profissionais sejam cidadãos críticos e reflexivos capazes de compreender e atuar em sua realidade, explorando o uso das tecnologias com responsabilidade social (IFSUL, 2014b, p. 8).

No curso TSPI, em cada semestre, ingressam duas novas turmas de Algoritmos (AL), uma no turno da noite e outra no da manhã. Cada turma tem, aproximadamente, entre vinte e quarenta alunos, os quais geralmente apresentam níveis de conhecimento diferenciados, pois enquanto muitos são procedentes de cursos técnicos de informática, outros nunca ouviram falar dos conteúdos que compõem a disciplina.

Foram selecionados para participar desta pesquisa, os alunos da disciplina Algoritmos (AL) do curso de TSPI de duas turmas: a turma da manhã (1M1/2014-2) iniciou o semestre com trinta alunos e a da noite (1N1/2014-2) com 42 alunos.

A 1M1 que iniciou com trinta alunos finalizou a primeira etapa com 26, pois quatro desistiram nas primeiras aulas, dos 26 que permaneceram, cinco são repetentes e os demais são alunos novos. Dos 21 alunos novos, somente dois já haviam cursado essa disciplina em outros cursos de informática. A idade média dos 26 alunos iniciantes é de 29,1 anos, sendo que oito retomaram seus estudos com idade próxima de quarenta anos.

A 1N1 iniciou com 42 alunos e, no decorrer da primeira etapa, desistiram quinze, compondo a turma então 27 alunos, dos quais somente oito eram repetentes e já tinham tido contato com os conteúdos da disciplina. A idade média dos alunos da turma noturna é 26,11 e, entre os 27 alunos, cinco retomaram seus estudos próximos aos quarenta anos.

Uma característica que observamos nesse semestre é a idade média dos alunos, o que nos aponta para um aluno trabalhador, cada dia mais presente nas intuições de ensino superior, segundo estudos realizados por Cardoso e Sampaio ainda em 1994.

> Os estudantes que não trabalham, em quase todos os cursos, tendem a concentrar-se na faixa de idade mais jovem: entre dezoito e vinte anos. Para ambos os sexos, 20,6 anos é a idade média de quem não trabalha. A idade média dos estudantes homens que trabalham é de 24,6 anos, enquanto a das mulheres é de 22,6 anos (CARDOSO e SAMPAIO, 1994, p. 5).

As autoras concluem que as transformações ocorridas no ensino superior decorrente do aumento de vagas, a reorganização dos cursos e as carreiras do mercado de trabalho contribuíram para a entrada de trabalhadores entre os estudantes, fazendo diluir a distinção entre o estudo e trabalho. Elas ainda afirmam que a universidade, hoje, disputa com o ambiente de trabalho e já não é mais um local restrito à sociabilidade estudantil, pois mais de 50% dos estudantes trabalham.

Outra característica observada nas turmas de LA do curso do TSPI é a falta de comprometimento com a frequência, existem alunos matriculados que não frequentaram nenhuma das aulas, ou frequentaram poucos dias e desistiram. Outra particularidade dessas turmas é a relevante presença de alunos que admitem não ter conseguido fazer uma graduação e, agora, próximo aos trinta ou quarenta anos, retomam seus estudos, muitas vezes, não por ser o curso de sua preferência (o TSPI), mas pela oportunidade de estudar numa instituição federal sem custos.

2.2 Aspectos metodológicos gerais

Os questionamentos referentes à aprendizagem e às dificuldades dos alunos na disciplina de Algoritmos constituem um problema de ordem prática, da vida real, do dia a dia da sala de aula. Considerando que, para Minayo, a pesquisa é a atividade básica da ciência na sua indagação e construção da realidade, tal constatação está em consonância com a afirmação da autora, de que "Nada pode ser intelectualmente um problema se não tiver sido, em primeiro lugar um problema da vida prática. As questões de investigações estão, portanto, relacionadas a interesses e circunstâncias socialmente condicionadas" (2010, p. 16).

Para Esteban (2010), outra característica fundamental da pesquisa qualitativa é a atenção ao contexto, uma vez que acontecimentos e fenômenos não serão compreendidos adequadamente se separados desse. A autora conceitua pesquisa qualitativa como:

> [...] uma atividade sistemática orientada à compreensão em profundidade de fenômenos educativos e sociais, à transformação de práticas e cenários socioeducativos, à tomada de decisões e também ao descobrimento e desenvolvimento de um corpo organizado de conhecimentos (p. 127).

Segundo a autora, "nos estudos qualitativos, o próprio pesquisador se constitui no instrumento principal que, por meio da interação com a realidade, coleta dados sobre ela" (ESTEBAN, 2010, p. 129). Ela traz a expressão "*eu como pesquisador*", de Eisner (1998 apud ESTEBAN, 2010, p. 129), para destacar a importância adquirida pela pessoa que pesquisa na coleta de informações, afirmando, também, que o caráter interpretativo é outro traço que identifica esses estudos.

Essa mesma característica é apresentada pelos autores Bogdan e Biklen como um dos pontos comuns nas pesquisas qualitativas, os autores citam que, "Na investigação qualitativa a fonte direta de dados é o ambiente natural, constituindo o investigador o instrumento principal" (1994, p. 47). Eles mencionam, ainda, que os dados coletados são revistos na sua totalidade e que o entendimento que o investigador tem deles será o instrumento chave da análise.

Para reforçar a importância das pesquisas qualitativas Esteban (2010) afirma que "o momento atual reivindica uma *pesquisa qualitativa* cuja característica fundamental está na *reflexividade"* (p. 129). Para a autora, essa modalidade de pesquisa se propõe a dar atenção

à forma que diferentes elementos linguísticos, sociais, culturais, políticos e teóricos influem de maneira conjunta no processo de interpretação.

Os autores Bogdan e Biklen (1994) também trazem como característica das pesquisas qualitativas o maior interesse dos pesquisadores pelo processo do que simplesmente pelos resultados ou produtos. Para eles, as estratégias qualitativas expuseram como as expectativas do professor se traduzem nas atividades e interações diárias, focando no modo de como as percepções se formam, percepções essas que os professores têm dos alunos, que os alunos têm de si próprios e dos outros.

Outra consideração interessante apresentada por Minayo (2010) é a diferença entre abordagem quantitativa e qualitativa:

> [...] é de natureza e não de escala hierárquica. Enquanto os cientistas sociais que trabalham com estatísticas visam a criar modelos abstratos ou a descrever e explicar fenômenos que produzem regularidade, são recorrentes e exteriores aos sujeitos, a abordagem qualitativa se aprofunda no mundo dos significados (p. 22).

As considerações apresentadas serviram de base para compreender e justificar a escolha da abordagem qualitativa, uma vez que buscamos investigar as contribuições do uso de uma ferramenta gráfica no ensino-aprendizagem da disciplina de Algoritmos.

Para uma melhor organização, tomando por base os fundamentos de Minayo (2010), dividimos o processo em três etapas, que a autora denomina de ciclo de pesquisa: a primeira é chamada de exploratória, a segunda de trabalho de campo e a terceira de análise e tratamento do material empírico e documental.

A primeira fase é denominada de exploratória, considerando os procedimentos necessários para entrada em campo. Nessa fase, primeiramente, realizamos as leituras sobre pesquisa qualitativa, características da investigação qualitativa e buscamos no meio acadêmico outras pesquisas que abordassem o tema proposto, a fim de identificar metodologias, fundamentos, resultados, bem como ferramentas gráficas utilizadas para o ensino de Algoritmos. Ainda fizemos pesquisas em bibliografias específicas da área de informática, com o objetivo de definir algoritmos e fazer uma análise das diferentes formas de representá-los, apresentadas por distintos autores. Além disso, iniciamos a busca na literatura por autores que pudessem subsidiar teoricamente a pesquisa.

A segunda fase, característica do trabalho de campo, consiste em levar para a prática empírica a construção teórica elaborada na primeira fase, período em que ocorre o levantamento de material documental. Uma das atividades iniciais da pesquisa, de entrada

no trabalho de campo foi a escolha pela turma de Algoritmos para fazer parte da pesquisa, sendo que todo o semestre do curso de TSPI tem duas turmas, uma no turno da noite (1N1) e outra no da manhã (1M1). Para isso, lançamos, juntamente com um trabalho avaliativo da disciplina, uma pergunta relacionada à apresentação ou não de dificuldades com Algoritmos, perguntando se o aluno estaria encontrando dificuldades, o que deveria ser respondido com justificativa, tanto em caso afirmativo como em caso negativo (Apêndice A). Observamos, já nessa atividade inicial, uma pequena variação, ou seja, a turma 1M1 admitia estar encontrando mais dificuldades com a disciplina como mostram as tabelas a seguir (Tabela 1 e Tabela 2).

Tabela 1 - Dificuldades na disciplina (turma 1N1)

Turma 1N1 (n = 23)			
Sim		**Não**	
Entender como fazer/Interpretar a lógica	5	Estou entendendo	3
Quase tudo	1	Não especificou	2
Comando If/Else/Switch	4	Por já ter cursado esta disciplina	2
Conversão para a linguagem C	1	Aulas bem explicativas	1
Turma muito grande	1	Aulas bem ministradas	1
Encontrar erros compilação	1	Já trabalho na área	1
Total SIM:	**13**	**Total Não:**	**10**
% SIM :	**56,5**	**% Não:**	**43,5**

Fonte: da pesquisa.

Dos 27 alunos da 1N1, 23 responderam ao questionário apresentando suas justificativas, porém, na turma 1M1, dos 26 alunos, 25 responderam.

Tabela 2 - Dificuldades na disciplina (turma 1M1).

Turma 1M1 (n = 25)			
Sim		**Não**	
Não consigo pensar/ Falta raciocínio	1	Gosto de matemática	1
Não consigo interpretar o texto e montar	4	A professora explica muito bem	1
Faltei aulas	1	Com os exercícios vai ficando mais claro	2
Os cálculos matemáticos	1	Gostei da matéria e estou estudando bastante	1
Conversão para a linguagem C	1	Por já ter cursado esta disciplina	3
Manipulação vetores e matrizes	4	No início está sendo fácil	1
Falta de atenção	1		
Falta tempo para estudar	2		
Conta com muito raciocínio	1		
Total SIM:	**16**	**Total Não:**	**9**
% SIM :	**64**	**% Não:**	**36**

Fonte: da pesquisa.

Observamos que, na turma 1M1, um percentual de 64% dos alunos admitiram estar se deparando com dificuldades quanto à aprendizagem de algoritmos, enquanto na turma 1N1 esse valor é um pouco menor, 56% reconheceram estar encontrando dificuldades.

Com a correção do trabalho avaliativo e a aplicação da avaliação (prova) da etapa, confirmamos, então, a maior dificuldade apresentada na turma 1M1 em relação a turma 1N1. Outro aspecto analisado foi quanto à média geral da turma 1M1 que foi de 4,41, enquanto a da turma 1N1 foi de 5,18, apresentando, assim, novamente um melhor desempenho em relação a turma da manhã. Com base nessas informações, optamos, então, pela turma 1M1 para aplicação desta pesquisa.

Nessa fase, também iniciamos o processo de cadastramento do projeto de pesquisa no comitê de ética da plataforma Brasil, ao que se seguiu a investigação, com o início da aplicação da ferramenta gráfica escolhida, a observação e o registro das atividades realizadas com os alunos participantes. A aplicação da ferramenta iniciou na segunda quinzena do mês de outubro, faltando praticamente pouco mais de um mês para finalizar as aulas do semestre.

A terceira e última etapa foi estabelecida como análise e tratamento do material empírico e documental, fase necessária para interpretar e compreender os dados, bem como articulá-los com teorias que fundamentaram a pesquisa ou ainda, relacioná-los com outras leituras teóricas e interpretativas.

Por meio dessa última etapa, desse processo de análise de dados, verificamos que o *software* utilizado possibilitou que identificássemos dificuldades conceituais da construção de algoritmos. Nessa análise do processo vivenciado pelos alunos na utilização do *Blockly*, identificamos quatro categorias principais: interpretar enunciados; comandos; posicionar-se em relação ao objeto e criar representações livres. Categorias que serão descritas no capítulo cinco dessa dissertação.

3 FUNDAMENTOS PARA O PROCESSO DA PESQUISA

Neste capítulo, apresentamos os fundamentos para a pesquisa, iniciando com conceitos de algoritmos, em seguida, apresentamos no referencial teórico as teorias que embasam esta pesquisa. Primeiramente, abordamos reflexões sobre algumas noções da didática da matemática. Abordamos a resolução de problemas por meio de uma abordagem cognitiva, buscamos trazer alguns autores que articulam acerca do desenvolvimento geral da capacidade de raciocínio dos alunos e o desenvolvimento das suas representações mentais como a teoria dos registros e representações e semióticas de Duval. Buscamos contribuições da psicologia sobre o desenvolvimento do pensamento por meio do conceito das funções psicológicas superiores ou dos processos mentais superiores de Vygotsky.

Para finalizar este capítulo, apresentamos pesquisas que abordaram temas de aprendizagem de algoritmos e uso de ferramentas gráficas.

3.1 Sobre algoritmos

Especificamente sobre algoritmos, apresentamos as definições, as formas de representação e os conceitos para a construção da investigação proposta. Assim, também, expomos as ferramentas gráficas analisadas para o ensino de algoritmos e a ferramenta *Blockly* selecionada e utilizada nesta pesquisa.

3.1.1 Definições e formas de representação de algoritmos

Os conceitos envolvidos na disciplina de Algoritmos vão além dos conceitos de programação e da simples interpretação da sintaxe de uma linguagem de programação, pois estão relacionados a conteúdos da matemática. Brookshear (2005) declara que o estudo dos algoritmos começou como um objetivo da matemática e que a procura por algoritmos era uma atividade significativa dos matemáticos muito antes do desenvolvimento dos computadores. Para o autor, esses matemáticos tinham como objetivo principal descobrir um conjunto único de diretrizes que descrevessem como todos os problemas de um determinado tipo poderiam ser resolvidos.

Um algoritmo é a definição de uma atividade que desejamos que o computador execute ou uma série de passos que devem ser executadas com uma determinada ordem pré-definida pelo programador. O papel do programador é intermediar o diálogo entre o

computador e o usuário, portanto para fazer com que esse diálogo ocorra, ou seja, para que o usuário e o computador se entendam é preciso um *software* (um programa). Um *software*, por sua vez, é construído com algoritmos que, escritos em uma linguagem de programação, tornam possível a execução em um computador ou qualquer dispositivo computacional. Assim, por meio desses *softwares* desenvolvidos com algoritmos, temos a impressão que estamos dando vida às máquinas.

Para que o computador entenda nosso algoritmo, nossas instruções devem ser escritas em linguagens de programação, que assim é definida por Puga e Rissetti: "A linguagem de programação, como qualquer outra linguagem, é formada por palavras. Estas são agrupadas em frases para produzir um determinado significado" (2009, p. 20). Os autores ainda comparam essas palavras com "palavras-chaves" e as frases criadas com a "estrutura de programação", bem como as regras dessa linguagem e sua gramática com a sintaxe da linguagem. Citam como exemplos algumas linguagens mais utilizadas: Pascal, C, C++ e Java.

Berlinski (2002) afirma "que todo computador se divide entre *hardware* e *software*, a máquina hospedeira de seu algoritmo, o ser humano de sua mente. Não é de surpreender que homens e mulheres tenham feito o que os computadores agora fazem muito antes que eles pudessem fazer qualquer coisa" (p. 12).

Forbellone e Eberspächer (2005) definem que "um algoritmo pode ser definido como uma sequência de passo que visam a atingir um objetivo bem definido" (p. 3). Os autores ainda mencionam que no momento de especificar esta sequência de passos é preciso utilizar uma ordem, ou ainda, de "pensar com ordem" e explicam que,

> Quando elaboramos um algoritmo, devemos especificar ações claras e precisas, que a partir de um estado inicial, após um período de tempo finito, produzem um estado final previsível e bem definidas. Isso significa que o algoritmo fixa um padrão de comportamento a ser seguido, uma norma de execução a ser trilhada, com vistas a alcançar, como resultado final, a solução de um problema, garantindo que sempre que executado, sob as mesmas condições, produza o mesmo resultado (p. 3).

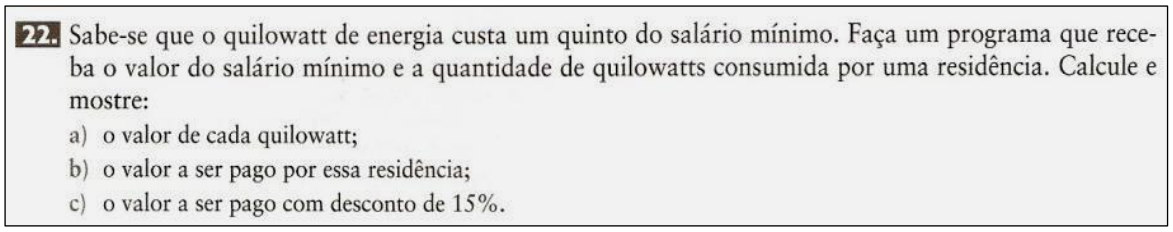
Já os autores Lopes e Garcia citam o fato de que não existe um único algoritmo capaz de resolver um determinado problema, pois existem muitas maneiras de se revolver o mesmo problema. Afirmam que "Algoritmo é um conjunto de passos (ações) que levam à solução de determinado problema, ou, então, é um caminho para a solução de um

problema e, em geral, os caminhos que levam a uma solução são muitos" (LOPES; GARCIA, 2002, p. 1).

Um exemplo clássico de algoritmo utilizado por vários autores inclusive por Forbellone e Eberspächer (2005) é a receita de um bolo, pois nela se indicam os ingredientes e a definição de uma sequência de passos (ações) que devem ser fielmente seguidos para que se consiga o objetivo desejado.

Um exemplo de enunciado clássico, utilizado nos livros de algoritmos, é ilustrado na Figura 1, o qual solicita a construção de um algoritmo para calcular o valor a ser pago de energia elétrica por uma residência.

Figura 1- Exemplo de um exercício de Algoritmos



22. Sabe-se que o quilowatt de energia custa um quinto do salário mínimo. Faça um programa que receba o valor do salário mínimo e a quantidade de quilowatts consumida por uma residência. Calcule e mostre:

a) o valor de cada quilowatt;
b) o valor a ser pago por essa residência;
c) o valor a ser pago com desconto de 15%.

Fonte: ASCENCIO; CAMPOS, 2012, p. 48.

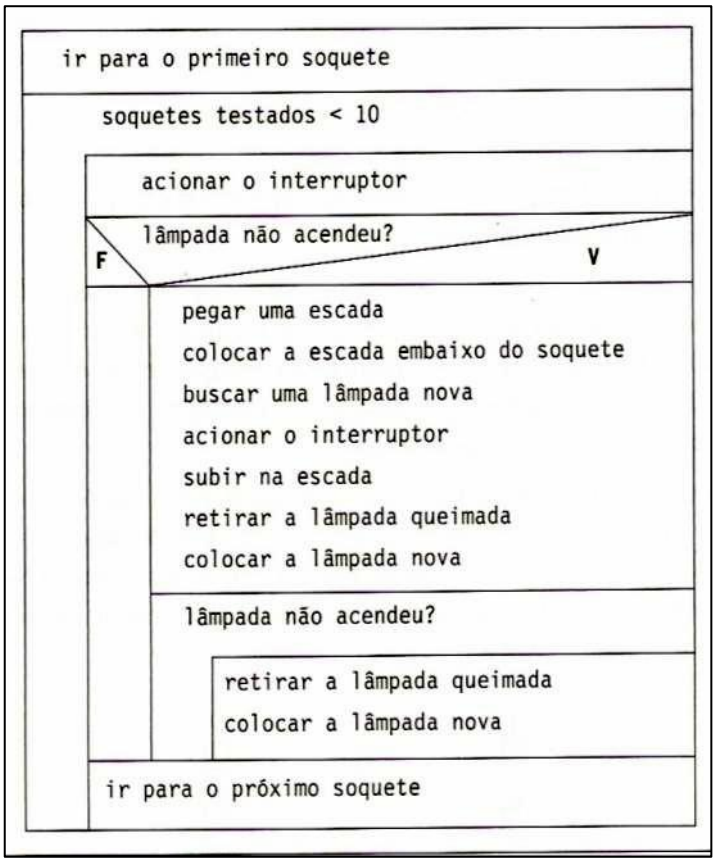
A proposta do exercício é criar a sequência de passos necessária para resolver o problema. Para a construção de um algoritmo, os autores apresentam como proposta um método com os seguintes passos:

1) Compreender completamente o problema a ser resolvido, destacando os pontos mais importantes e os objetos que o compõem.
2) Definir os dados de entrada, ou seja, quais dados serão fornecidos e quais objetos fazem parte desse cenário problema.
3) Definir o processamento, ou seja, quais cálculos serão efetuados e quais as restrições para esses cálculos. O processamento é responsável pela transformação dos dados de entrada em dados de saída. Além disso, deve-se verificar quais objetos são responsáveis pelas atividades.
4) Definir os dados de saída, ou seja, quais dados serão gerados depois do processamento.
5) Construir o algoritmo utilizando um dos tipos.
6) Testar o algoritmo realizando simulações (ASCENCIO; CAMPOS, 2012, p. 3).

Puga e Rissetti (2009) apresentam quatro formas de como representar um algoritmo: diagrama de Chapin, descrição narrativa, pseudocódigo e fluxograma. Mas destacam que o fluxograma e o pseudocódigo são as duas formas mais comuns de representação.

O diagrama de Chapin (Figura 2) apresenta a solução por meio de diagrama de quadros com uma visão hierárquica e estruturada.

Figura 2 - Diagrama de Chapin



Fonte: FORBELLONE; EBERSPÄCHER, 2005, p. 11.

A proposta desse diagrama é representar as ações de um algoritmo dentro de um único retângulo, subdividindo-o em retângulos menores, que representam os diferentes caminhos ou blocos de sequência de ações do algoritmo.

Outra forma de representar um algoritmo é a descrição narrativa, que utiliza linguagem natural para especificar os passos, que, segundo os autores, dá margem a interpretações errôneas e ambiguidades inerentes a sua característica pouco formal.

A forma pseudocódigo, também chamada por outros autores como Forbellone e Eberspächer (2005) de português estruturado, ou ainda, por Ascencio e Campos (2012), denominada Portugol, é uma forma de representação de algoritmos que utiliza uma linguagem flexível e intermediária entre a natural e a de programação. É usada para organizar o raciocínio lógico a ser seguido e apresenta os comandos necessários para a resolução de um determinado problema. No enunciado de um algoritmo, apresentado por

Ascencio e Campos (2012), em que é solicitado o cálculo do valor a ser pago de energia elétrica por uma residência, os autores apontam todos os comandos e fórmulas matemáticas necessárias para a resolução do problema em questão (Figura 3).

Figura 3 – Exemplo 1 de algoritmo em pseudocódigo

```
ALGORITMO
  DECLARE vlr_sal, qtd_kw, vlr_kw, vlr_reais, desc, vlr_desc NUMÉRICO
  LEIA vlr_sal
  LEIA qtd_kw
  vlr_kw ← vlr_sal / 5
  vlr_reais ← vlr_kw * qtd_kw
  desc ← vlr_reais * 15 / 100
  vlr_desc ← vlr_reais − desc
  ESCREVA vlr_kw
  ESCREVA vlr_reais
  ESCREVA vlr_desc
FIM_ALGORITMO.
```

Fonte: ASCENCIO; CAMPOS, 2012, p. 48.

Inicialmente, no algoritmo apresentado (Figura 3) são declaradas todas as variáveis, com o comando "DECLARE". Puga e Rissetti ao conceituar variáveis afirmam que "Nos algoritmos, as variáveis são utilizadas para representar valores desconhecidos, porém necessários para a resolução do problema, os quais poderão ser alterados de acordo com a situação, armazenando valores, dados temporariamente" (2009, p. 38). Assim, após a declaração das variáveis, essas são utilizadas para receber os valores necessários para o cálculo e para mostrar os valores solicitados pelo enunciado. A variável declarada como vlr_sal é utilizada para ler o valor do salário mínimo e a variável qtd_kw para a quantidade de quilowatts consumida na residência. Em seguida, é feita a leitura dessas variáveis com os comandos "LEIA vlr_sal" "LEIA qtd_kw", ou seja, "fazer a leitura" significa que o valor digitado via teclado, será armazenado nelas. A variável vlr_kw é utilizada para calcular o valor de 1(um) quilowatt, pois, segundo o enunciado, o valor de cada quilowatt é 1/5 do salário mínimo, portanto, essa variável recebe o salário mínimo dividido por cinco por meio da expressão (vlr_kw ← vlr_sal / 5).

Na próxima linha, é calculado o valor da conta, multiplicando o valor de um quilowatt (vlr_kw) pela quantidade de quilowatts gasto na residência (vlr_reais ← vlr_kw * qtd_kw). Logo em seguida, é calculado o valor do desconto sobre o valor da conta recém-calculada na variável vlr_reais, para isso, é realizada a operação de multiplicação da variável vlr_reais por quinze e dividido por cem. É ainda calculado o valor com o desconto, a variável vlr_desc recebe o valor da conta (vlr_reais) menos o desconto

(vlr_desc). Para finalizar o algoritmo, são apresentados os valores solicitados: o valor de cada quilowatt na variável vlr_kw; o valor a ser pago pela residência na vlr_reais e, por último, o valor a ser pago com 15% de desconto com a variável vlr_desc.

Ainda sobre pseudocódigo, Ascencio e Campos (2012) descrevem que a sua construção consiste em analisar o enunciado e, por meio de regras predefinidas, escrever as instruções a serem seguidas para a resolução de um problema. Como vantagem, os autores citam que a passagem do algoritmo para qualquer linguagem de programação é quase imediata, bastando conhecer as palavras, ou seja, a sintaxe da linguagem a ser utilizada. Como desvantagem, citam a necessidade de aprender as regras do pseudocódigo.

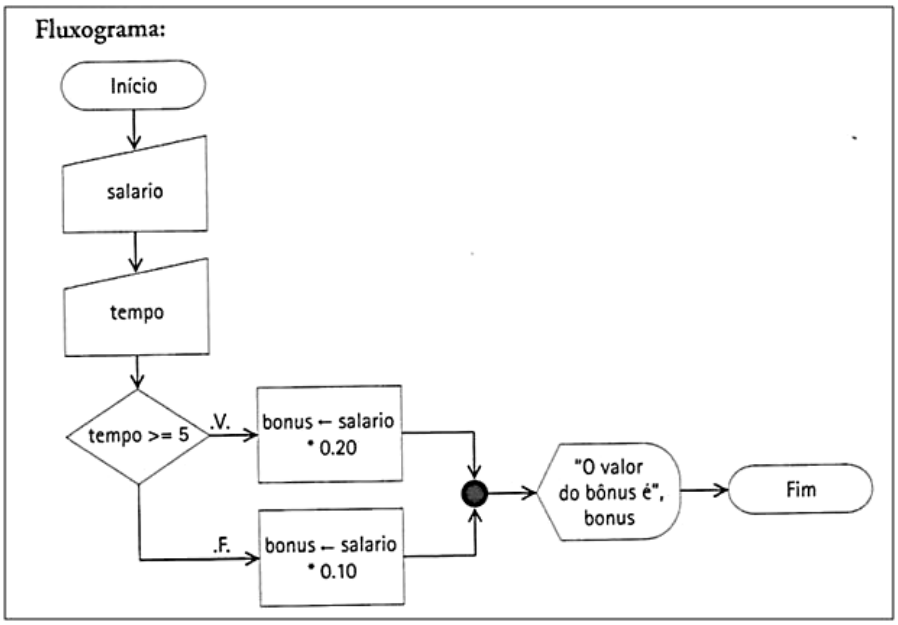
A quarta e última forma de representar um algoritmo é o fluxograma, esse utiliza figuras geométricas para ilustrar os passos a serem seguidos, também é conhecido como diagrama de blocos, mas não é recomendado para algoritmos extensos. Assim como o pseudocódigo, o fluxograma é utilizado para organizar o raciocínio lógico, ou seja, constituem os passos estabelecidos para a resolução de um problema ou a execução de uma tarefa. Mas como já foi mencionado, os autores só recomendam seu uso para algoritmos menores (PUGA; RISSETTTI, 2005).

Forbellone e Eberspächer (2005) salientam que as técnicas, tanto da forma gráfica como da textual permitem um nível grande de clareza quanto ao fluxo de execução. Entretanto destacam que é mais difícil entender as representações apresentadas de maneira gráfica pela necessidade de entender suas convenções, já que somos mais condicionados a nos expressar por palavras. Citam ainda como desvantagem o fato de ser mais trabalhoso fazer um desenho do que escrever um texto, o que consideram que pode desencorajar a construção de algoritmos.

No exemplo listado a seguir, apresentado por Puga e Risseti (2009), é representado o mesmo algoritmo nos formatos de fluxograma (Figura 4) e de pseudocódigo (Figura 5), que são os dois formatos mais utilizados e encontrados nos livros de lógica de programação. O enunciado do algoritmo solicita que seja calculado o valor de um bônus de acordo com o tempo de trabalho na empresa, conforme se pode observar nas palavras dos autores: “A empresa XSoftware Ltda concedeu um bônus de 20 por cento do valor do salário a todos os funcionários com tempo de trabalho na empresa igual ou superior a cinco anos e de 10 por cento aos demais. Calcular e exibir o valor do bônus” (PUGA; RISSETI, 2009, p. 59).

Os autores explicam que para resolver o problema é necessário conhecer o valor do salário e o tempo de serviço do funcionário. Assim, são utilizadas as variáveis “salário” e “tempo” para representar esses valores e para armazenar o valor do bônus calculado é utilizado a variável bônus. Apresentamos, a seguir, esse algoritmo resolvido no formato de fluxograma criado pelos autores:

Figura 4 - Exemplo de algoritmo em fluxograma



Fonte: PUGA; RISSETTI, 2009, p. 60.

Os autores expõem, ainda, o significado de cada símbolo do fluxograma da seguinte forma:

Terminal: representa início e fim .

Processamento: execução de operações, cálculos aritméticos, atribuição de valores a variáveis.

Teclado: entrada e saída de dados para variáveis via teclado.

Vídeo: saída de informações via vídeo ou outro dispositivo visual de saída de dados.

Decisão: uma ação lógica que apresentará o resultado de “verdadeiro” ou “falso”, sendo que cada uma realizará uma sequência já estabelecida.

Conector: interliga partes do fluxograma ou desvia o fluxo para um determinado trecho.

→ **Seta de Orientação do fluxo:** orienta a sequência de execução ou leitura que pode ocorrer de forma horizontal ou vertical.

Portanto, esse mesmo algoritmo, escrito no formato de pseudocódigo (Figura 5), apresenta estrutura bem diferente para o mesmo algoritmo que calcula o valor de um bônus de acordo com o tempo de trabalho na empresa.

Figura 5 - Exemplo2 de algoritmo em pseudocódigo

Pseudocódigo:

```
1.      Algoritmo Premio
2.      Var
3.        salario, bonus: real
4.        tempo: inteiro
5.      Início
6.        Ler (salario)
7.        Ler (tempo)
8.        Se (tempo >= 5) então
9.          bonus  ←  salario * 0.20
10.       Senão
11.         bonus  ← salario * 0.10
12.       Fim-Se
13.       Mostrar ("O valor do bônus é", bonus)
14.     Fim.
```

Fonte: PUGA; RISSETTI, 2009, p. 60.

Apesar de haver consenso entre os autores quanto à forma mais indicada e utilizada para representação dos algoritmos, a saber: o pseudocódigo, existe um fator que vem a ser desencorajador para os principiantes na construção dos algoritmos que diz respeito ao fato de cada autor criar um estilo de pseudocódigo, não existindo um padrão, cada autor apresenta o "seu".

Para exemplificar essa variação, apresentamos, a seguir, um algoritmo de repetição de cada um dos autores já referenciados. Algoritmos de repetição também são conhecidos por sua tradução em inglês de *loops* ou *looping*, ganham esse nome por representarem uma execução finita em círculo, que depois segue seu curso normal. Repetição consiste em uma estrutura de controle do fluxo de execução que permite repetir diversas vezes um mesmo trecho do algoritmo (FORBELLONE; EBERSPÄCHER, 2005).

As diferenças apresentadas em cada pseudocódigo, geralmente, estão atreladas à linguagem de programação que o autor do livro domina, aquela que usará para programar tais algoritmos. Nos exemplos apresentados nas Figuras 6 e 7, a declaração de variáveis é

destacada com a inserção da palavra VAR e DECLARE, assim como é exigido pela sintaxe da linguagem de programação Pascal. Enquanto nas Figuras 8 e 9, que são mais próximas da linguagem C, não é utilizada a especificação VAR e, sim, a especificação do tipo de dado que será armazenado na variável.

Enquanto alguns pseudocódigos utilizam para ler uma variável o "LEIA xx", (Figuras 6, 8 e 9) outro utiliza "LER (xx)" (Figura 7), também podemos observar alguns comandos de leitura que utilizam parênteses e outros que não mais utilizam esse recurso. Outro comando que apresenta diferenças na sua forma de apresentação é o "ESCREVA xxx", comando utilizado para mostrar no vídeo o resultado dos cálculos executados no algoritmo, enquanto alguns utilizam o "ESCREVA xxx", outros o "Mostrar (xx)", ainda nesses exemplos, temos o "imprima xx".

Figura 6 – Modelo de pseudocódigo conforme Campos

```
ALGORITMO
DECLARE i, ano_atual, salario NUMÉRICO
        novo_salario, percentual NUMÉRICO
LEIA ano_atual
salario ← 1000
percentual ← 1,5/100
novo_salario ← salario + percentual * salario
PARA i ← 2007 ATÉ ano_atual FAÇA
INÍCIO
percentual ← 2 * percentual
novo_salario ← novo_salario + percentual * novo_salario
FIM
ESCREVA novo_salario
FIM_ALGORITMO.
```

Fonte: ASCENCIO; CAMPOS, 2007, p. 108.

Figura 7 - Modelo de pseudocódigo conforme Puga e Rissetti

Pseudocódigo:

```
1.      Algoritmo ex_para
2.      Var
3.        soma, num, media: real
4.        cont: inteiro
5.      Início
6.        soma ← 0
7.        Para cont ← 1 até 850 Passo 1 Faça
8.          Ler (num)
9.          soma ← soma + num
10.       Fim-para
11.       media ← soma / cont
12.       Mostrar ("Média= ", media)
13.     Fim.
```

Fonte: PUGA; RISSETTI, 2009, p. 77.

Observamos que os pseudocódigos não apresentam um padrão quanto à escrita de comandos, nem com relação às letras maiúsculas ou minúsculas. Alguns autores utilizam letras maiúsculas nos comandos para diferenciá-los das outras operações matemáticas e das variáveis (Figura 6), enquanto outros utilizam somente letras minúsculas (Figuras 8 e 9) e há, ainda, aqueles que mesclam, utilizando a primeira letra da palavra maiúscula (Figura 7) e, para o restante dessa, letras minúsculas.

Figura 8 - Modelo de pseudocódigo conforme Forbellone e Eberspächer

EXERCÍCIO 3.4 (página 61)

```
1. início
2.    real: H; // resultado da série
3.    inteiro: N, // denominador fornecido pelo usuário
4.             V; // variável de controle
5.    leia (N);
6.    H ← 0;
7.    para V de 1 até H passo 1 faça
8.       H ← H + 1 / V;
9.    fimpara;
10.   escreva ("Resultado da série = ", H);
11. fim.
```

Fonte: FORBELLONE; EBERSPÄCHER, 2005, p. 189.

Figura 9 - Modelo de pseudocódigo conforme Lopes e Garcia

```
prog para28
 int ini, vf, soma, i;
 imprima "\ndigite valor inicial e valor final de um intervalo,
          pressionando enter apos digitar cada um: ";
 leia ini;
 leia vf;
 soma <-0;
 se( ini % 2 == 0)
 { ini <- ini +  2; }
 senao
 { ini <- ini +  1; }
 vf--;
 para( i <- ini ; i <= vf;  i <- i +  2)
 {
    soma <- soma + i;
    imprima i, "   ";
 }
 imprima "\nsoma = ", soma;
 imprima "\n";
fimprog
```

Fonte: LOPES; GARCIA, 2002, p. 146.

Percebemos que existe certa aproximação com a linguagem de programação Pascal nos exemplos dos autores Puga e Rissetti (Figura 7), como no dos autores Forbellone e Eberspächer (Figura 8), principalmente com o comando de repetição “para” que faz uso no final da palavra “Fim-Para”, como exigido na sintaxe da linguagem de programação Pascal. Portanto, o mesmo comando representado no exemplo dos autores Lopes e Garcia é, de outra forma, mais próximo da linguagem C (Figura 9), pois faz uso da chave “{ }” para sinalizar início e fim de comandos.

Embora exista essa aproximação com as linguagens de programação, observamos falta de conformidade na elaboração dos pseudocódigos apresentados nos livros de algoritmos e esse fator pode consistir em mais um obstáculo para a aprendizagem desses, sobretudo para alunos iniciantes.

Portanto, para construir um algoritmo utilizando pseudocódigo, é necessário adotarmos uma linguagem, definirmos uma sintaxe básica. Os elementos desse pseudocódigo representam os conceitos básicos da disciplina de Algoritmo que são: *variáveis; comandos de entrada e saída; operadores: aritméticos, relacionais e lógicos; estruturas de seleção; estruturas de repetição, vetores, matrizes e Funções*. E para esses conceitos básicos descreveremos brevemente suas principais definições.

Uma *variável* pode assumir qualquer valor, de acordo com o tipo de dado selecionado na sua definição e tem a possibilidade de ser alterada durante a execução do algoritmo. As linguagens de programação contêm alguns conjuntos de domínio pré-definidos, os quais são denominados de tipos de dados nativos, são eles: Inteiro, Real, Caracter e Lógico.

Quando um algoritmo necessita de informações externas, as quais devem ser inseridas no computador, é necessário haver uma *entrada de dados*. Por exemplo, um programa que permita pagamento com cartão de crédito precisa que o usuário forneça o número do cartão e a senha. Esses dados podem ser captados via cartão magnético ou teclado e serão algumas das variáveis do programa.

Forbellone e Eberspächer, sobre os comandos de entrada e saída, afirmam que “Os algoritmos precisam ser ‘alimentados’ com dados provenientes do meio externo para efetuarem as operações e os cálculos que são necessários a fim de alcançar o resultado desejado” (2005, p. 26). Logo após os dados de entrada terem sido processados, mostram-se ou gravam-se os resultados num dispositivo de *saída de dados*: impressora, monitor de vídeo, discos.

Os *operadores aritméticos*, na definição de Forbellone e Eberspächer, constituem "o conjunto de símbolos que representam as operações básicas da matemática" (2005, p. 19). Esses autores listam como exemplo os operadores: + adição, - subtração, * multiplicação e / divisão. Os *operadores relacionais* são utilizados para "realizar comparações entre dois valores de mesmo tipo primitivo. Tais valores são representados por constantes, variáveis e expressões" (2005, p. 21). Forbellone e Eberspächer listam como exemplo de operadores relacionais os apresentados na Tabela 3.

Tabela 3 - Operadores relacionais

Operador	Função	Exemplos
= =	Igual a	3 = 3, X = Y
>	Maior que	5 > 4, X > Y
<	Menor que	3 < 6, X < Y
> =	Maior ou igual a	5 > = 3, X > = Y
< =	Menor ou igual a	3 < = 5, X < = Y
< >	Diferente de	8 < > 9, X < > Y

Fonte: FORBELLONE; EBERSPÄCHER, 2005, p. 21.

Ainda sobre operadores relacionais, os autores salientam que o resultado obtido de uma relação é sempre um valor lógico.

Puga e Rissetti definem *operadores lógicos* como os que "são utilizados para estabelecer uma relação de comparação entre valores ou expressões. O resultado desta comparação é sempre um valor lógico (booleano) verdadeiro ou falso" (2009, p. 41). Os autores listam exemplos de operadores lógicos, conforme Tabela 4.

Tabela 4 - Operadores lógicos

Operador	Representação, utilizando-se a notação algorítmica	Representação, utilizando-se a notação para linguagem Java	Exemplos em Java
e	. e .	& &	a = 5 && b != 9 (se o valor de a for igual a 5 e o valor de b diferente de 9, então retornará verdadeiro. Caso contrário, retornará falso)
ou	. ou .	| |	a = 5 || b != 9 (se o valor de a for igual a 5 ou o valor de b diferente de 9, então retornará verdadeiro. Se ambas as comparações retornarem falso, retornará falso)
não	.não.	!	! a > 5 (se o valor de a for maior do que 5, retornará falso. Caso contrário, retornará verdadeiro)

Fonte: PUGA; RISSETTI, 2009, p. 42.

Esses autores observam que para o operador “. e .”, o resultado será verdadeiro somente se ambas as expressões associadas assumirem o resultado verdadeiro; para o operador “. ou .” o resultado será verdadeiro se pelo menos uma das expressões associadas assumir o resultado verdadeiro.

A *estrutura de seleção* “permite a escolha do grupo de ações (blocos) a serem executadas quando determinadas condições, representadas por expressões lógicas ou relacionais, são ou não satisfeitas” (FORBELLONE; EBERSPÄCHER, 2005, p. 189). Essa estrutura pode ser simples ou composta. Simples quando precisamos verificar uma condição antes de executar uma ação; composta para quando houver situações em que duas alternativas de uma mesma condição precisam ser verificadas, uma diz respeito à condição ser verdadeira e a outra à condição ser falsa.

Apresentamos, a seguir, as estruturas de seleção simples e composta de Forbellone e Eberspächer (2005). No exemplo da seleção simples (Figura 10), quando a expressão condicional é verdadeira, o bloco de comandos que segue a palavra "então" é executado e a sequência de comandos C1;...; Cn é executada.

Figura 10 - Exemplo de seleção simples

```
se <condição>
   então
      início // início do bloco verdade
         C1;
         C2; // seqüência de comandos
          .
         .  .
         Cn;
      fim;    // fim do bloco verdade
fimse;
```

Fonte: FORBELLONE; EBERSPÄCHER, 2005, p. 34.

Como citado anteriormente, se for necessário decidir também quais as ações a serem executadas, quando a condição for falsa, utilizamos a seleção composta, assim, o bloco verdade continua existindo, porém, será definida a sequência de comandos, para quando o resultado da condição seja falso, executando o bloco que segue a palavra senão. Como ilustrado no exemplo que segue, se a condição for verdadeira, então, a sequência de comandos do *bloco verdade* C1;...; Cn é executada; caso contrário, se a condição for falsa, a sequência do bloco falsidade C1;...; Cn será executada (Figura 11).

Figura 11 - Exemplo de seleção composta

```
se <condição>
   então
      início // início do bloco verdade
            C1;
            C2; // seqüência de comandos
            .
            .
            .
            Cn;
      fim; // fim do bloco verdade
   senão
      início // início do bloco falsidade
            C1;
            C2; // seqüência de comandos
            .
            .
            Cn;
      fim; // fim do bloco falsidade
fimse;
```

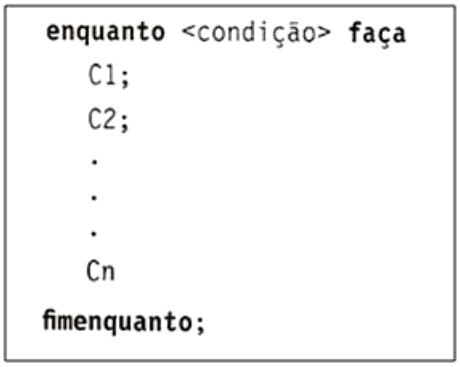
Fonte: FORBELLONE; EBERSPÄCHER, 2005, p. 36.

Existem situações nas quais precisamos repetir certos comandos, ou certas sequências de comandos para resolver problemas computacionais. Uma solução é o uso de estruturas *de repetição* ou (*looping*), as quais permitem que um bloco de comandos seja executado uma ou mais vezes, quando determinadas condições, representadas por expressões lógicas, são, ou não, satisfeitas.

As estruturas de repetição apresentadas por Forbellone e Eberspächer (2005) apresentam três formas: repetição com teste no início; repetição com teste no final e repetição com variável de controle.

Repetição com teste no início repete diversas vezes um mesmo trecho de comandos, porém são executados se a condição resultar em valor verdadeiro; nesse caso, o comando utilizado é o Enquanto. A sequência de comandos C1;...; Cn é executada enquanto a <condição> for verdadeira. Quando ela se tornar falsa, a execução do algoritmo passa ao comando após o "fim enquanto" (Figura 12).

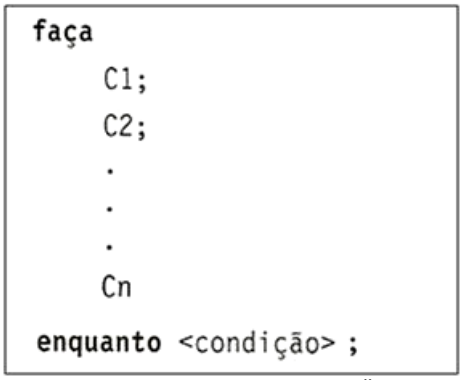
Figura 12 - Repetição teste no início



Fonte: FORBELLONE; EBERSPÄCHER, 2005, p. 48.

Outra forma de controle das estruturas de repetição é realizada com um teste no final, em que os comandos são executados pelo menos uma vez e, então, a condição é avaliada. Caso a condição resulte em valor verdadeiro, o processo é repetido. Esse comando é o "repita – enquanto" (Figura 13). Nesse comando, a sequência de comandos C1;...; Cn é executada uma vez, então, se a <condição> for verdadeira, a execução se repete enquanto a <condição> continuar verdadeira. Quando ela se tornar falsa, a execução do programa passa ao comando seguinte.

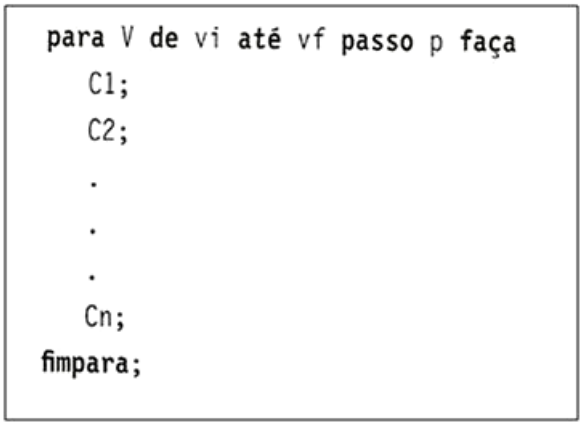
Figura 13 - Repetição teste no final



Fonte: FORBELLONE; EBERSPÄCHER, 2005, p. 53.

Nas estruturas de repetição vistas, ocorrem casos em que se torna difícil determinar o número de vezes em que um bloco será executado. A repetição com variável de controle reproduz a execução do bloco de comando a um número predeterminado de vezes, pois há limites fixos. Vejamos, a seguir, o modelo genérico para a estrutura de repetição "para".

Figura 14 - Repetição com variável controle



```
para V de vi até vf passo p faça
   C1;
   C2;
   .
   .
   .
   Cn;
fimpara;
```

Fonte: FORBELLONE; EBERSPÄCHER, 2005, p. 53.

No exemplo apresentado (Figura 14), sendo a variável de controle a “V”, essa terá o valor inicial estabelecido pelo “vi” e o valor final de “V” pelo valor de “vf”, ou seja, a variável “V” inicia em “vi” e vai até chegar em “vf”. A variável “p” é o valor de incremento da variável “V”, assim, a sequência de comandos “C1;...; Cn” é executada um número de vezes entre o intervalo de “vi” até “vf”, determinada pelo passo “p”. Trazemos como exemplo, se “vi” tiver o valor de 1, “vf” de 10 e passo “p” de 2 a sequência de comandos inseridas no laço executará cinco vezes.

Outro conceito muito utilizado na disciplina de Algoritmos *é de vetores e matrizes*, na definição de Ascencio e Campos (2012), “se trata [sic] de um conjunto de variáveis de mesmo tipo, que possui o mesmo indicador (nome) e são alocadas sequencialmente na memória”, (p. 151). Como essas variáveis têm o mesmo nome, o que as distingue é um índice que referencia sua localização dentro do vetor. O que diferencia um vetor de outra variável é a presença de colchetes logo após o seu nome, no momento da declaração.

No exemplo de vetores apresentados por Lopes e Garcia (2002), são definidos dois vetores, o primeiro denominado “a” com tamanho 10, o segundo é o “b” de tipo real com tamanho também de 10 (Figura 15).

real a[10] , b[10];

Figura 15 - Exemplo de vetor

Criar um algoritmo que leia um vetor A de dez valores e construa outro vetor B, da seguinte forma:

Ex.: Vetor A	*3*	*8*	*4*	*2*	*...*	*5*
Vetor B	*9*	*4*	*12*	*1*	*...*	*15*

```
prog vetor24
 int L ;
 real a[10], b[10];
 #vetor com 10 elementos
 para( L<- 0;L <= 9 ; L++)
 {
    imprima "\ndigite numero :";
    leia a[L ];
    se(L   % 2 == 0)
    { b[L ] <- a[L ] / 2; }
     senao
     { b[L ] <- a[L ] * 3; }
 }
```

Fonte: LOPES; GARCIA, 2002, p. 295.

No exemplo (Figura 15), estão representados dois vetores, o vetor “a” que tem dez posições, iniciando na 0 e finalizando na posição 9, sendo que em cada uma das posições poderão ser armazenados números reais. O outro vetor ilustrado é o “b”, também de 10 posições, em que armazenará números reais gerados a partir dos valores de “a”, observando a regra : se o valor estiver em uma posição par (se L % 2 = = 0) é dividido por dois (b[L] <- a[L] /2), caso contrário, ou seja, se estiver em uma posição impar será multiplicado por 3 (b[L] <- a[L] * 3). Nas linguagens de programação mais utilizadas, atualmente, os vetores iniciam sempre na posição 0.

As *Matrizes*, assim como os vetores, são formadas por uma sequência de variáveis, “todas do mesmo tipo, com o mesmo indicador nome e alocadas sequencialmente na memória” (ASCENCIO; CAMPOS, 2012, p. 151). Matrizes, também denominadas pelos autores de variável composta homogênea multidimensional, necessitam de um índice para cada uma de suas dimensões. Vejamos um exemplo de matriz declarada com o nome de m, com três linhas e três colunas e que armazenara números inteiros (m[3][3]). O primeiro colchete representa a definição de quantas linhas terá a matriz e o segundo representa o número de colunas, M[linhas][colunas]. Cada índice dessa matriz poderá armazenar números inteiros (Figura 16).

Figura 16 - Exemplo de matriz

*Criar um algoritmo que entre com valores inteiros para uma matriz **m** 3 × 3 e imprima a matriz final, conforme mostrado a seguir:*

↪ *A matriz gira 180º.*

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \qquad \begin{bmatrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{bmatrix}$$

```
prog matriz31
 int L, c, m[3][3], final[3][3];
 para( L<- 0; L<=2; L++)
 {
   para( c<- 0; c<=2; c++)
   { imprima "\ndigite elemento: ",L + 1," - ",c + 1," : "; leia  m[L][c];  }
 }
 imprima "\nmatriz original\n";
 para( L<- 0; L<=2; L++)
 {
    para( c<- 0; c<=2; c++)
    { imprima m[L][c], "\t";  }
    imprima "\n";
 }
 imprima "\nmatriz gira 180 \n";
 para( L<- 2; L>= 0; L--)
 {
   para( c<- 2; c>=0; c--)
   { imprima m[L][c], "\t";}
   imprima "\n";
 }
 imprima "\n";
fimprog
```

Fonte: LOPES; GARCIA, 2002, p. 360.

As Matrizes, como também os vetores, iniciam seus índices com zero, assim, quando for referenciado m[0][0], estaremos apontando para o primeiro índice dessa matriz, pois está na linha “0” e na coluna “0”.

O último dos conceitos abordados na disciplina de Algoritmos são as *funções*, que são blocos de comandos que realizam tarefas específicas, as quais, geralmente, ficam separadas da rotina principal do algoritmo, muitas vezes, em arquivos separados. Ascencio e Campos salientam que “Como o problema pode ser subdividido em pequenas tarefas, os programas tendem a ficar menores e mais organizados” (2012, p. 252). As funções

abordam também parâmetros, ou seja, são dados passados do algoritmo principal para os subalgoritmos e os valores que esses enviam como retorno.

Todos esses conceitos abordados em Algoritmos são, muitas vezes, norteados pela sintaxe da linguagem de programação utilizada para codificar e, geralmente, baseadas na "escrita" das instruções a serem seguidas para a resolução de um problema. Portanto, apresentamos, a seguir, as ferramentas gráficas que se desviam do uso de escrita para a resolução de algoritmos e propõem o uso de objetos gráficos.

3.1.2 Ferramentas gráficas utilizadas no ensino de algoritmos

Ferramenta gráfica para o ensino de algoritmos é um programa que, ao invés de utilizar linhas de códigos, na forma de texto, quase sempre no idioma inglês, utiliza-se de objetos gráficos para montar o algoritmo. Como nosso primeiro contato com ferramentas utilizadas para ensino de Algoritmos foi com o *Scratch*, buscamos opiniões de outras pesquisas e revistas sobre o seu uso, que apresentavam um enfoque para ensino de nível superior. Além do *Scratch*, também procuramos conhecer outras ferramentas, buscando não limitar o estudo.

Aureliano e Tedesco (2013) apresentam um estudo que objetivou avaliar o potencial do *Scratch* no processo de ensino aprendizagem de programação para iniciantes na EaD a partir da coleta das opiniões dos alunos. Os resultados obtidos indicam que os alunos ficaram divididos em relação à contribuição do ambiente para o seu aprendizado. Esses afirmaram que o *Scratch* é voltado principalmente para crianças ou alunos bem iniciantes em programação.

Na revista *College & Research Libraries Ne*ws, Spina (2013) cita, como exemplos, três ferramentas visuais para aprendizes de programação de computadores, a primeira o *Scratch*, a segunda o Google *Blockly* e a terceira o MIT App *Inventor*. O autor afirma que essas ferramentas podem tornar o processo de ensinar conceitos básicos de programação mais acessível por meio da sua utilização, por apresentarem elementos de arrastar-e-soltar e, também, por terem abordagem visual. A seguir, serão descritas as principais características expostas pelo autor dessas três ferramentas visuais.

O *Scratch* foi criado pelo MIT no Lab's Lifelong Kindergarten, como uma ferramenta para crianças. Ao invés de solicitar que os usuários escrevam códigos, o *Scratch* faz uso de peças de um quebra-cabeça, que representam diferentes conceitos de programação de computadores, como *loops* e variáveis. Os usuários arrastam as peças para

criar jogos e projetos interativos. As peças do quebra-cabeça se encaixam apenas em combinações selecionadas para criar um programa, dessa forma, são ensinados, aos usuários, os conceitos básicos, centrais para a programação em qualquer linguagem. Apesar de ter sido, inicialmente, desenvolvido para crianças, pode ser usado por pessoas de todas as idades, como uma introdução para os conceitos básicos de programação de computadores.

A segunda ferramenta citada por Spina (2013) é o Google *Blockly*, descrito como um editor de programação gráfica, o qual faz uso de blocos semelhantes às peças encontradas no *Scratch* para criar programas. O que o diferencia do *Scratch* é o fato de que os programas que são criados com a utilização dessa ferramenta podem ser exportados para outras linguagens como JavaScript, Python, ou código XML, o que pode ajudar os usuários a fazer a conexão entre a interface gráfica e as outras linguagens de programação. Todo o código fonte é aberto, tornando possível, aos educadores, a utilização em seus próprios projetos.

Assim como Spina, Jordão (2012), também alega que o *Blockly* é um editor visual de programação semelhante ao disponibilizado pelo MIT, que facilita o aprendizado por usar a lógica de um quebra-cabeça, considerando que essa ferramenta é um bom começo para aprender a decompor um problema em etapas e ordená-las logicamente. Explica que seu funcionamento é semelhante ao *scratch*, é necessário comandar um boneco para que chegue ao seu destino, dando coordenadas em forma de algoritmo.

A terceira ferramenta citada por Spina (2013), O MIT *App Inventor* usa uma interface semelhante ao do Google *Blockly*, ele permite aos usuários criar aplicativos para dispositivos Android[1], sem escrever código sendo mais acessível para iniciantes.

Outro estudo que aborda *softwares* de apoio ao ensino de programação é apresentado por Valaski e Paraiso (2012), a pesquisa propõe a analisar o uso do *software* Alice na aprendizagem de conceitos básicos de programação para alunos do primeiro período do curso de Bacharelado em Sistemas de Informação. Os autores afirmam que os *softwares* para o ensino de programação, em sua maioria, têm como público alvo crianças, jovens e iniciantes na área. O fato de o *software* Alice estar em inglês gerou dificuldades para os alunos, também desinteresse pelo fato de que o propósito dele é muito distante de um desenvolvimento de um sistema comercial, pois a maioria dos alunos pretende entrar no mercado de trabalho. Por essa razão, eles se sentem mais motivados em aprender uma

[1] Android é um sistema operacional, ou seja, um sistema que gerencia todos os recursos dos dispositivos móveis, como celulares e tablets.

ferramenta que o mercado está utilizando do que uma ferramenta somente de caráter educacional.

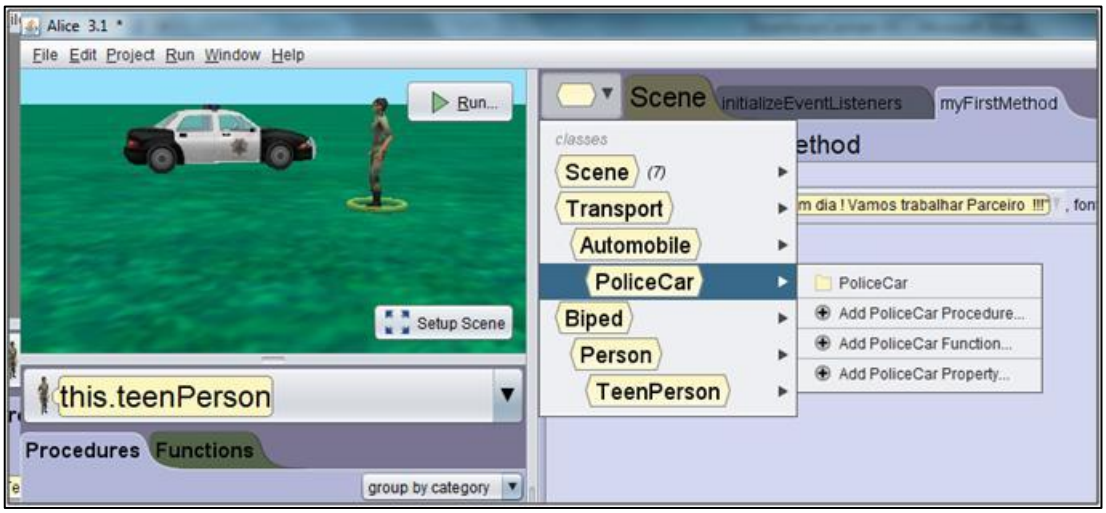
Chan (2013) apresenta algumas ferramentas gratuitas para aprender a programar, alegando que são plataformas intuitivas e interativas, perfeitas para aqueles que querem dar os primeiros passos no mundo da programação, dentre as citadas, novamente, vemos o nome de ferramentas já mencionadas por outros autores, como o *Scratch* e o *Blockly*.

Alice, *Scratch* e *Blockly* foram as ferramentas escolhidas para fazer uma avaliação para esta pesquisa e serão apresentadas, a seguir, suas principais características.

Alice tem ambiente de programação 3D, iniciado em 1999 na Carnegie Mellon University, o que torna mais fácil o desenvolvimento de animações, jogos interativos. É uma ferramenta de ensino projetada para auxiliar a introdução dos estudantes na programação orientada a objetos e possibilita que os alunos aprendam conceitos fundamentais de programação no contexto de criação de filmes animados e videogames simples (ALICE, 2008).

Alice não executa pela web, para esse software é preciso instalar o programa sendo que a versão mais recente é o Alice 3.1, com cerca de um 1 GB. Como o objetivo é programação orientada a objeto, a ferramenta dá ênfase maior para as imagens inseridas, e apresenta as propriedades de cada uma em forma de Procedures, isso é, identifica a ação a ser executada por cada imagem. No exemplo criado (Figura 17), quando clicarmos em um dos objetos *Person* (imagem pessoa) que está inserido na *Scene* (cena) é possível adicionarmos uma *procedure*, uma *function* ou uma *proprierty*. Nessas *procedures* ou *functions*, poderão ser inseridos os comandos para que façam os objetos da cena se moverem ou falarem. Os conceitos apresentados como métodos, que se referenciam ao objeto, exemplo *this.teenPerson* (Figura 17), são conceitos importantes de orientação a objetos e que somente serão estudados pelos alunos do curso do TSPI no terceiro semestre do na disciplina de Programação Orientada a Objetos.

Figura 17 - Inserindo objetos no Alice



Fonte: da pesquisa.

Essas cenas são programadas segundo metodologia de classes e objetos, o que dificulta sua manipulação, principalmente por alunos iniciantes em algoritmos. Todas as imagens inseridas, chamadas pelo Alice de objetos, irão interagir na cena, formando uma história conforme programado (Figura 18).

Figura 18 - Executando a cena no Alice

Fonte: da pesquisa.

Todos os elementos do *software* estão organizados com o intuito de introduzir, naturalmente, para o aluno os conceitos básicos de orientação a objetos. Portanto, como

nesse nível do curso estes conteúdos não são apresentados, não seria a opção mais indicada a utilização do Alice com o objetivo de auxiliar em Algoritmos.

A segunda ferramenta escolhida para avaliação foi o *Scratch*, esse é um programa livre desenvolvido no MIT (*Massachusettes Institute of Tecnology*) em 2007. Constitui-se de uma linguagem de programação visual que permite ao usuário construir, interativamente, seus próprios programas, animações, histórias, jogos e ambientes visuais de aprendizagem. Em suas versões mais recentes, o programa pode ser utilizado diretamente na internet, sem precisar instalar.

Os comandos são representados por blocos que podem ser arrastados para a área de *script* (programação) e conectados, com uma interface intuitiva, os blocos têm os encaixes característicos do seu tipo e cada bloco de código está sempre associado a um objeto. Como ilustrado (Figura 19) no programa criado como exemplo.

Figura 19 - Encaixes característicos dos blocos do *Scratch*

Fonte: da pesquisa.

Os comandos de um grupo apresentam a mesma cor, facilitando a sua localização, da seguinte maneira: no grupo Movimento (em azul escuro), estão os comandos relacionados com a animação do objeto, como: gire 90 graus, ilustrado no programa criado como exemplo (Figura 19). O grupo Controle (em amarelo) tem os blocos de controle condicional, com os quais é possível fazer o programa executar diferentes comandos, dependendo da condição estabelecida, com os laços de repetição, o *Scratch* pode ser

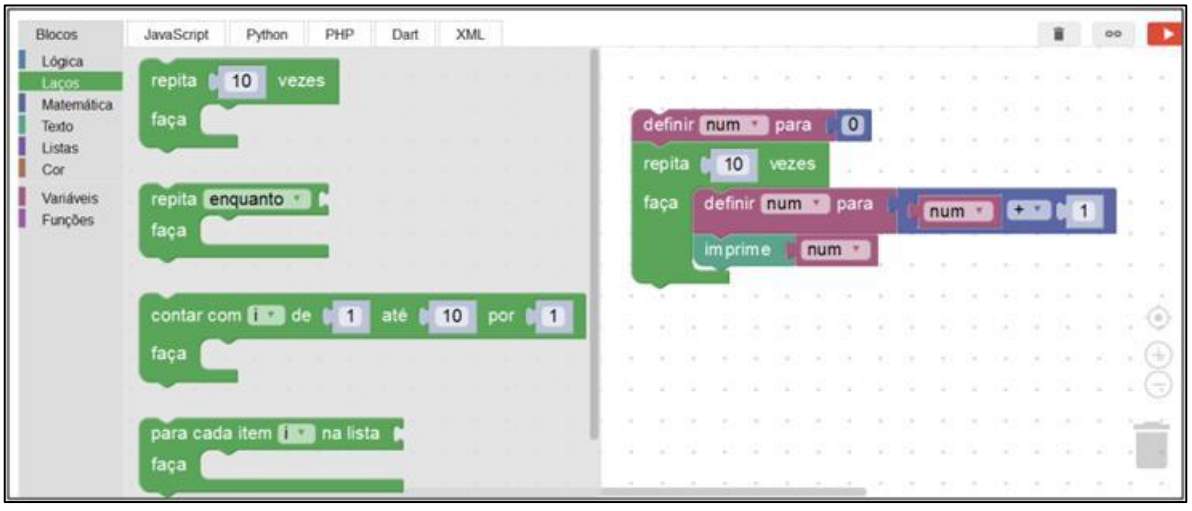
orientado a repetir “n” vezes, assim, não é necessário colocar “manualmente” os “n” comandos.

A terceira e última ferramenta escolhida para avaliação foi o *Blockly*, um editor de programação visual para programadores, desenvolvido pelo Google, que descreve seu próprio aplicativo como parte de um número crescente de ambientes de programação visual, sendo que muitos desses têm raízes no MIT e apresentam aparência semelhante. *Blockly* foi influenciado pela App Inventor, que por sua vez foi influenciado pelo *Scratch*, que, por sua vez, foi influenciado por StarLogo (GOOGLE DEVELPERS, 2015).

Segundo Jordão (2012), o *Blockly* facilita muito o aprendizado, faz uso de blocos que são semelhantes às peças encontradas no *Scratch*. Segundo a Google Developers (2015), outra característica importante é que o *Blockly* em si não é uma aplicação educativa, é um editor que pode ser usado como parte de uma ferramenta. Os autores afirmam que, atualmente, está sendo usado como um editor visual por centenas de projetos na sua maioria de natureza educacional.

O *Blockly* organiza os comandos por grupos e esses têm uma cor característica para facilitar sua localização, assim como no *Scratch*. Apresentamos, a seguir, (Figura 20) um algoritmo criado para exemplo, construído utilizando o *Blockly*.

Figura 20 - Algoritmo construído com *Blockly*

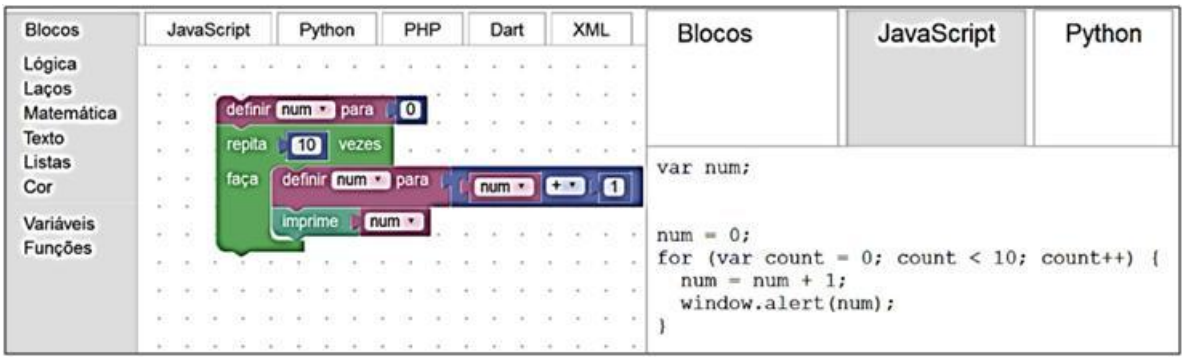


Fonte: da pesquisa.

Além dos blocos para a construção dos Algoritmos, o *Blockly* tem como saída códigos em uma linguagem de formato textual, ou seja, é capaz de gerar em linguagens como Javascript, Python, PHP, Dart e XML, com base nos blocos do algoritmo construído, como ilustrado na Figura 21. O comando inserido na guia Blocos, automaticamente,

converterá e atualizará as outras guias das cinco linguagens de programação. Entretanto, não é indicado para ser utilizado para programação em larga escala, uma vez que é projetado para criar pequenos programas.

Figura 21 - Trecho de Algoritmo visualizado na linguagem JavaScript



Fonte: da pesquisa.

Por meio de estudos sobre a linguagem *Blockly*, localizamos vários exemplos de projetos que a utilizam com fins educativos e com o objetivo de instigar nos alunos a vontade de programar. Segundo a revista Olhardigital (2013), um dos projetos com esse fim é o *code.org*, construído com o objetivo de difundir o ensino de programação. O principal objetivo dessa ferramenta é apresentar conceitos de algoritmos na forma de desafios que devem ser resolvidos, com o estilo de programação de arrastar e soltar.

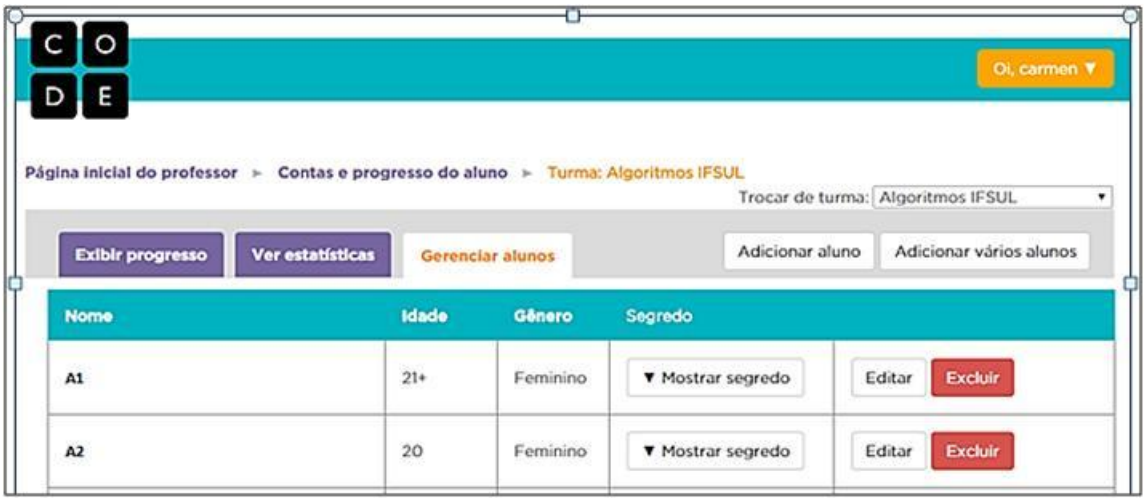
> Lançado em 2013, Code.org é uma organização sem fins lucrativos, dedicada a ampliar a participação em ciência da computação, tornando-a disponível em mais escolas, e aumentar a participação de mulheres e estudantes sub-representado de cor. Nossa visão é que todos os alunos em todas as escolas devem ter a oportunidade de aprender ciência da computação. Nós acreditamos que a ciência da computação e programação de computadores deve ser parte do currículo na educação, ao lado de outra ciência, tecnologia, engenharia e matemática (STEM), tais como biologia, física, química e álgebra (CODE.ORG, 2015, tradução nossa).

O site da Code.org apresenta algumas opções de ferramentas construídas com o *Blockly* e a escolha por uma delas a para a pesquisa deu-se pelo motivo de essa apresentar algumas caraterísticas semelhantes com as linguagens de programação, que serão vistas no curso de TSPI nos próximos semestres. Dentre as opções da ferramenta desenvolvida com o *Blockly* a *Computer Science Fundamentals*, , segundo seus criadores, foi criada com o objetivo de ensinar Algoritmos e os passos das atividades foram planejados com base nos nos seus principais conceitos.

Como Valaski e Paraiso (2012), percebemos que os alunos na sua maioria pretendem entrar rapidamente no mercado de trabalho, por essa razão, concordamos com os autores quando dizem que eles se sentiriam mais motivados em aprender uma ferramenta semelhante com as utilizadas na programação formal. Acreditamos que pelo fato de essa ferramenta *Computer Science Fundamentals* do Code.org ter sido desenvolvida com o *Blockly*, utilizando-se do seu padrão e recursos, os alunos estariam aptos a utilizá-la para construir suas próprias linhas de códigos, com a possibilidade de converter para linguagem textual e inserir em seus futuros programas.

A ferramenta da Code.org também oferece a opção de cadastrar e gerenciar turmas e, assim, possibilita o acompanhamento e a evolução de forma individualizada, opção que se mostra adequada para a pesquisa, pois torna possível a análise das atividades dos alunos (Figura 22).

Figura 22 - Opção para gerenciar e acompanhar a turma do Code.org



Fonte: da pesquisa.

Com o uso dessa ferramenta, é possível visualizar a forma com que cada aluno resolveu cada um dos passos dos exercícios propostos, o que possibilita uma melhor identificação das dificuldades individuais, no processo de construção de algoritmos.

3.2 Em busca de um referencial teórico para o ensino de algoritmos

No referencial teórico, apresentamos reflexões sobre algumas noções da didática da matemática, assim como da teoria das situações didáticas e alguns de seus principais conceitos. Abordamos a resolução de problemas por meio de uma abordagem cognitiva, o

desenvolvimento das representações mentais como a teoria dos registros e representações e semióticas de Duval. Expomos contribuições da psicologia sobre o desenvolvimento do pensamento por intermédio do conceito das funções psicológicas superiores de Vygotsky.

3.2.1 Didática da Matemática

A Didática da Matemática, como teorização, iniciou na década de 1970, na França, com intuito de sistematizar os estudos acerca do ensino da matemática. Alguns dos principais autores foram de Guy Brousseau em 1976 e Regine Douady em 1984. Trazemos o conceito que Brousseau (1986) atribui para a Didática da Matemática apresentados por Passos e Teixeira.

> A Didática da Matemática estuda atividades didáticas que têm como objetivo o ensino da parte específica dos saberes matemáticos, propiciando explicações, conceitos e teorias, assim como meios de previsão e análise; incorporando resultados relativos aos comportamentos cognitivos dos alunos, além dos tipos de situações utilizadas e os fenômenos de comunicação do saber (2013, p. 157).

Os autores complementam, conceituando Didática da Matemática como "a arte de conceber e conduzir condições que podem determinar a aprendizagem de um saber matemático por parte de um sujeito" (2013, p. 157). Eles apresentam a definição de Didática exposta por Brousseau (1986) como uma relação específica entre os conteúdos de ensino, a maneira como os alunos adquirem conhecimentos e os métodos. Ainda asseveram que Brousseau desenvolveu uma teoria para compreender as relações que acontecem entre os alunos, o professor e o saber em sala de aula. Tal teoria é conhecida como Teoria das Situações Didáticas, em que alunos e professores são atores indispensáveis da relação de ensino e aprendizagem, bem como o meio em que a situação didática se faz presente.

Brousseau (2008, p. 21) define uma "situação como um modelo de interação de um sujeito com um meio determinado". O autor ainda menciona que o recurso que esse sujeito dispõe para alcançar um estado favorável nesse meio é um leque de decisões que dependem do emprego de um conhecimento específico.

A Teoria das Situações Didáticas por meio da abordagem desenvolvida por Brousseau é conceituada também por Freitas (2008) como as formas de apresentar o conteúdo matemático aos alunos, bem como compreender o fenômeno da aprendizagem matemática. Para o autor, essa teoria constitui um referencial para a educação matemática, pois valoriza os conhecimentos mobilizados pelo aluno e seu envolvimento, bem como o

trabalho do professor, que deve criar condições suficientes para que o aluno se aproprie dos conteúdos.

O conceito de Teoria das Situações Didáticas apresentado por Almouloud (2007) descreve que essa teoria busca criar um modelo de interação entre o aprendiz, o saber e o meio em que o aprendizado deve se desenrolar. O autor menciona que o objeto central de estudo nessa teoria não é o sujeito cognitivo, mas a situação didática na qual são identificadas as interações estabelecidas.

O objeto central de estudo dessa teoria de Brousseau não é o sujeito cognitivo, e, sim, as situações didáticas é igualmente salientado por Passos e Teixeira (2013). Os autores apresentam, ainda, a importância dos erros ao declarar: "Algum erro cometido pelo aluno, nessa teoria, quando identificado, constitui-se como valiosa fonte de informação para a elaboração de boas questões ou para novas situações problemas que possam atender, mais claramente, os objetivos desejáveis" (p. 158).

Cury (2007) expõe a análise de erros apresentada por Brousseau, que assim define o erro:

> [...] não é somente o efeito da ignorância, da incerteza, do acaso, como se acredita nas teorias empiristas ou behavioristas da aprendizagem, mas o efeito de um conhecimento anterior, que tinha seu interesse, seu sucesso, mas que agora se revela falso, ou simplesmente inadaptado. Os erros desse tipo não são instáveis e imprevisíveis, eles são construídos sem obstáculos (BROUSSEAU, 1983 apud CURY, 2007, p. 33).

Brousseau (2008) destaca que os obstáculos se manifestam pelos erros e esses não desaparecem com a aprendizagem de um novo conceito, mas oferecem resistência à compreensão do novo, retardando sua aplicação, sendo, portanto inútil ignorar um obstáculo. Almouloud (2007), também com base em Brousseau, aponta que o estudo dos obstáculos passa pelo estudo de erros resistentes dos alunos. Grando (1995) expõe que alguns obstáculos devem ser evitados e outros não, de forma que "o importante é que o professor e o aluno estejam alertas para tomar consciência deles no processo de ensino-aprendizagem; para evitá-los ou para mudar de concepção quando a situação exigir" (p. 121).

Um dos grandes desafios é conseguir envolver todos os alunos, fazer com que se comprometam com o seu aprendizado e aceitem o desafio de resolver, aparentemente sozinhos, os problemas. Sobre esse aspecto, Brousseau (2008) faz a seguinte observação:

> Tais problemas, escolhidos de modo que o estudante os possa aceitar, devem fazer pela própria dinâmica, com que o aluno atue, fale, reflita e evolua. Do momento em que o aluno aceita o problema como seu até aquele em que se produza a resposta, o professor se recusa a intervir como fornecedor dos conhecimentos que quer ver seguir. O aluno sabe que o problema foi escolhido para fazer com que ele adquira um conhecimento novo, mas precisa saber, também, que este conhecimento é inteiramente justificado pela lógica interna da situação e que pode prescindir das razões didáticas para construí-lo (p. 35).

Situações sem nenhuma indicação intencional, que o autor denomina de adidáticas, são também as que o aluno de imediato não consegue resolver, sendo assim, o professor apresenta, inicialmente, aquelas que o aluno consegue solucionar. Essa situação ou problema escolhido pelo professor em um jogo com o sistema de interação entre o aluno e o seu meio constituem o que o autor denomina de situação didática.

Porém a aprendizagem apresenta rupturas e algumas das concepções já adquiridas não desaparecem em benefício de outras melhores, elas resistem provocando erros, tornando-se, então, como afirma o autor, obstáculo epistemológico. O conceito de obstáculo epistemológico surgiu com Bachelard, mas Brousseau propôs uma adaptação para a didática da matemática.

> - Um obstáculo é um "conhecimento" no sentido que lhe demos de "forma regular de considerar um conjunto de situações".
> - Tal conhecimento dá resultados corretos ou vantagens observáveis em um determinado contexto, mas revela-se falso ou totalmente inadequado em um contexto novo ou mais amplo.
> - O conhecimento novo, verdadeiro ou válido sobre um contexto mais amplo não é determinado "de acordo com" o conhecimento anterior, mas em oposição a ele: utilizam outros pontos de vista, outros métodos etc. Entre eles não existem relações "lógicas" evidentes que permitam desacreditar facilmente o erro antigo por meio do conhecimento novo. Ao contrário, a competição entre eles acontece no primeiro contexto.
> - Os conhecimentos aqui considerados não são construções pessoais variáveis, mas, sim respostas "universais" em contextos precisos. Portanto, surgem quase necessariamente na origem de um saber, seja ela histórica ou didática (BROUSSEAU, 2008, p. 49).

Ainda com base na teoria das situações didáticas, podemos refletir um pouco mais sobre as ações e questões que surgiram pela prática do dia a dia em sala de aula. Percebemos que, na angústia de ajudar os alunos a entender os conteúdos propostos, acabamos por "ajudá-los demais" e por privá-los de produzir conhecimento.

Após alguns anos trabalhando com a mesma disciplina de Algoritmos, observamos que é importante para o aluno a ajuda de colegas e do professor, mas no momento de introdução dos novos conceitos, depois é importante que ele se torne independente dessa

ajuda. Notamos que quanto mais os alunos "trabalham sozinhos" e se desafiam a resolver os exercícios sem pedir muita ajuda, melhor é o seu desempenho com relação aos conteúdos. Essa reflexão nos reporta aos conceitos de situação adidática introduzida por Brousseau e que Freitas caracteriza essencialmente por "representar determinados momentos do processo de aprendizagem nos quais os alunos trabalham de maneira independente, não sofrendo nenhum tipo de controle direto do professor relativamente ao conteúdo matemático em jogo" (FREITAS, 2008, p. 44).

Sobre situações adidáticas, o autor apresenta, ainda, que essas representam os momentos mais importantes da aprendizagem, pois o sucesso do aluno significa que ele, por seu próprio mérito, conseguiu sintetizar algum conhecimento. Outra noção importante originada da Teoria das Situações Didáticas é o contrato didático, que analisa as relações que se estabelecem entre professores e alunos, mesmo que de forma implícita e, também, a influência dessa no ensino da matemática. Sobre a relação professor e aluno, Grando et al. (1996) salienta que essa é um tipo especial de relação, sempre mediada pelo saber, é formalmente elaborada com a finalidade de possibilitar o alcance desse conhecimento.

Sobre contrato didático, Almouloud (2007), baseado nas afirmações feitas por Brousseau, destaca três observações que considera importantes:

1) Um contrato didático é específico dos conhecimentos em jogo, sempre pode ser mudado, tendo em vista que os conhecimentos e os saberes mudam. Diferente do chamado contrato pedagógico que privilegia as relações sociais, as atitudes, as regras e as convenções, mas coloca em jogo o saber.

2) O contrato didático tem, fundamentalmente, por objetivo a aquisição dos saberes pelos alunos; seu funcionamento depende de diferentes contextos de ensino e aprendizagem. Fazem parte dos determinantes essenciais do contrato didático, as escolhas pedagógicas, o tipo de trabalho proposta ao aluno, as condições de avaliação.

3) Um contrato didático mal administrado pelos envolvidos (professor e alunos), pode ser fonte de dificuldades de aprendizagem de novos conceitos matemáticos e vir a ser motivo de renegociação.

Almouloud (2007) também apresenta o conceito de ruptura de contrato didático, que corresponde à sua renegociação e pode provocar a entrada em cena de fatores positivos ou negativos para a aprendizagem. Geralmente, essa ruptura é causada pelo avanço no processo de conhecimento. As atitudes ou práticas consideradas como rupturas do contrato didático são avaliadas como efeitos do contrato didático, sendo rotuladas pelo autor de:

efeito pigmaleão, efeito Topaze, efeito Jordam, deslize metacognitivo e uso abusivo de analogia.

Na sala de aula, podemos observar, praticamente, todos esses efeitos, o Pigmaleão ilustra o que os psicólogos chamam de fenômeno das expectativas. Esse está relacionado com a imagem que o professor tem da turma, ou de um aluno em particular, e faz com que, em alguns momentos, limite seu nível de exigência em função dessa projeção.

O efeito Topaze é percebido quando o professor, ao perceber que seu aluno encontra dificuldades, cria condições para que esse as supere, sem um real engajamento desse aluno. O professor acaba por fazer o trabalho que, na realidade, é de responsabilidade do aluno e acaba escolhendo questões que provoquem as respostas esperadas.

O efeito Jordam se caracteriza pelo fato de o professor interpretar como um saber científico o que tem significação trivial de senso comum. O deslize metacognitivo ocorre quando o professor considera útil uma técnica para resolver um problema como objeto de estudo e perde de vista o real saber a desenvolver, ou, ainda, quando utiliza suas próprias palavras e crenças como objeto de estudo, ao invés, do adequado conhecimento matemático.

A última das práticas avaliadas pelo autor como efeitos do contrato didático é o uso exagerado da analogia, útil para fazer compreender o significado de um conceito, mas quando utilizado de maneira abusiva pode descaracterizar o conceito. Assim, o professor por se encontrar numa situação de cobranças e, muitas vezes, por falta de uma capacitação adequada, faz uso desses efeitos conforme expõe Almouloud (2007):

> Analisando esses efeitos, percebe-se que o professor se encontra muitas vezes numa situação difícil, pode-se dizer que se encontra num paradoxo: ele deve criar condições para a aprendizagem dos alunos, mas quase tudo que ele faz para conseguir uma resposta satisfatória pode estar prejudicando a aprendizagem, por não permitir que os alunos cheguem sozinhos à resposta esperada. O aluno também fica numa posição paradoxal, pois não constrói, por conta própria, o saber que o professor quer lhe ensinar (p. 96).

Entretanto, para Papert (1994), promover situações em que o aluno assuma o comando é desafiador, pois a partir dessas podem surgir outras circunstâncias inesperadas e o professor, nas novas situações, também passa a exercer o papel de aprendiz. O professor assume, desse modo, que não é o único conhecedor e passa a dividir a responsabilidade da aprendizagem com seus alunos. Essa divisão estimula o aluno a

desenvolver a capacidade de pesquisar e cooperar com o colega, sem perder a disposição de desafiar-se a resolver os exercícios. O autor relata um fato ocorrido com Joe, um professor de 5ª série, conforme se pode verificar na citação a seguir.

> Desde que os computadores surgiram, comecei a temer o dia em que meus alunos saberiam mais sobre programação do que eu jamais saberia [...] [um dia] percebi que os estudantes tinham problemas que eu não consegui nem mesmo entender, quanto mais resolver, lutei para enfrentar o fato de que eu não poderia manter minha posição de saber mais do que sabia. Eu estava com medo de que desistir destruiria minha autoridade como professor. A situação, no entanto, piorou. Por fim, sucumbi e disse que não entendera o problema – "vão e discutam-no com alguns colegas da classe que poderiam ajudar" [...]. E ocorreu que juntas as crianças encontraram uma solução. Agora, a coisa espantosa que eu temia terminou sendo uma liberação. Eu não tinha mais medo de ficar exposto [...]. Senti que não podia mais fingir saber tudo sobre outras matérias também. Que alívio! Isso mudou meu relacionamento com as crianças e comigo mesmo. Minha classe tornou-se mais uma comunidade colaborativa onde estávamos todos aprendendo juntos (PAPERT, 1994, p. 63).

Nesse sentido, Papadopoulos (2005) sugere que é preciso desenvolver as motivações dos alunos, se a intenção é aumentar sua propensão a aprender, tanto na escola, quanto, mais tarde, fora dessa. Em muitos casos, esse processo é freado, por um sistema concebido para ressaltar o fracasso, no sentido escolar do termo, e não para estimular o potencial de êxito de cada indivíduo.

É preciso envolver o aluno em atividades participativas, que estimulem seu raciocínio, em consonância com uma prática formativa e não meramente armazenadora de informações e, assim, modificar nossos hábitos e rotinas que um dia nos foram ensinados. Seguindo na mesma abordagem, sobre a importância de um "sentido" dos conteúdos para os alunos, Papert (1994), idealizador do paradigma construcionista, aponta que a ênfase no processo de aprendizagem em que o aluno entra em contato direto com o concreto, qualifica-o significativamente. Nesse contexto, segundo o autor, o estudante pode manipular, errar e superar os erros, por meio da interação com os objetos em uso. Além disso, esse processo também pode qualificar a interação entre colegas e a interação do aluno com o professor, estabelecendo um ambiente favorável para a aprendizagem, de modo que competiria ao educador promover situações que permitam ao aluno construir sua aprendizagem, sem perder de vista a cientificidade dos conhecimentos propostos.

Sobre esses conhecimentos, existe certa preocupação referente às diversas transformações ou adaptações que "um conhecimento científico" sofre desde a sua elaboração até a socialização e a apropriação desse pelos alunos. Tais transformações

constituem o processo denominado de transposição didática, assim contextualizada por Chevallard (2000):

> Um conteúdo escolar que tenha sido designado como saber a ensinar, sofre desde então um conjunto de transformações adaptativas que irão torná-lo apto para ocupar lugar entre os objetos de ensino. O trabalho que transforma um objeto de saber a ensinar em um objeto de ensino é denominado de transposição didática (p. 65, tradução nossa).

Para Grando (2000), as transformações desse conhecimento podem ocorrer em momentos diferentes e ocorrem na produção e na publicação, na seleção do que será levado para a escola, naquilo que é, efetivamente, ensinado e no que é, efetivamente, internalizado pelo aluno. Com base na obra de Chevallard, a autora apresenta uma reflexão sobre até que ponto os professores podem adaptar esses conhecimentos, sendo que não podem ficar tão afastados do saber científico a ponto de torná-lo banalizado. Essa reflexão traz à tona a permanente vigilância que o professor precisa desenvolver para que, na angústia de ver as dificuldades de seus alunos solucionadas, não acabe mudando o conhecimento.

A transposição didática, para Pais (2008), pode ser analisada com base nos três tipos de saberes, o saber científico, o saber a ensinar e o saber ensinado. O autor descreve o saber científico como um saber que, normalmente, é desenvolvido nas universidades ou institutos de pesquisas, dependendo em parte do financiamento de pesquisas. O saber a ensinar é descrito pelo autor como um saber ligado a uma forma didática que serve para ser apresentado ao aluno. Sobre saber ensinado, Pais expõe que "Na passagem do saber científico ao saber a ser ensinado ocorre a criação de um verdadeiro modelo teórico que ultrapassa os próprios limites do saber matemático" (2008, p. 24). Assim, Pais traz as seguintes reflexões: o saber ensinado coincide com o planejado ao nível do saber a ensinar? Como ficam as possíveis perdas, muitas vezes, apresentadas no caso do saber científico? Nas palavras do autor:

> A transposição didática é um modelo teórico que possibilita uma leitura dessa possível perda do significado do saber, buscando compreender questões contextuais que surgem nas relações criadas entre as instituições envolvidas. A análise da transposição didática envolve, além das noções matemáticas, noções que, mesmo sendo necessárias à aprendizagem, geralmente não são ensinadas (PAIS, 2008, p. 45).

As noções matemáticas necessárias e que, geralmente, não são ensinadas, são chamadas de paramatemáticas e protomatemáticas. Percebemos, com base na nossa experiência de docência, que os alunos chegam à universidade e ainda não compreendem certas noções matemáticas as quais são consideradas noções que já deveriam ter sido vistas ainda no ensino fundamental. Chevallard define as noções matemáticas como objetos de saber, e os referencia como objetos possíveis de serem ensinados. O autor assevera: "O que é um objeto de saber? Para um professor de matemática, certamente incluirá nessa categoria as noções matemáticas: por exemplo, a adição, o círculo, a derivação, as equações" (1991, p. 57, tradução nossa). Nesse caso, fazem parte dessa categoria todos os objetos de saber que o professor seleciona para suas aulas.

Para que a aprendizagem dessas noções matemáticas aconteça, são necessários outros objetos de saber denominados por Chevallard (2000, p. 58) de "noções paramatemáticas", as quais são consideradas "noções ferramentas" para a "atividade matemática". Para o autor, somente os objetos de saber são objetos ensinados, já as noções paramatemáticas devem ser apreendidas, mas, dificilmente, são ensinadas. As noções protomatemáticas são noções mobilizadas implicitamente pelo contrato didático, ou, ainda, conforme Chevallard, "Uma dificuldade desse tipo pode surgir da falta de domínio de uma capacidade requerida pelo contrato didático para seu entendimento" (2000, p. 57, tradução nossa).

Chevallard relaciona e exemplifica os três tipos de noções:

> Noções matemáticas, noções paramatemáticas, noções protomatemáticas constituem extratos cada vez mais profundos do funcionamento didático do saber. *Sua consideração diferencial é necessária para a análise didática*; por isso a análise da transposição didática de qualquer noção matemática (por exemplo, a identidade $a^2 – b^2 = (a + b) (a – b)$) supõe a consideração de noções paramatemáticas (por exemplo, as noções de *fatoração e de simplificação*), e que por sua vez devem ser consideradas a luz de certas noções protomatemáticas (a noção de "padrão", de "simplicidade", etc.) (CHEVALLARD, 2000, p. 65, grifos do autor, tradução nossa).

O autor aponta para a possibilidade de, às vezes, elevar-se uma noção de um nível dado para um nível superior, assim, noções paramatemáticas podem ser objetos de definições precisas em lógica matemática, como qualquer noção protomatemática pode tornar-se uma noção paramatemática.

Ainda sobre transposição didática, Pais (2008) aponta elementos importantes que falam do processo da preparação prévia pelo qual passa o conteúdo a ser ensinado, ao que

chama de “textualização do saber” (p. 32). O autor ainda destaca duas variáveis que considera fundamentais quando se trata de programação de ensino, quais sejam: o tempo didático e o tempo de aprendizagem. Tempo didático “aquele marcado nos programas escolares e nos livros didáticos em cumprimento a uma exigência legal” (p. 33); e tempo de aprendizagem que “está mais vinculado com as rupturas e os conflitos do conhecimento, exigindo uma permanente reorganização de informações e que caracteriza toda a complexidade do ato de aprender” (p. 34).

O tempo didático admite que seja possível enquadrar o conteúdo num determinado espaço de tempo, preocupando-se mais em cumprir o programa proposto do que com a aprendizagem. Por sua vez, o tempo de aprendizagem respeita o tempo de cada sujeito, aquele necessário ao sujeito para superar seus bloqueios. Na concepção de Pais (2008) “a superação da distância entre o tempo de aprendizagem e o didático passa por uma retomada constante das noções já estudadas, nas mais variadas situações, sempre buscando novos níveis de formalização das atividades” (p. 35). O autor menciona que para compreender melhor o entrelaçamento entre esses dois tempos “é necessário voltar à outra especificidade do ensino da matemática, que é a resolução de problemas” (p. 35). Para ele, o problema impulsiona o saber matemático, assim como Pais (2008), Freitas também afirma que “o problema se constitui num verdadeiro eixo condutor de toda aprendizagem da matemática” (2008, p. 89).

3.2.2 Resolução de Problemas

Os conceitos sobre resolução de problemas são semelhantes aos conceitos da construção de algoritmos, pois, ao desenvolver um algoritmo, os alunos estão apresentando uma solução para determinado problema.

Resolução de problemas também propõe considerar uma série de situações em que os alunos, aplicando seus conhecimentos prévios, apresentem uma solução, sendo que o resultado não é mais importante do que a própria resolução. E essa resolução é um processo construído de forma individual, assim como menciona Freitas:

> Chega um momento em que cada um deve dar o seu próprio passo. O aluno deve, assim, ser permanentemente motivado a engajar-se nessa linha de raciocínio, por seu próprio mérito, ao longo de todo o processo de ensino. [...] devemos possibilitar ao aluno o máximo de independência para que ele possa desenvolver automaticamente seus próprios mecanismos de resolução do problema (2008, p. 89-91).

Essa capacidade de resolver problemas não é uma atividade que será útil somente para o aluno dentro do contexto escolar, mas, sim, para solução de problemas de ordem prática do dia a dia, da vida. Echeverría e Pozo sintetizam a proposta com base na resolução de problema, citando que:

> Ensinar a resolver problemas não consiste somente em dotar os alunos de habilidades e estratégias eficazes, mas também em criar neles o hábito e atitude de enfrentar a aprendizagem como um problema para o qual deve ser encontrada uma resposta. Não é uma questão de somente ensinar a resolver problemas, mas também de ensinar a propor problemas para si mesmo, a transformar a realidade em um problema que mereça ser questionado e estudado (1998, p. 14-15).

Ainda sobre resolução de problemas, citamos Polya (1995) que apresenta quatro fases como sugestão de como resolver um problema: compreender o problema, verificar como os dados estão inter-relacionados para estabelecer um plano, executar o plano e realizar o retrospecto da resolução.

Sobre a primeira das fases, a compreensão do problema, o autor assim descreve sua importância:

> É uma tolice responder a uma pergunta que não tenha sido compreendida. É triste trabalhar para um fim que não se deseja. Estas coisas tolas e tristes fazem-se muitas vezes, mas cabe ao professor evitar que elas ocorram O aluno precisa compreender o problema, mas não só isto: deve também desejar resolvê-lo. Se lhe faltar compreensão e interesse, isto nem sempre será culpa sua. O problema deve ser bem escolhido, nem muito difícil nem muito fácil, natural e interessante, e certo tempo deve ser dedicado à sua apresentação natural e interessante (POLYA, 1995, p. 4).

Na visão desse autor, a incompreensão do problema faz com que “alguns alunos atirem-se ao cálculo e ao desenho sem qualquer plano ou idéia geral; outros esperam desajeitadamente que surja alguma idéia e nada fazem para expressar sua aparição” (1995, p. 57).

A segunda das fases, a elaboração do plano, consiste em relacionar os dados do problema à pergunta feita e procurar achar uma estratégia para que se possa chegar à solução. Nessa fase, o autor sugere que o trabalho inicie com a indagação referente a conhecer algum problema correspondente. Como é comum encontrarmos problemas, há mais situações correlacionadas. Nesse momento, o autor sugere que nos preocupemos em pensar num problema conhecido que tenha a mesma incógnita ou, ao menos uma que seja

semelhante. Reforça, ainda, a importância de o professor auxiliar o aluno por meio de discretas sugestões e indagações que o conduzam ao caminho certo.

Quanto à terceira fase, a execução do plano, é o momento em que o roteiro estabelecido deve ser executado. O autor argumenta que, nessa fase, é fundamental a paciência para examinar todos os passos até que tudo fique claro, sem que haja nada obscuro no que se possa ocultar um erro.

A quarta e última fase sobre resolução de problemas é o retrospecto da resolução. Essa fase do processo pode ser a mais proveitosa, pois, reexaminar o caminho que levou ao resultado pode consolidar o conhecimento, aperfeiçoando a capacidade de resolver problemas. Para o autor, "um bom professor precisa compreender e transmitir a seus alunos o conceito de que problema algum fica completamente esgotado. Resta sempre alguma coisa a fazer" (POLYA, 1995, p.10). Fica evidente que sempre é possível aperfeiçoar nossa compreensão da resolução proposta.

3.2.3 Teoria dos Registros de Representações Semióticas

Para melhor avaliar as dificuldades na aprendizagem da disciplina de Algoritmos, buscamos também subsídios em Duval (2003) que procura compreender e avaliar as dificuldades dos alunos na compreensão da matemática, bem como a natureza dessas. O autor propõe uma abordagem cognitiva, com o objetivo de contribuir para o desenvolvimento geral da capacidade de raciocínio dos alunos. Segundo Duval (2003),

> A diferença entre a atividade cognitiva requerida pela matemática e aquela requerida em outros domínios do conhecimento não deve ser procurada nos conceitos – pois não há domínio de conhecimento que não desenvolva um contingente de conceitos mais ou menos complexo - mas nas duas características seguintes:
> 1. A importância das representações semióticas e
> 2. A grande variedade de representações semióticas utilizadas em matemática (p. 12).

Assim, com base nas afirmações do autor, deduzimos que as dificuldades de compreensão na aprendizagem da matemática não estão relacionadas aos conceitos, mas às muitas representações semióticas utilizadas e à confusão no uso que se faz dessas. Em suas palavras, "A distinção entre o objeto e sua representação é, portanto, um ponto estratégico para a compreensão da matemática" (DUVAL, 2012, p. 269).

Duval avalia que, por um lado, se anuncia a importância da linguagem (natural) na atividade matemática, por outro, é privilegiado o uso de símbolos e de representações geométricas e gráficas. Assim, relaciona as dificuldades de compreensão na aprendizagem matemática não aos conceitos, mas à variedade de representações semióticas utilizadas e ao uso "confuso" que se faz dessas. Afirma que "para compreender bem o impacto das especificidades da matemática em relação aos processos de compreensão na aprendizagem, é preciso considerar as duas faces da atividade matemática" (DUVAL, 2013, p. 17). Tais faces são denominadas exposta e oculta. A face exposta corresponde aos objetos matemáticos como números, funções, equações, polígonos, às suas propriedades, às fórmulas, às demonstrações; a face oculta corresponde aos gestos intelectuais que constituem o caráter cognitivo e epistemológico específicos da matemática. Essa última se manifesta por meio de bloqueios ou erros recorrentes quando do não reconhecimento de um mesmo objeto em duas escritas diferentes, ou em representações semióticas produzidas em dois registros diferentes, que é um sintoma frequente e que passa despercebido, ou, ainda, é considerado uma incompreensão do conceito a ser utilizado. Por fim, Duval (2003) afirma que:

> A teoria dos registros de representação semiótica diz respeito à face oculta da atividade matemática. Ela visa à modelagem do funcionamento semicognitivo que está subjacente ao pensamento matemático. Sem o desenvolvimento deste não podemos compreender e nem conduzir uma atividade matemática (p. 18).

O fato de confundir os objetos matemáticos com suas representações semióticas é considerado pelo autor um "paradoxo cognitivo do pensamento matemático" (DUVAL, 2012, p. 268), não sendo percebido no ensino porque é dada muito mais importância às representações mentais do que às representações semióticas. Para uma melhor compreensão dessa ideia, o autor faz uma distinção entre representações mentais e semióticas.

> As representações mentais recobrem o conjunto de imagens e, mais globalmente as conceitualizações que um indivíduo pode ter de um objeto, sobre uma situação e sobre o que lhe é associado.
> As representações semióticas são produções constituídas pelo emprego de signos pertencentes a um sistema de representações que tem inconvenientes próprios de significação e de funcionamento (DUVAL, 2012, p. 269).

Também, por meio de exemplos, conforme citação a seguir, esse autor mostra a importância e o papel das representações semióticas.

> Uma figura geométrica, um enunciado em língua natural, uma fórmula algébrica, um gráfico são representações semióticas que exibem sistemas semióticos diferentes. Consideram-se, geralmente, as representações semióticas como um simples meio de exteriorização de representações mentais para fins de comunicação, quer dizer para torná-las visíveis ou acessíveis a outrem. Ora, este ponto de vista é enganoso. As representações não são somente necessárias para fins de comunicação, elas são igualmente essenciais à atividade cognitiva do pensamento (DUVAL, 2012, p. 269).

Portanto, o desenvolvimento das representações mentais depende de uma interiorização das representações semióticas, e somente essas últimas permitem preencher algumas funções cognitivas essenciais. Duval afirma que "o funcionamento cognitivo do pensamento humano se revela inseparável da existência de uma diversidade de registros semióticos de representação" (2012, p. 270).

O autor apresenta três tipos de atividades de transformação de representação semiótica que são:

> A formação de uma representação identificável como uma representação de registro dado: enunciado de uma frase (compreensível numa língua natural dada), composição de um texto, elaboração de um esquema.
> Os tratamentos são transformações de representações dentro de um mesmo registro: por exemplo, efetuar um cálculo ficando estritamente no mesmo sistema de escrita ou de representação dos números.
> As conversões são transformações de representações que consistem em mudar registro conservando os mesmos objetos denotados: por exemplo, passar da escrita algébrica de uma equação à sua representação gráfica (DUVAL, 2003, p. 16).

A conversão, segundo Duval (2003), conduz aos mecanismos subjacentes à compreensão. Mas é comum descrever a conversão como uma associação preestabelecida entre nomes e figuras, ou tratá-la como uma codificação, reduzindo o ato de conversão a uma das formas simples de tratamento. É condição para a compreensão em matemática, a diversidade de registro de representação semióticas e a articulação desses diferentes registros, embora, segundo o autor, essa diversidade raramente seja considerada por várias abordagens didáticas de ensino. Portanto, para Duval (2003), se o objetivo é analisar as dificuldades de aprendizagem em matemática, é preciso estudar prioritariamente a conversão das representações.

Portanto ao considerarmos as diferentes formas de representar um mesmo algoritmo e as dificuldades apresentadas pelos alunos na compreensão dessas, podemos confrontar com as ponderações de Duval, quando o autor relaciona as dificuldades de

compreensão na aprendizagem matemática não aos conceitos, mas à variedade de representações semióticas utilizada e ao uso "confuso" que se faz dessas. Assim como Duval considera ser condição para a compreensão em matemática, a diversidade de registro de representação semióticas e a articulação desses diferentes registros, podemos ponderar essa necessidade de articulação entre diferentes formas de representar algoritmos para sua real compreensão. Apesar de ser considerada por Duval condição para a compreensão em matemática, segundo o autor, essa diversidade raramente é considerada por várias abordagens didáticas de ensino. Deste modo, observamos que na disciplina de Algoritmos esta articulação entre diferentes formas de representação, também não é privilegiada na maioria das suas abordagens didáticas.

3.2.4 Contribuições da Psicologia sobre o desenvolvimento do pensamento

Vygotsky, um estudioso de literatura e psicólogo do desenvolvimento, direcionou seus estudos para os processos de transformação do desenvolvimento humano e deteve-se nos estudos dos mecanismos psicológicos mais sofisticados. Para Vygotsky, era preciso sistematizar uma nova abordagem sobre o processo de desenvolvimento do pensamento, que envolvesse as funções cognitivas complexas de um sujeito e possibilitasse a compreensão da natureza do comportamento humano. O autor analisa um sujeito contextualizado, histórico e sua relação com os aspectos sociais (PALANGANA, 2001, p. 92-93). O desenvolvimento, para Vygotsky, decorre das interações entre o sujeito e seu contexto social, cultural e histórico, sendo essas interações que determinam o desenvolvimento das funções mentais superiores. Ele crê que é da relação entre a fala e a inteligência prática (instrumento e o signo) que emergem as funções, caracterizadas pelo fato de o sujeito internalizar de maneira progressiva a fala, pela qual, o sujeito adquire a função de auto regulação, tornando-se capaz de controlar suas atividades mentais e seu comportamento. Assim, afirma Palangana (2001) que, "para Vygotsky, a história da socialização da inteligência é definida pela história do processo de internalização da fala social" (p. 101).

Essa capacidade de tornar-se capaz de controlar as atividades mentais e seu comportamento torna possível elaborar um plano de ação, atividade essa denominada por Oliveira (1992) de "funções psicológicas superiores ou processos mentais superiores" (p.26). A autora, com base nos estudos de Vygotsky, descreve esse tipo de atividade como

a possibilidade que o ser humano tem de "pensar em objetos ausentes, imaginar objetos nunca vividos, planejar ações a serem realizadas em momentos posteriores" (p. 26).

Para Oliveira, o conceito central para a compreensão das concepções Vygotskianas sobre o funcionamento psicológico é o de mediação. Mediação é definida como "o processo de intervenção de um elemento intermediário numa relação; a relação deixa, então, de ser direta e passa a ser mediada por este elemento" (OLIVEIRA, 1992, p. 26) Ainda sobre mediação, Oliveira aborda o conceito trabalhado por Vygotsky, em que a relação do homem com o mundo é, fundamentalmente, uma relação mediada, distinguindo dois tipos de elementos mediadores: os instrumentos e os signos. Os instrumentos são os elementos interpostos entre os indivíduos e o mundo; os signos são os elementos que representam outros objetos, eventos ou situações. Os instrumentos são elementos externos ao indivíduo, enquanto os signos são orientados para dentro e para o próprio indivíduo (OLIVEIRA, 1992).

Os signos também denominados por Vygotsky de "instrumentos psicológicos" são marcas externas, que auxiliam no desempenho de atividades que exigem memória e atenção, melhorando as possibilidades de armazenamento de informações e de controle de ação psicológica. Segundo a autora, os indivíduos, aos poucos, tornam desnecessária a utilização dessas marcas externas, transformando-se em processos internos, o que Vygotsky denominou de "processo de internalização".

Oliveira (1992), ao descrever esse processo de internalização, expõe que:

> Ao longo do desenvolvimento, o indivíduo deixa de necessitar de marcas externas e passa a utilizar signos internos, isto é, representações mentais que substituem ao objeto do mundo real. Os signos internalizados são, como as marcas exteriores, elementos que representam objetos, eventos e situações. Assim como [...], minha idéia de "mãe" representa a pessoa real da minha mãe e me permite lidar mentalmente com ela, mesmo na sua ausência (p. 35).

Para explicar as dimensões do desenvolvimento, Vigotski[2] (1998) desenvolveu o conceito de zona de desenvolvimento proximal (ZDP). Na sua concepção, a ZDP consiste na distância entre aquilo que o sujeito sabe e a possibilidade que ele tem de aprender com a ajuda de outras pessoas. O autor reconhece que o aprendizado precisa ser combinado com o nível de desenvolvimento da criança, mas afirma que esse não é o único elemento a ser considerado na aprendizagem. Argumenta, ainda, que a sintonia entre o processo de

[2] Indicaremos as grafias do autor, conforme a ficha catalográfica das obras originais que referenciamos: Vigotski (1998) e Vygotsky (1998). No decorrer do texto, quando fazemos referência a esse autor sem citar obra, optamos pela grafia "Vygotsky".

aprendizagem e de desenvolvimento revela o potencial do aprendiz, destacando a existência de, pelo menos, dois níveis de desenvolvimento: o real e o potencial.

Para Vigotski (1998), a zona de desenvolvimento proximal trata das funções que ainda não amadureceram, mas que estão em processo de amadurecimento, e estabelece parâmetros que possibilitam avaliar o grau de desenvolvimento do aprendiz. Segundo o autor, a ZDP revela o desenvolvimento potencial, estabelece o que o aprendiz já sabe, norteando estratégias de aprendizagem que levem ao desenvolvimento do sujeito e afirma: "o 'bom aprendizado' é somente aquele que se adianta ao desenvolvimento" (p. 117).

Vigotski ainda salienta que um dos aspectos essenciais do aprendizado é o fato de ele criar a ZDP, conforme afirma:

> O aprendizado desperta vários processos internos de desenvolvimento, que são capazes de operar somente quando interage com pessoas em seu ambiente e quando em cooperação com seus companheiros. Uma vez internalizados, esses processos tornam-se parte das aquisições do desenvolvimento independente da criança (1998, p. 117-118).

Segundo Vygotsky (1998), aprendizado escolar induz à percepção e desempenha um papel decisivo na conscientização da criança acerca de seus próprios processos mentais. Para o autor, a percepção e os conceitos científicos estão organizados em sistemas complexos de inter-relações. A aquisição de novos conhecimentos implica a sua inserção nesse sistema, o que demanda uma atitude de consciência e controle deliberado do aluno, a qual domina a sua relação com outros conceitos já presentes na sua estrutura psicológica.

Para Vygotsky, a existência de um sistema é a principal diferença psicológica entre os conhecimentos científicos e os cotidianos. O estabelecimento de um sistema de conceitos promove o aprendiz para níveis mais elevados de desenvolvimento. A aquisição de conceitos científicos altera, dinamicamente, a estrutura psicológica: "Uma vez que a criança já atingiu consciência e controle de um tipo de conceito, todos os conceitos anteriormente formados são reconstruídos da mesma forma. " (VYGOTSKY, 1998, p. 134). O autor observa que, aos poucos, a criança agrega à sua estrutura psicológica, novos conceitos, que lhe permitem reelaborar as suas concepções de tempo e espaço. Para Vygotsky, os conceitos espontâneos e científicos estão intimamente relacionados, já que é necessário que o desenvolvimento de um conceito espontâneo tenha atingido certo nível, para que a criança absorva um conceito científico correspondente.

Por meio das menções de Vygotsky, referente ao processo de desenvolvimento do pensamento, tornou-se possível a reavaliação de diversas questões que não eram ponderadas no planejamento das atividades docentes.

Um exemplo que não percebíamos, entendido agora, é a dificuldade que os alunos apresentavam de, em determinados momentos, "pensar em objetos ausentes", ou ainda de "planejar ações a serem realizadas". Compreendemos a necessidade que os alunos ainda apresentam, as "marcas externas", com isso, viabilizando que possamos auxiliá-los no desempenho das atividades, ou que lhes dediquemos a atenção que ainda é necessária para a mediação e seus elementos mediadores, os instrumentos e os signos.

Com base nos apontamentos de Vygotsky, sobre a importância dos signos para auxiliar no desempenho das atividades que exigem memória e atenção, bem como a afirmativa de que é somente aos poucos que os indivíduos tornam desnecessárias as marcas externas, nos induziram a refletir sobre os "processos mentais superiores" dos alunos. Cabe repensar sobre o estagio de cada sujeito, e se todos conseguem pensar, imaginar objetos ausentes, bem como esquematizar ações a serem realizadas posteriormente por eles.

3.3 Pesquisas relacionadas ao tema de estudo

Com a finalidade de lançar luz ao tema da pesquisa, buscamos informações sobre os assuntos relevantes, observando diversos trabalhos tanto da área de Informática, quanto da área da Educação Matemática, bem como da área de Educação. No levantamento, examinamos artigos, dissertações de mestrado e teses de doutorado disponíveis no site da Capes e de diversas universidades.

No início desta pesquisa, buscamos por estudos que abordassem e apresentassem ferramentas gráficas para a disciplina de Algoritmos, com a expectativa de encontrar alguma ferramenta milagrosa, que por meio de seu uso pudesse ajudar os alunos a aprender algoritmos. Mas, percebendo que a presente pesquisa envolveria coisas mais complexas do que o uso de uma ferramenta, tornou-se necessário examinar mais profundamente o contexto da sala de aula, buscando pesquisas que abordassem o tema de ensino de algoritmos. Estudos que serão apresentados neste item.

Com o objetivo de diminuir a sobrecarga cognitiva para os alunos, permitindo que eles se concentrem em organizar o pensamento e criar boas estratégias para resolver os problemas, Brandão et al. (2012) utilizaram uma ferramenta chamada iVprog, desenvolvida no departamento de Ciência da Computação do IME-USP. O experimento foi realizado com os alunos em dois cursos de Introdução à Programação, para duas turmas de Licenciatura em Matemática, em 2010 (curso II) e 2011 (curso III), dados que foram comparados aos do curso I, ministrado em 2005 que não utilizou qualquer ambiente de programação visual.

Os autores observaram um interessante aumento na frequência às aulas nos cursos II e III em relação ao curso I. As médias em avaliações e exercícios foram maiores com o uso do iVProg, inclusive as médias finais. Com base nesses números, presumem que o iVProg teve impacto positivo na motivação e no aprendizado dos alunos nos cursos II e III, sugerindo ainda que a transição para a linguagem C é melhor realizada quando o iVProg e a linguagem C são apresentados em paralelo.

O estudo de Neto (2013) expõe a avaliação da utilização do *software Scratch* no ambiente introdutório de ensino da disciplina de Lógica de Programação, realizado com alunos do curso Técnico de Informática de uma Instituição de Ensino Técnico. O autor entende que o *Scratch* é uma excelente ferramenta para o ensino de conceitos de Lógica de Programação e afirma:

> Por meio da ferramenta Scratch é possível aproximar o usuário cada vez mais do ambiente de programação, sem que haja necessariamente a necessidade de aprender uma linguagem de programação específica. Além disso, por não trabalhar com linhas de código (usa-se somente interface), possibilita a criação de programas de maneira mais simples e dinâmica, além de estimular o raciocínio lógico, e de permitir visualizar graficamente a execução do programa criado (NETO, 2013, p. 262).

Neto (2013) observou que houve uma melhora expressiva, quando comparou os resultados com os dados das turmas passadas, principalmente quanto à motivação dos alunos em prosseguir com o curso. Apresenta algumas vantagens e desvantagens identificadas entre os alunos com o uso do *Scratch* na disciplina de Lógica de Programação.

Como desvantagens, o autor menciona a baixa qualidade das imagens; a limitação quanto à quantidade e às opções, bem como os poucos recursos do editor para o tratamento das imagens, além de os disponibilizados serem restritos e mostrarem-se trabalhosos para utilização; a incompatibilidade de alguns arquivos de áudio e de vídeos; o fato de a ferramenta não ter um editor de áudio; o baixo desempenho em casos de sobrecarga de recursos (imagens, sons, músicas), o que deixa o programa pesado e, dependendo da capacidade da máquina utilizada, os comandos ficam lentos e acabam travando; o fato de precisar utilizar muitos blocos de comando para realizar algo, pois em alguns casos, conforme o que se deseja fazer, é necessária a junção de muitos blocos de comando, tornando a programação trabalhosa. Como vantagem, o autor aponta que o ambiente descontraído de programação proporcionado pelo *Scratch* torna a programação atraente, fácil e mais intuitiva, uma vez que os alunos podem inserir, em seus programas, elementos como sons e imagens produzidos por eles mesmos ou não.

Neto (2013) também indica que o uso do aplicativo no início da disciplina de Lógica de Programação, além de motivar mais os alunos, apesar das desvantagens e dificuldades que poderão ser encontradas na interação com a ferramenta, poderá facilitar o entendimento das estruturas e dos comandos. O fato defendido é de que o *Scratch* apresenta de maneira diferenciada o mundo da programação para alunos iniciantes nos cursos da área de informática, o que poderá amenizar o problema pertinente à compreensão da lógica computacional e contribuir de forma significativa para a formação do profissional da área.

Outra pesquisa foi desenvolvida com o curso Profissional Técnico de Gestão e Programação de Sistemas Informáticos de nível secundário, numa escola da cidade de

Lisboa/PT, com uma turma constituída por 21 alunos com idade entre 14 e 17 anos (SANTOS, 2013). Santos (2013) apresentou como uma das características da turma, o fato de que somente um dos alunos não tinha repetências, os demais reprovaram um ou mais anos. O autor descreve três razões que foram as mais repetidas pelos alunos para justificar a dificuldade que sentem na escola: falta de gosto pela aprendizagem; falta de hábitos e métodos de trabalho e estudo; e dificuldades em compreender a explicação do professor. Com o objetivo de amenizar as dificuldades, elaborou estratégias pedagógicas e implementou um projeto modular para desenvolver os conteúdos, e fez uso de uma Linguagem de Programação Visual, o *Scratch*. Considerou que, por meio dessa linguagem, foi possível obter resultados mais rápidos, evitando desgaste e frustrações dos alunos na utilização da sintaxe, fornecendo um ambiente de desenvolvimento atrativo e fácil de usar.

Santos (2013) ainda destaca que o uso do *Scratch* se apresentou como bom veículo para a introdução da programação aos alunos, mas que durante a aplicação da ferramenta constatou a necessidade de prolongar a fase de desenvolvimento de pequenos problemas. Tempo necessário para que, assim, os alunos pudessem familiarizar-se melhor com a linguagem de programação visual e desenvolver a sua capacidade de construção de algoritmos.

Com base em estudos na literatura relacionada com o ensino-aprendizagem de algoritmos e a programação de computadores, Jesus e Brito (2009) apresentam suas concepções, pontuando as principais dificuldades enfrentadas pelos alunos e expondo o perfil dos professores que ministram essa disciplina. As autoras salientam que uma característica importante dos algoritmos é que não existe uma única solução para um determinado problema. Alegam que, como é necessário aplicar lógica na construção de um algoritmo e/ou programa, a solução é bastante subjetiva, pois o raciocínio lógico é particular, de cada pessoa. Apontam que para um determinado problema é possível apresentar várias soluções com caminhos diferentes, mas todos serão capazes de alcançarem o mesmo resultado.

Outra questão apontada por Jesus e Brito (2009) é que os professores da disciplina, geralmente, apresentam a teoria por meio de modelos prontos, ou seja, fazendo uso de técnicas, estratégias e soluções do professor. Isso leva o aluno a reproduzir o que foi apresentado em sala ao se deparar com um problema a ser solucionado e uma solução a ser formalizada. Perceberam que o aluno, naquele momento da aula, tem a percepção de que entendeu a solução apresentada pelo professor, mas como não compreendeu como esse processo se desenvolveu, tem o desenvolvimento de suas próprias soluções dificultado.

Portanto, o que foi passado para o aluno, foi um modelo de algoritmo pronto e não o processo de desenvolvimento desse modelo.

As autoras descrevem que os alunos não sabem por onde começar quando um problema lhe é apresentado e uma solução para esse, solicitada, que isso se deve ao fato de os alunos apresentarem dificuldades na interpretação do enunciado do problema. Eles também não conseguem identificar no texto quais são as variáveis de entrada, o que precisa ser processado e quais são as variáveis de saída. Ou seja, ter conhecimento da sintaxe e da semântica das linguagens de programação não é suficiente se os alunos não conseguem compreender e propor soluções para os problemas, etapa que antecede a construção de algoritmos e programas computacionais.

Jesus e Brito (2009) concluem que é de suma importância que professores que lecionam algoritmos e programação de computadores tenham uma definição bem clara da concepção de ensino-aprendizagem desses conteúdos, a fim de que possam desenvolver metodologias que realmente contribuam para o desenvolvimento das competências e habilidades exigidas, além de fazerem uso adequado de aplicativos destinados a esse fim.

Por meio de um estudo de caso, Setti (2009) observou na disciplina de Lógica de Programação do Curso de Tecnologia de Sistemas para Internet, que mesmo os estudantes que conseguem resolver problemas matemáticos, encontram dificuldades na passagem do raciocínio matemático para o correspondente computacional. Na visão da autora:

> Para realizar esta passagem, é necessário utilizar os conhecimentos adquiridos previamente com um novo formato, devido ao processo de discretização necessário para transformar o raciocínio matemático no correspondente computacional. Para conceber esta discretização, os conhecimentos matemáticos estabelecidos irão sofrer uma "ruptura epistemológica", pois se trata de uma mudança na forma de compreender um conhecimento (SETTI, 2009, p. 10).

Segundo Setti (2009), nos últimos anos, alguns questionamentos têm surgido, por conta das preocupações com a aprendizagem de algoritmos e as dificuldades dos alunos em relação às novas formas de pensamento que surgem diante da necessidade de trabalhar com processos repetitivos e iterativos. Esses processos envolvem a identificação das regularidades do problema que se quer solucionar e a consequente discretização do raciocínio, necessária à transformação desse no correspondente de representação computacional. Também destaca, a autora, que, normalmente, aqueles alunos que não compreendem esses processos, tendem a ficar desmotivados, fazendo com que tais dificuldades se agravem ao longo do curso.

Para a autora, o estudo de caso revelou que os alunos elaboravam o raciocínio esperado mais facilmente, utilizando fluxograma do que utilizando o pseudocódigo. Ainda, que, por meio do fluxograma, os alunos perceberam mais claramente a relação hierárquica entre as ações e que esse mostrou-se mais intuitivo e adequado para a introdução à aprendizagem de algoritmos (SETTI, 2009).

Outra conclusão, oriunda do estudo de Setti (2009), declara que a principal barreira na aprendizagem de algoritmos é constituída pela dificuldade em discretizar o raciocínio matemático para conceber o raciocínio computacional, aliada à dificuldade em perceber a regularidade das situações em análise. De outro modo, a autora aponta que os alunos que não elaboraram o raciocínio matemático adequado, na maioria dos casos, não propuseram a solução algorítmica e aqueles que esboçaram algum tipo de solução não tiveram êxito. O que a levou a concluir que a capacidade de elaborar o raciocínio matemático adequado não é suficiente para garantir o sucesso na elaboração do raciocínio algorítmico correspondente, embora seja necessária.

Numa pesquisa mais atual, Fonseca et al. (2015), visando contribuir com o processo de ensino e aprendizagem na disciplina de Lógica de Programação, apresentaram a proposta de implementação de uma ferramenta de autoria e aplicação de prova para alunos dos cursos da área de computação com correção automática, integrando as ferramentas *Blockly* e Boca. Enquanto o *Blockly* é uma ferramenta de programação que faz uso de uma linguagem visual, permitindo aos usuários escrever códigos conectados em blocos, o Boca é um sistema utilizado para gerenciar competições de programação de computadores.

Os autores projetaram a ferramenta para realizar avaliações automáticas de lógica de programação, e possibilitar a criação, a correção e o reuso de questões e provas, cadastradas na base de dados do sistema pelo professor, gerando um retorno imediato para o aluno a cada submissão de questão. A proposta de desenvolvimento do ambiente é considerada por Fonseca et al. (2015) como de fundamental importância para uma estruturação fundamentada pedagogicamente para o processo de mediação do conhecimento.

Para acrescentar, destacamos uma pesquisa bibliográfica, com levantamento de artigos que apresentam ferramentas TICs e abordagens para o ensino-aprendizagem de programação (VIEGAS et al., 2015). Os autores pesquisaram artigos que descrevem o uso das várias ferramentas como *Scratch*, Alice, *Greenfoot*, *Feeper*, *Prog1box*, *TstView*, MIT-App Inventor, Proglib, Visual Jo2 .

Numa análise geral, Viegas et al. (2015) apontaram que todas as ferramentas apresentadas nos artigos auxiliam no ensino-aprendizagem de programação de alguma forma, pois como essas foram pensadas para o ensino, todas permitem que o aluno seja considerado um agente ativo, porém, a maioria não apresenta os conceitos de programação, nem aumenta o nível de dificuldade das atividades propostas. Os autores observaram que tais ferramentas se preocupam apenas com o resultado final, não levando em conta todo o processo de ensino-aprendizagem que deveria embasá-las, considerando que isso pode levar o aluno a se desmotivar, pois não encontra, na ferramenta, subsídios auxiliares para seu aprendizado, nem desafios para motivá-los.

Viegas et al. (2015) concluem que o uso das TICs auxilia no processo de ensino-aprendizagem de programação, mas o professor deve planejar suas aulas para aproveitar ao máximo o que essas ferramentas podem oferecer. Além disso, alertam que "não se pode pensar que elas sozinhas irão ensinar e resolver as dificuldades que disciplinas de programação apresentam. O professor é responsável pela intermediação sujeito-objeto" (2015, p. 785).

Outro apontamento feito por Viegas et al. (2015) é que a maioria dos artigos pesquisados apontam essas ferramentas como sendo úteis apenas para a introdução aos conceitos de programação. Os autores ainda questionam: "Se essas ferramentas de fato auxiliam na introdução aos conceitos de programação, porque normalmente elas não se encontram na introdução desses conceitos?!" (p. 785).

A leitura desses estudos que abordaram e apresentaram ferramentas gráficas para a disciplina de Algoritmos e pesquisas que trataram o ensino de algoritmos permitiu reflexões que possibilitaram traçar possíveis direcionamentos para nossa pesquisa, bem como ampliaram nosso conhecimento sobre o tema em estudo.

4 ASPECTOS RELACIONADOS À DISCIPLINA DE ALGORITMOS

Neste item serão apresentados aspectos que envolveram as aulas de Algoritmos, a metodologia utilizada na disciplina, bem como as dificuldades apresentadas pelos alunos.

4.1 Metodologia utilizada nas aulas

As aulas da disciplina são expositivas e práticas, acontecem em um laboratório de informática onde cada aluno utiliza um computador, entre os disponíveis no laboratório ou o seu próprio notebook. Essas aulas são ministradas sempre instigando a cooperação entre os alunos, lembrando-os de que a construção de algoritmos é um processo individual, pois com base em determinado problema enunciado, o aluno deverá ser capaz de criar uma solução. Inicialmente, poderá ter ajuda dos colegas e/ou professor, mas aos poucos deverá desenvolver autonomia suficiente para construir as respostas sozinho.

Para disponibilizar o material das aulas e acompanhar a construção dos exercícios é utilizado o Ambiente virtual Moodle, e com a intenção de estimular a cooperação entre os alunos, propomos fóruns para que esses pudessem compartilhar os exercícios construídos. Como ilustrado na Figura 23, no tópico 3, estão disponíveis os slides da Aula 1, logo abaixo a lista proposta de exercícios 1 e em seguida o fórum denominado “Postar exercícios Lista 1 (Pseudocódigo ou c++)” .

Figura 23 – Ambiente da disciplina no Moodle

Fonte: da pesquisa.

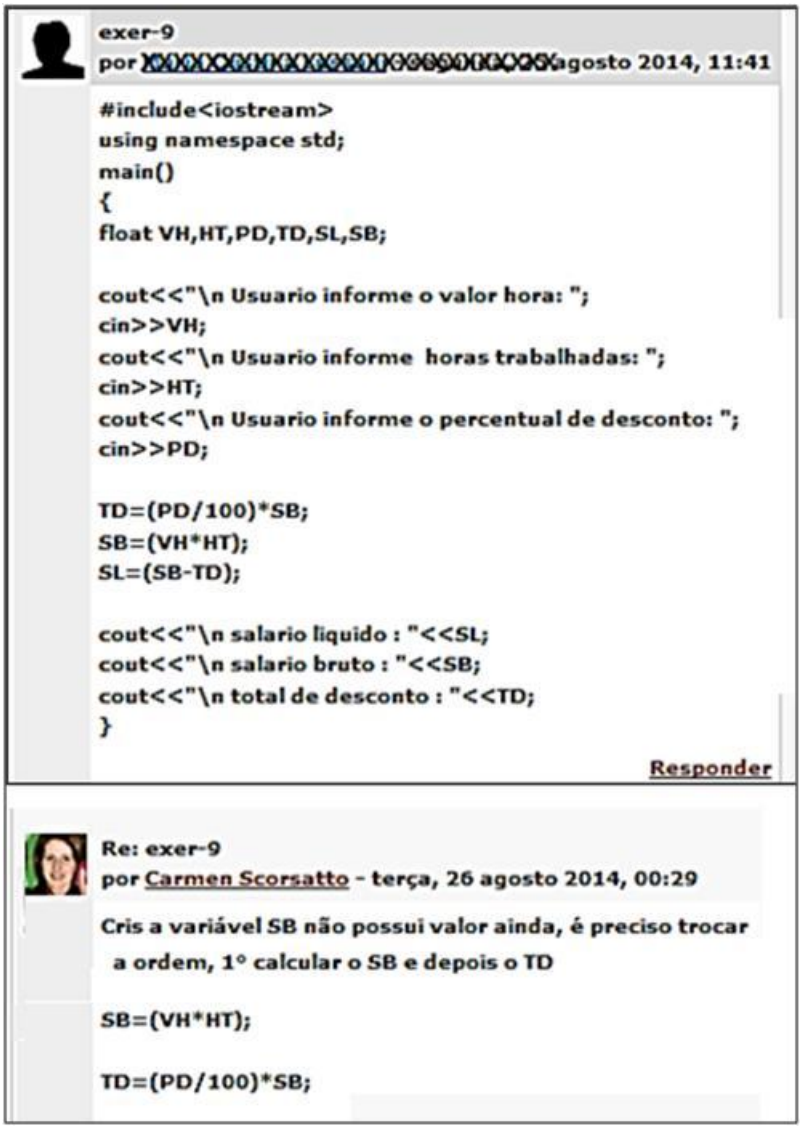
Ao propor os fóruns, almejávamos que cada um acompanhasse a lista de exercícios no seu ritmo e que fossem postando seus algoritmos. A proposta do fórum era compartilhar os códigos dos exercícios, para que, no momento em que os alunos se deparassem com dúvidas, pudessem olhar nos já compartilhados pelos colegas e, assim, conseguissem desenvolver os seus algoritmos, não somente dentro do horário de aula.

Figura 24 - Fórum de exercício da lista 1

Fonte: da pesquisa.

Dentro das possibilidades para cada exercício proposto, foi dado um *feedback*, com correções ou sugestões para o exercício (Figura 25). Esses poderiam ser feitos não só pela professora, como também pelos colegas, apesar de que poucos se arriscaram a contribuir com comentários. Além da professora e os alunos participantes, o monitor da disciplina de Algoritmos tinha permissão para dar o *feedback*. Os monitores são bolsistas do Ifsul, que estão em níveis mais avançados do curso, permanecem por um turno à disposição dos alunos para tirar dúvidas e auxiliá-los na disciplina.

Figura 25 - Feedback do Fórum



Fonte: da pesquisa.

Quanto ao processo de avaliação da disciplina, foi feito por meio de provas individuais escritas (no papel), ou práticas (no computador), foi disponibilizada aos alunos a opção de escolher a forma em que se sentiam seguros para resolver. Observamos que a prova prática feita no computador, muitas vezes, pode prejudicar alguns alunos, pois esses acabam ficando ansiosos e não conseguem encontrar erros de sintaxe apresentados pela linguagem no momento da execução do algoritmo. Por sua vez, a escrita poderia prejudicar os alunos que gostam de digitar e testar os algoritmos, pois, sentir-se-iam desmotivados ao ter que escrever no papel, sem ter a possibilidade de ver a execução.

Foram realizadas duas provas por semestre, uma para cada etapa, os alunos necessitam ser aprovados em ambas, por sua vez, quem não obteve êxito nas etapas

oferecidas teve a oportunidade de reavaliação para a recuperação. Todas as provas foram entregues na aula posterior e refeitas no grande grupo.

Os percentuais de evasão e de reprovação apresentados na disciplina mostram o quão complexa é a atividade de ensino e aprendizagem de Algoritmos. O percentual de alunos aprovados na turma anterior a essa da pesquisa, a de 2014-1 (Figura 26) foi de 36,4%, ou seja, somente oito passaram em uma turma de 22 alunos, cinco desistiram ainda na primeira etapa e nove reprovaram por nota.

Figura 26 - Turma 2014-1

Nº Alunos	%
8	36,4
5	22,7
9	40,9
22	100,00

Fonte: da pesquisa.

A soma dos percentuais de reprovação e evasão ultrapassam 60%, índices que deixam clara a necessidade de estudo. Apesar da singularidade da sala de aula, existem pesquisas acadêmicas que se propõem a estudar as situações de ensino, objetivo que é reforçado por Gauthier et al. (2013) quando alegam que "É possível assim estudar o ensino enquanto fenômeno estável, sem, todavia, esquecer a singularidade das situações tanto na compreensão dos fenômenos quanto da utilização dos resultados da pesquisa" (p. 90).

No decorrer do semestre, apesar da nossa disponibilidade de, como professora, sempre que solicitada, revisar os conteúdos, alguns alunos continuaram com dificuldades em compreender determinados conteúdos de Algoritmos. A seguir, descreveremos sobre essas dificuldades observadas com a turma escolhida para esta pesquisa.

4.2 Dificuldades apresentadas pelos alunos

Observando as reprovações e as dificuldades apresentadas pela turma que antecedeu a 1M1/2014-2, buscando encontrar explicação para essas, encontramos em Brousseau (2008) apontamentos que indicam que quando o aluno não consegue resolver determinado problema, fica clara a necessidade de um ensino mais adequado. Para o autor, a marca de um saber, na concepção geral de ensino, é a associação de boas perguntas e respostas, conforme exemplifica "O professor propõe um problema; se o aluno resolver,

mostra que sabe; caso contrário, fica clara a necessidade de conhecimento, que requer uma informação, um ensino" (2008, p. 35). Para o autor, todos os procedimentos aceitáveis para fazer com que o aluno adquira esse saber são aqueles que o professor não dá a resposta, sendo que esse deveria realizar uma seleção de questões a serem trabalhadas que fosse centrada em problemas que provoquem o aluno.

As respostas da turma na pergunta (Apêndice A) que dava conta de esclarecer se os alunos estariam encontrando dificuldades, com justificativa, tanto em caso afirmativo como negativo, demonstrou que a principal dificuldade encontrada pelos alunos é de "Pensar" em como elaborar uma solução para os problemas propostos. Serão apontadas, a seguir, algumas das dificuldades registradas pelos alunos da turma em análise, sempre que indagados durante as aulas:

- Pensar em como elaborar uma solução para os problemas propostos.
- Interpretar os enunciados.
- Necessita de muito raciocínio.
- Utilizar as fórmulas, os cálculos matemáticos.
- Converter as expressões matemáticas para expressão computacional.
- Converter do pseudocódigo para a linguagem C++.
- Manipular vetores e matrizes;

Essas dificuldades assinaladas também foram percebidas no acompanhamento das atividades práticas realizadas pelos alunos. Observamos erros recorrentes praticados pelos mesmos alunos, o que confirma a assertiva de Brousseau (2008), ao destacar que os obstáculos se manifestam pelos erros, e que esses não desaparecem com a aprendizagem de um novo conceito, e, sim, oferecem resistência à sua compreensão, retardando sua aplicação, sendo, portanto, inútil ignorar um obstáculo. Almouloud (2007), com base em Brousseau, expõe que o estudo dos obstáculos passa pelo estudo de erros resistentes por parte dos alunos.

Com base em tais reflexões sobre erros e obstáculos, ficaram mais evidentes as dificuldades apresentadas pelos alunos da disciplina de Algoritmo da turma 1M1. Muitos desses erros são habilidades necessárias para que o conhecimento evolua, assim, é inútil ignorar determinados erros, pois a ideia de que um novo conhecimento fará com que o aluno supere as dificuldades anteriores é falsa.

Se assumimos as premissas de que o conhecimento para o qual os alunos apresentam dificuldade deveria ser ensinado de forma a oferecer desafio aos discentes e, ainda, que é inútil burlar o imperativo dessa aprendizagem, podemos afirmar que

dificuldades com relação ao ensino e aprendizagem de algoritmos não pertencem somente aos alunos, sendo, também, vistas como dificuldades do professor, já que esse deveria elaborar uma proposta que desse conta dessas percepções em sala de aula. Como ministrar uma disciplina baseada praticamente em resolução de problemas sem ensinar como resolver, sem dar respostas. Como fazer o aluno aceitar o problema como "seu" até que se produza a resposta. Como envolver todos os alunos, fazer com que se comprometam com o seu aprendizado e aceitem o desafio de resolver os problemas aparentemente sozinhos. Como tratar os erros dos alunos? Buscando por respostas, encontramos em Almouloud (2007) a afirmativa em relação à aprendizagem de conceitos matemáticos, de que a maioria dos pesquisadores em didática da matemática defende a ideia de que o tratamento que os professores dão ao erro do aluno é um fator influente na aprendizagem.

A análise dos conceitos apresentados por Brousseau (2008) e demais autores citados, referente a erros e obstáculos, nos ajudou a compreender melhor o contexto que envolve esta pesquisa, que iniciaremos, a seguir, a descrição.

5 FERRAMENTA GRÁFICA *BLOCKLY* NA DISCIPLINA DE ALGORITMOS

O estudo foi realizado utilizando a ferramenta do *code.org*, construído com a linguagem *Blockly*, que apresenta os conceitos de algoritmos na forma de desafios e utiliza o estilo da programação gráfica de arrastar e soltar os blocos de comandos. Propusemo-nos a investigar se a ferramenta gráfica *Blockly* contribui para o ensino-aprendizagem de algoritmos, no Curso Superior de Tecnologia em Sistemas para a Internet (TSPI).

Portanto, será exposta, neste capítulo, a descrição do estilo padrão das atividades da ferramenta bem como a relação dessas com os conceitos de algoritmos. Além disso, será esclarecido como foi, nas aulas de Algoritmos, a sua utilização pela turma, a descrição e a análise dos dados originados das atividades realizadas por cada aluno, relacionando com os exercícios construídos nas atividades de sala de aula e com as nossas observações, no papel de professora.

5.1 Descrição dos padrões das atividades utilizadas no *software Blockly*

O principal objetivo da ferramenta *Blockly* é apresentar os conceitos de algoritmos na forma de desafios que devem ser resolvidos, totalizando 98 passos, distribuídos entre nove atividades, sendo que essas não apresentam o mesmo número de passos. Os passos sempre iniciam com algum comando já incluso, para estimular o aluno a prosseguir e a finalizar, dando um *feedback* a cada tentativa, informando se completou com sucesso, se poderia ter concluído com menos comandos ou se os comandos não estão na ordem correta para a execução.

De maneira geral, podemos afirmar que os conteúdos apresentados pela ferramenta (Quadro 1) e os conteúdos definidos no plano de ensino de Algoritmos (Quadro 2) envolvem praticamente os mesmos conceitos.

Quadro 1 – Conceitos abordados pelo *software*

Sequência	Sequência
Blocos	"se"
	"se" - "se não"
Bloco	"Repetir Vezes"

Laços	“Repita até que"
	"Enquanto"
	"Contador"
Bloco Funções	Funções simples
	Funções com parâmetros

Fonte: da pesquisa.

Elencando os conceitos de ambos (Ferramenta e Plano de Ensino), observamos que os conceitos em que a ferramenta não abrange e que faz parte do conteúdo programático da disciplina são a seleção múltipla (case) e as estruturas de vetores e matrizes.

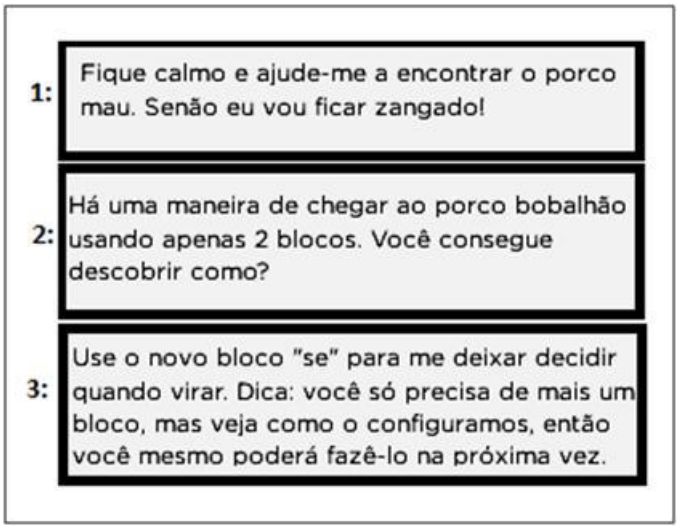
Quadro 2 – Conceitos abordados no Plano de Ensino.

Algoritmos	Sequenciais
Com Seleção	Simples "se"
	Composta "se" - "se não"
	Múltipla "case"
Com Repetição	“Enquanto” / “Faça ....Enquanto”
	Com variável de controle "Para” “contador"
Com Estrutura de dados Homogêneas	Vetores e Matrizes
Funções	Funções simples/ com parâmetros

Fonte: da pesquisa.

Ao apresentar o passo da atividade, a ferramenta dá dicas para que o aluno consiga executá-lo, apresenta, paulatinamente, os passos e prepara o ambiente para um melhor entendimento dos próximos conceitos, conforme a sequência de enunciados que será exposta a seguir (Figura 27).

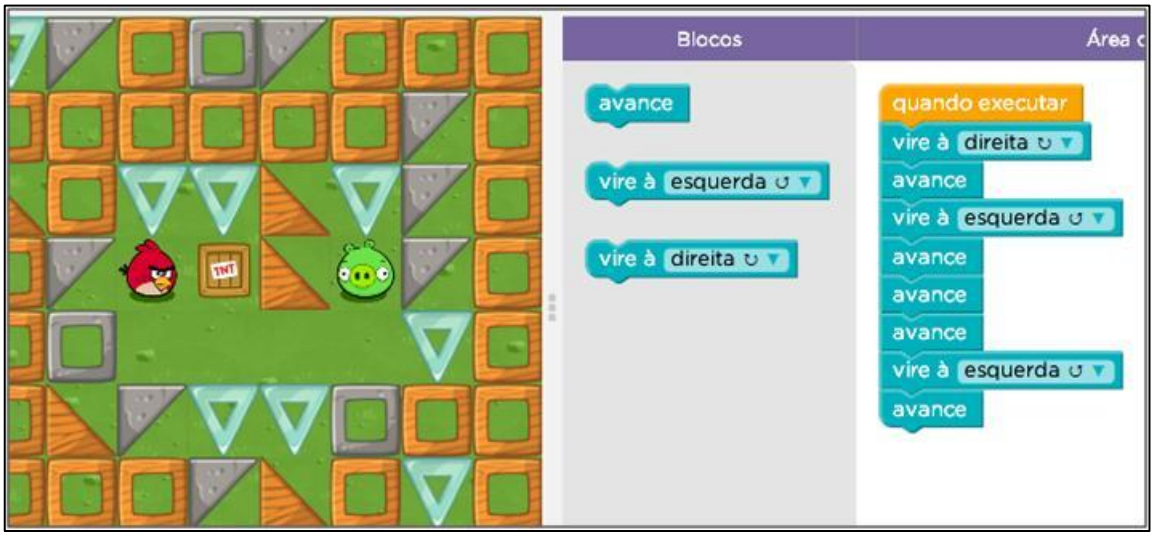
Figura 27 - Exemplo de enunciados



Fonte: da pesquisa.

No primeiro enunciado da Figura 27, mesmo que ainda não tenha apresentado o comando repetição, o passo direciona para a necessidade de utilizar o "repita 5 vezes" do segundo enunciado, pois ao solicitar o trajeto para encontrar o objeto "porco" era necessário repetir o comando "avance". Conforme demonstramos pela resolução construída por um dos alunos (A_{16}) ilustrado (Figura 28).

Figura 28 – Resolução do primeiro enunciado (A_{16}).



Fonte: da pesquisa.

Assim, no segundo enunciado, da Figura 27, a proposta era de construir o trajeto com apenas dois blocos de comandos e só então a ferramenta apresenta o comando "repita 5 vezes". Processo utilizado para que o aluno conclua que, ao invés de usar cinco blocos do

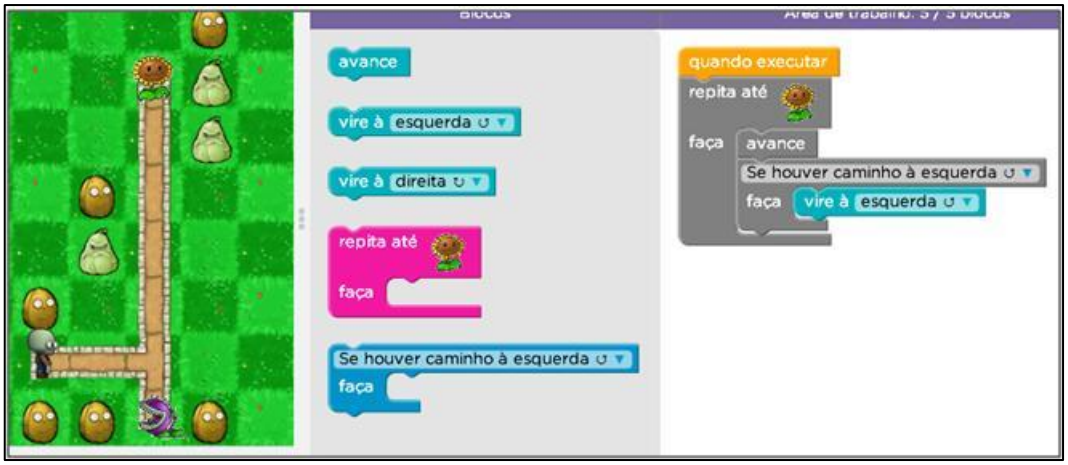
comando “avance”, terá o mesmo resultado se utilizar somente um “avance” envolto pelo “repita 5 vezes” (Figura 29).

Figura 29 - Resolução do segundo enunciado (A_{16})

Fonte: da pesquisa.

Para o terceiro enunciado, da Figura 30, a ferramenta além de disponibilizar o comando “repita” já anteriormente apresentado, dá dica de como utilizar o comando “se”, utilizado para decidir quando virar à esquerda. Esse enunciado também solicita a atenção do aluno para que ele observe como esse foi configurado, para que, assim, configure-o sozinho no próximo passo. Os comandos na cor cinza, apresentados no decorrer dos passos das atividades são incluídos pela própria ferramenta, com o intuito de auxiliar aos alunos a entender o funcionamento desses (Figura 30).

Figura 30 - Resolução do terceiro enunciado (A_{16})



Fonte: da pesquisa.

O único comando selecionado para a resolução do passo apresentado no passo da Figura 30 é o "vire à esquerda", pois os demais estão na cor cinza e já estavam inseridos.

Assim como demonstrado nos três enunciados e nas resoluções apresentadas construídas pelo aluno A_{16}, a ferramenta vai induzindo o aluno para que esse consiga executar os passos da atividade sozinho, vai apresentando os passos de maneira a facilitar o entendimento dos próximos conceitos. Ainda assim, permite finalizar cada passo da atividade, mesmo com os blocos de comandos incorretos ou selecionados na ordem incorreta, aceita que prossiga sem passar por todos os passos propostos. Por exemplo, se o aluno não conseguiu finalizar todos os passos propostos da atividade dois, mesmo assim, poderá iniciar a atividade cinco, assim como, também, poderá iniciar em qualquer um dos passos dentro de cada atividade.

A correção dos passos feita pelo *software* permite, assim, como nos algoritmos, diferentes possibilidades de resolução, respeitando a maneira pela qual cada pessoa apresenta uma solução. Isso pode ser observado nas Figuras 31 e 32 do passo 13.1, em que cada aluno (A_{19} e A_{20}) resolveu usando comandos e sentidos diferentes.

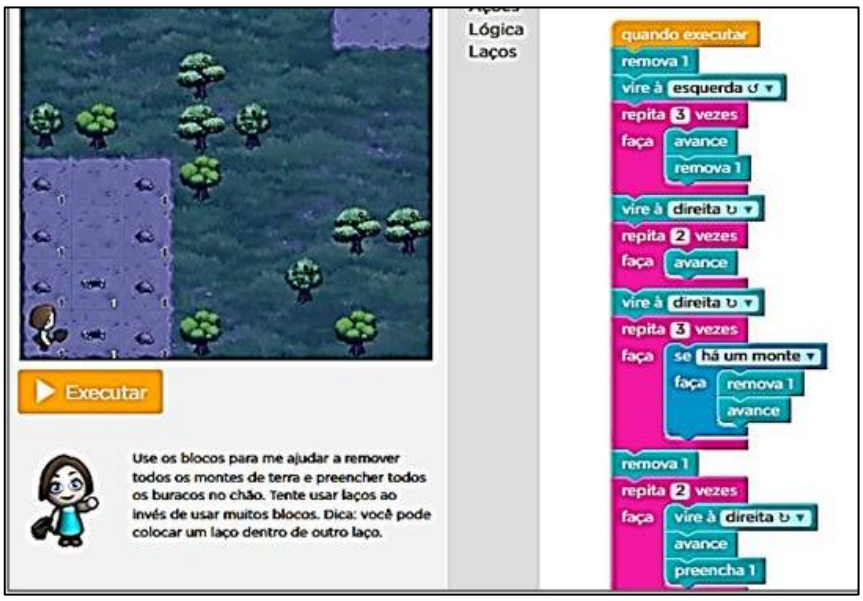
Na figura 31, observamos que a resolução proposta coloca todos os comandos dentro do "enquanto houver caminho em frente faça", e para garantir que todos os espaços estarão sendo analisados, envolve esse comando por outro de repetição, comando "repita 20 vezes".

Figura 31 – Exemplo 1 da resolução do passo 1 da atividade 13 (A_{19})

Fonte: da pesquisa.

Enquanto o mesmo passo criado por outro aluno (A_{20}) foi construído de modo diferente, e metodicamente planejado, como ilustra a Figura 32, as instruções foram selecionadas de maneira a resolver o passo de maneira fragmentada.

Figura 32 - Exemplo 2 da resolução do passo 1 da atividade 13 (A_{20})

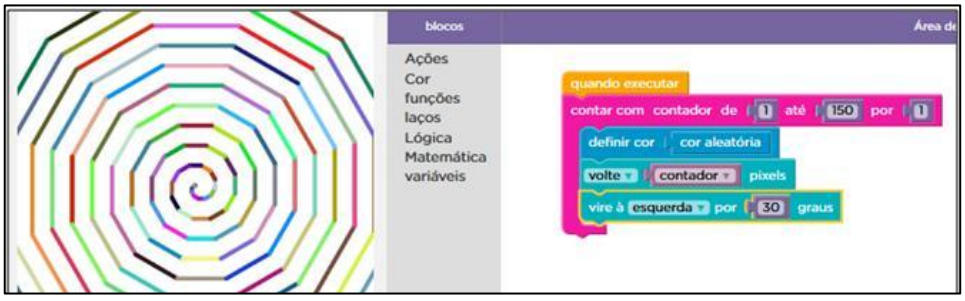


Fonte: da pesquisa.

Com a ilustração desses passos recém-apresentados descrevemos como conseguimos acompanhar o caminho proposto por cada aluno, ou seja, o plano de ação construído por eles para resolver o mesmo enunciado.

Na grande maioria das atividades, os enunciados solicitam para, simplesmente, completar o desafio, seguir os passos anunciados, não há a necessidade de criar. Porém, outros passos das atividades solicitam que seja criada uma imagem, com uso dos comandos já conhecidos, ou que seja alterado o código para criar imagens, como ilustrado na Figura 33.

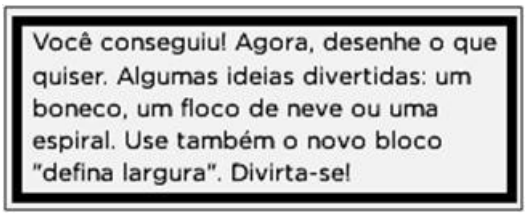
Figura 33 - Exemplo de um passo que solicita criar



Fonte: da pesquisa.

Notamos que nos passos que exigem do aluno o “criar”, são transmitidas, nos enunciados, algumas ideias, com a intenção de motivá-los (Figura 34).

Figura 34 - Exemplo de enunciado para criar



Fonte: da pesquisa.

Para esses passos, em que o aluno pode criar, a ferramenta oferece a opção de salvar todas as imagens em uma galeria, permitindo sua visualização e/ou impressão sempre em que ao aluno desejar.

5.2 Sobre a utilização do *Blockly*

Nossa formação inicial é em Ciência da Computação e seguimos a mesma metodologia de ensino utilizada pelos nossos professores de Algoritmos, a saber: inicialmente desenvolver os conteúdos e depois apresentar listas de exercícios, seguindo sempre a ideia de “primeiro a teoria e depois prática”. Com essa prática supomos que deveríamos aplicar a ferramenta gráfica de construção de Algoritmos após ter finalizado os conteúdos propostos no plano de ensino, com a expectativa de que isso se constituísse numa maneira de auxiliar os alunos a aplicar e ampliar seus conhecimentos, supostamente já internalizados.

A aplicação da ferramenta para turma 1M1-2014/2 iniciou na segunda quinzena do mês de outubro, faltando praticamente pouco mais de um mês para finalizar as aulas do semestre. Após ter apresentado a ferramenta, disponibilizamos seu acesso por um link no Moodle, ambiente virtual utilizado para disponibilizar material das aulas. Justificamos o seu uso aos alunos, como uma aplicação com o objetivo de ajudá-los a revisar e ampliar os conceitos já estudados em aula. Após cadastrarmos a turma, foi possível inserir os alunos e gerar uma senha individual, assim, no decorrer do processo em que cada aluno foi realizando os passos das atividades, a ferramenta foi salvando-os automaticamente.

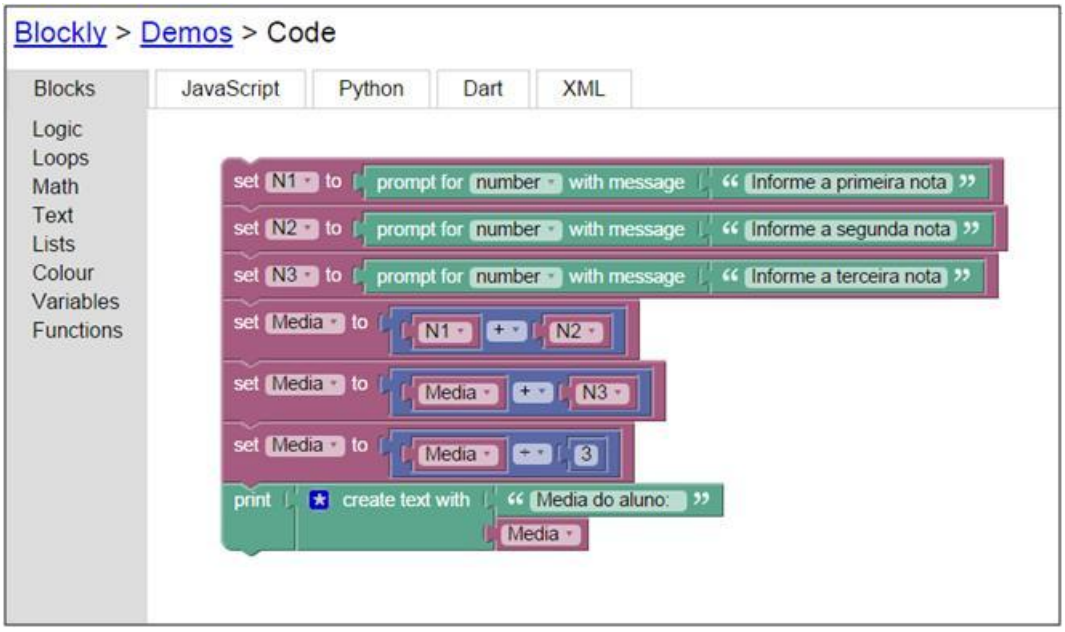
Para cadastrarmos a turma foi necessário criar uma conta no site da code.org e identificar essa conta do tipo “professor”, assim foi possível após finalizar o cadastro, inserir todos os alunos, bem como selecionar a ferramenta construída com o Blockly e disponibilizar para esta turma, a escolhida foi *Computer Science Fundamentals*.

Nessa primeira aula, os alunos pareceram empolgados, foram executando os passos com nossa ajuda e também dos colegas. A proposta do uso da ferramenta gráfica foi considerada uma atividade extra, a qual seria utilizada nos períodos finais da aula e como atividade extraclasse, ou seja, na sua maior parte realizada fora do horário de aula. Apesar de salientar a importância do seu uso, deixamos claro o caráter não obrigatório.

Mesmo sendo de livre adesão, por sugestão dos alunos, foi estabelecida uma aposta como desafio entre os participantes, de que quem conseguisse finalizar todas, ou quase todas as atividades propostas, ganharia, no último dia de aula, uma caixa de cerveja. Sugerimos (professora), então, que o prêmio poderia ser convertido em caixas de bombons. Assim, cinco alunos ganharam suas caixas como recompensa.

Além da aplicação da ferramenta, juntamente com a avaliação final da Etapa II, solicitamos um trabalho em que cada aluno escolhesse um algoritmo da lista dos exercícios implementados na linguagem padrão utilizada nas aulas (C++) e construísse no *Blockly*, como está ilustrado no exemplo da Figura 35, construído por um dos alunos (A_8).

Figura 35 - Exemplo de Algoritmo criado com o *Blockly* (A_8)



Fonte: da pesquisa.

Esse algoritmo representado em forma de blocos solicita via teclado a entrada de três notas, depois atribui para a variável média, a nota 1 (N1) e a nota 2 (N2); após, sobrepõe à variável média o valor dela mesma adicionando a nota 3 (N3); depois atribui, novamente, para a variável média seu conteúdo dividido por 3. Para finalizar mostra na tela a mensagem “Media do aluno” juntamente com o valor calculado da média das notas (N1, N2 e N3) informadas.

Mesmo que a proposta de uso da ferramenta gráfica, na sua maior parte, tenha sido como uma tarefa extraclasse observamos que gerou trocas de ideias entre alguns alunos durante as aulas, quanto à forma de resolver os passos estabelecidos nas atividades. Apesar de termos deixado claro seu caráter não obrigatório, salientamos a importância do seu uso, já que as atividades demonstravam de forma gráfica os conceitos já estudados em Algoritmos.

5.3 Aprendizagem de algoritmos e o uso da ferramenta gráfica *Blockly*

Inicialmente, a análise dos dados da pesquisa foi feita pela observação dos passos das atividades de cada aluno, sendo que no total foram 98 passos distribuídos entre as nove atividades propostas pela ferramenta gráfica. No Apêndice-B, apresentamos como exemplo todos os passos resolvidos de um dos alunos (A_{24}) que concluiu todas as atividades. Esse processo foi feito em duas etapas, na primeira, foram estudados todos os passos executados por cada aluno, observando o processo de desenvolvimento de cada passo construído, investigando de forma individual os conteúdos envolvidos nos passos em que os alunos desistiram, ou aqueles que foram finalizados com sucesso; quais foram os passos que poderiam ter sido concluídos com menos comandos; se esses estavam na ordem correta para a execução; acerca dos erros executados em comum, investigamos se, aparentemente, os alunos superaram algumas dificuldades com a evolução dos passos das atividades. Na segunda etapa, juntamos as observações feitas pela professora ao *feedback* gerado pela ferramenta, conforme ilustrado no Quadro 3, a seguir.

Quadro 3 – Análise individual

Aluno	Observações	*Feedback* da Ferramenta
A7	**Entendeu** - Sequência de passos - Soube usar o "se" e "senão" **Dificuldades** - Agrupar comandos semelhantes para não inserir estes duplicados - Não entendeu que os comandos incluídos dentro do laço executarão até encontrar o critério final - Para cada sequência de comandos criou um laço - Definiu todos os comandos sem perceber que poderia mandar repetir **Evoluiu** -Com o evolução dos exercícios mostrou ter entendido o uso do "repita até encontrar" - "Enquanto" e "repita 3 X" **Desistiu** - Fez menos de 50% das atividades - Parou sem erros no passo final)	

Fonte: da pesquisa.

Nesse *feedback* gerado na ferramenta são expostos troféus que foram preenchidos conforme os conceitos envolvidos nos passos das atividades executadas corretamente. Assim, quando um aluno completou todas as atividades referentes a um determinado conceito, esse foi recebendo troféus de acordo com seu progresso nas cores bronze, prata ou ouro. Essas imagens foram importantes, pois permitiram identificar mais rapidamente quais foram os conceitos envolvidos nas atividades em que os alunos conseguiram realizar.

Para iniciar a apresentação dos dados da pesquisa, apresentaremos o resultado acerca do desempenho geral da turma em estudo, em seguida, a análise do processo individual de utilização do *software* pelos alunos.

5.3.1 O desempenho geral da turma 1M1-2014/2

O número de alunos reprovados na turma 1M1-2014/2 foi um dos maiores entre as quatorze turmas da disciplina de Algoritmos nos seis anos do curso do TSPI do Ifsul, fato que demandou questionamentos sobre o que teria essa turma em particular para ter apresentado esse alto índice de reprovação. Por meio de informações obtidas no setor responsável pelos registros acadêmicos, verificamos que essa é a primeira turma para a qual foi realizado sorteio para preencher vagas do processo seletivo. De modo que, nove alunos, dos trinta que iniciaram na turma, não participaram do vestibular que é o processo seletivo habitual, mas foram sorteados. Observando o desempenho desses nove alunos, constatamos que somente um foi aprovado, sendo que esse já havia iniciado um curso superior em informática há uns quinze anos. Não cabe, neste estudo, avaliar a validade do sorteio para preenchimento de vagas ou a eficácia do processo seletivo, como não é foco desta pesquisa, isso pode, apenas, ser elencado como um tópico para futuros estudos.

No entanto, observamos que em relação às outras treze turmas de Algoritmos do TSPI, a 1M1 – 2014/2 obteve o maior índice de evolução, ou seja, os alunos apresentaram melhora das notas na segunda Etapa, aspecto não observado em nenhuma das outras turmas da disciplina. Dos 21 alunos que prosseguiram com disciplina até o final, quatorze conseguiram manter ou obtiveram uma evolução de suas notas, mostrando que 66,67% desses alunos, aparentemente, mantiveram-se interessados pelos conteúdos da disciplina e responderam positivamente aos estímulos propostos pelo professor.

Para contrapor essa informação, de que esse número referente à evolução é positivo, investigamos sobre isso nas outras turmas de Algoritmos do curso do TSPI do Ifsul, nas quais não se fez uso de qualquer ferramenta gráfica de apoio. Observamos que, em relação às outras treze turmas de Algoritmos do TSPI, a média calculada entre os alunos que conseguiram manter ou elevar suas notas na Etapa II foi de 34,70%. Considerando que o percentual dessa turma é de 66,67 %, observamos um aumento significativo da evolução das notas da etapa II. Como a complexidade do conteúdo aumenta na segunda etapa, essa estabilidade e evolução percebidas com relação às outras turmas já ministradas são consideradas positivas.

Outro ponto importante para se refletir sobre os resultados apresentados, quando observamos a turma como um todo, diz respeito à persistência dos alunos na realização das atividades. A turma, no início do semestre, era formada por trinta alunos, mas, no momento em que começou a aplicação da ferramenta, a turma estava com 26, e

observamos que entre os que começaram a utilizar a ferramenta somente cinco alunos (A_{20}, A_{21}, A_{22}, A_{23}, A_{24}) finalizaram todos os passos das atividades, ou seja, somente 19,23% da turma (Figura 36). Percebemos com esses dados, certa falta de motivação para prosseguir com as atividades propostas, as dificuldades parecem não serem encaradas pelos alunos como desafios e, sim, como motivos de frustração. Portanto, começam a surgir suposições, dúvidas sobre o que está por traz dessas dificuldades e desses desinteresses.

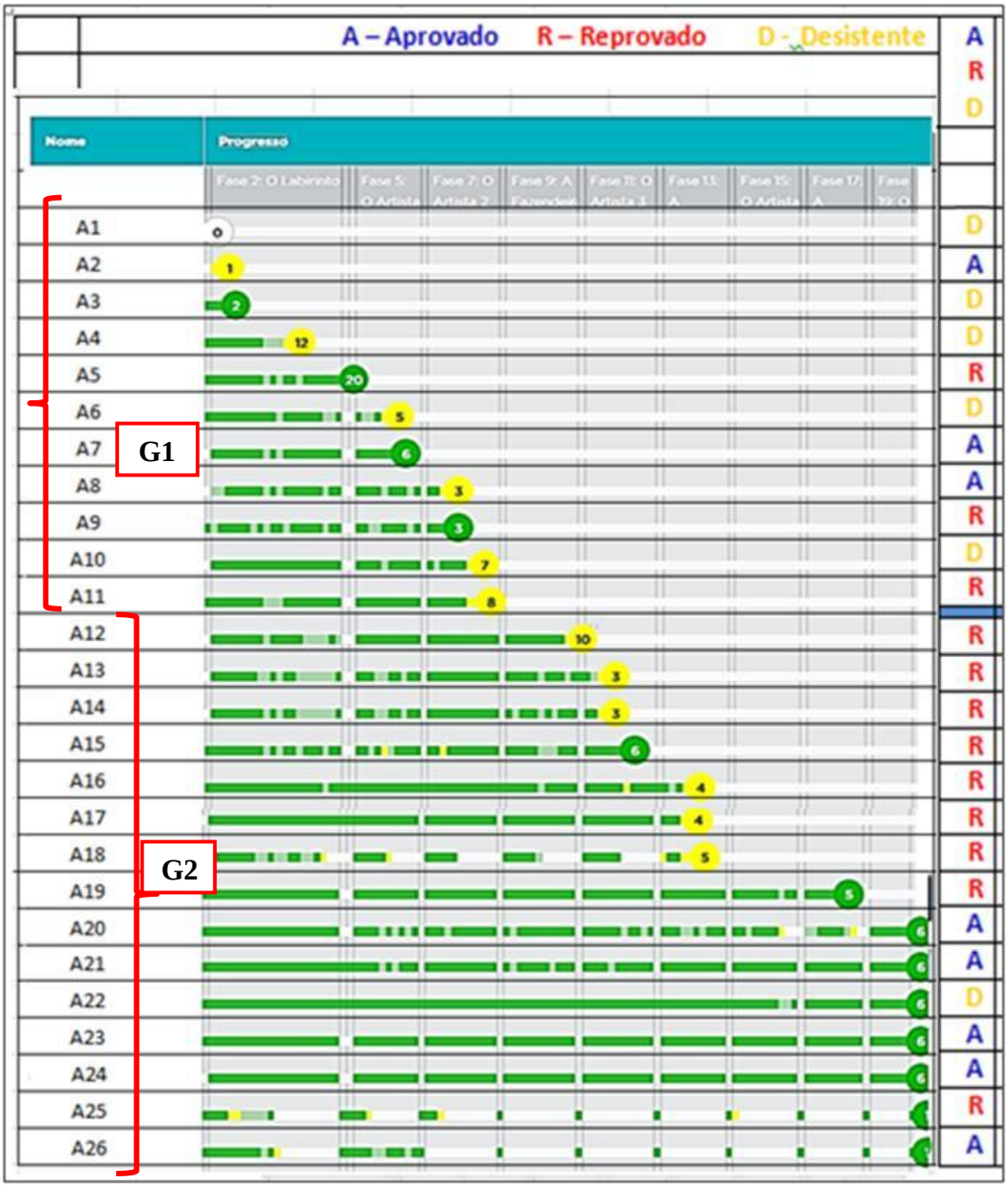
Para uma melhor avaliação, nessa análise, dividimos os alunos em dois grupos. Dos 26 matriculados na disciplina que começaram a utilizar a ferramenta e chegaram até 50% das atividades propostas, chamamos de Grupo 1 (G1), foram onze alunos (A_1 até A_{11}), representando 42,3% do total. No segundo grupo, estão os quinze alunos (A_{12} até A_{26}) que participaram de mais de 50% das atividades propostas, chamado de Grupo 2 (G2), representando os 57,7% (Figura 36).

Dos alunos do G1, somente sete (A_5 até A_{11}) conseguiram passar da primeira atividade proposta pela ferramenta (fase2), e quatro alunos (A_1 até A_4) ficaram ainda nos primeiros passos dessa atividade. Dos alunos do Grupo G2, somente cinco alunos (A_{20}, A_{21}, A_{22}, A_{23}, A_{24}) finalizaram todos os passos das atividades, ou seja, 19,23% da turma conseguiram completar, como ilustrado na Figura 36.

Observando os alunos (G1 e G2) que pararam sem finalizar os passos da maneira estabelecida pelo *software*, percebemos que entre os 26 alunos, treze (A_2, A_4, A_6, A_8, A_{10}, A_{11}, A_{12}, A_{13}, A_{14}, A_{16}, A_{17}, A_{18}, A_{25}) pararam por não conseguir realizar corretamente o passo proposto, (ilustrados na Figura 36 com bolinhas em amarelo). Somente um deles (A_{25}), entre os treze, seguiu adiante tentando executar outros passos das próximas atividades, deixando passos incompletos ou vazios. Essa observação começa a dar indícios sobre as dificuldades dos alunos.

Outros sete alunos (A_3, A_5, A_7, A_9, A_{15}, A_{19}, A_{26}), pertencentes ao G1 e G2, (ilustrados na Figura 36 com bolinhas em verde), mesmo com alguns erros no processo, finalizaram corretamente o passo em que pararam no *software*, mas não seguiram adiante. Portanto, não é possível afirmar que esses não apresentem dificuldade. Observamos também que, entre os sete alunos, somente dois deles (A_{25} e A_{26}) seguiram visualizando os passos das próximas atividades.

Figura 36 - Progresso dos alunos do Grupo G1 e G2



Fonte: da pesquisa.

Entre os alunos que finalizaram ou pararam na ferramenta com erro na construção da atividade, a grande maioria evadiu na disciplina, conforme aparecem marcados na Figura 36 com D (A_1, A_3, A_4, A_6, A_{10}, A_{22}.). Observamos que entre esses alunos desistentes, somente um (A_{22}) conseguiu realizar todas as atividades, sendo que esse aluno desistiu, alegando preferir cursar novamente a disciplina.

Com a percepção dessa falta de motivação dos alunos em prosseguir com as atividades propostas faz-se necessário compreender esse contexto, buscando entender a

relação estabelecida e a interação entre os alunos, o professor, e o meio em que a aprendizagem se desenvolve.

Segundo Costa (2013), a motivação tornou-se um problema para a educação e sua ausência representa queda de qualidade nas tarefas de aprendizagem. A autora destaca que, "Motivação pode ser entendido como aquilo que move uma pessoa ou que a põe em ação ou a faz mudar o curso da ação. A motivação tem sido entendida ora como um fator psicológico, ou conjunto de fatores, ora como um processo" (p. 2). Ainda faz referência aos efeitos da motivação, citando que:

> Os efeitos da motivação são observados no esforço por parte do aluno em se engajar no processo de aprender, enfrentando as tarefas desafiadoras, que por sua natureza, cobram maior empenho e perseverança. Mais ainda, a qualidade do investimento pessoal do estudante implica no uso de estratégias de aprendizagem e de gerenciamentos de recursos de maneira flexível. Os efeitos imediatos da motivação na aprendizagem podem também ser visualizados na escolha, na perseverança e no envolvimento de qualidade, que conduz a um resultado final quais seja a construção de conhecimentos e habilidades (COSTA, 2013, p. 3).

Costa (2013) descreve dois tipos de motivação: a intrínseca e a extrínseca. A intrínseca como uma característica natural dos seres humanos que envolvem o interesse individual para pôr em prática suas capacidades, buscando e alcançando desafios. Quando o envolvimento por parte do aluno acontece de maneira espontânea e a participação na atividade é a principal recompensa, não necessitando de pressões externas ou prêmios por seu cumprimento. Já a motivação extrínseca pressupõe realizar tarefas em função de fatores externos, como diplomas ou dinheiro. A melhor forma de identificar a orientação motivacional de um indivíduo é questionar se a pessoa exerceria o mesmo trabalho sem recompensas ou se não houvesse possibilidade de algum tipo de prêmio ou punição por não o fazer.

Nesse sentido, Eccheli (2008) traz que o professor como organizador da situação de aprendizagem, pode influenciar o nível de motivação dos alunos por meio da determinação das atividades propostas, das formas de avaliação e informações sobre o desempenho nas atividades realizadas. A autora explica sobre essas atividades, mencionando que:

> Se o professor quiser promover a motivação, deve planejar tarefas adequadas ao aluno. "Adequada" aqui significa aquela que "oferece perspectivas de êxito com esforço razoável". Se a tarefa é difícil demais para o aluno, não será possível estabelecer metas que sejam razoavelmente atingíveis, e não será possível atribuir o fracasso à falta de esforço, que estará diretamente relacionada à falta de capacidade, gerando sentimentos de incompetência, insegurança, ansiedade e frustração; o que em cadeia acaba afetando a motivação intrínseca. Por outro

> lado, a realização de desafios mais fáceis favorece a percepção de auto eficácia nas fases inicias da aquisição de novas habilidades, mas posteriormente, se for mantido o baixo nível de dificuldade das tarefas apresentadas, elas deixam de representar um desafio, podendo trazer prejuízos à motivação (ECCHELI, 2008, p. 204).

As colocações da autora refletem situações de motivação extrínseca que não favorecem a aprendizagem. Sobre o uso de recompensas em situação de aprendizagem, Eccheli (2008) aponta como resultado a possibilidade de surgir uma mentalidade interesseira, que o autor descreve como sendo situações em que os alunos buscam realizar eficientemente o mínimo necessário para conseguirem as recompensas, sem valorizarem a atividade em si, nem aspirarem a uma autêntica compreensão ou apresentarem um trabalho de qualidade.

Por meio dessas reflexões sobre motivação, apresentadas por Eccheli (2008) e Costa (2013), faz-se pertinente repensar em como a ferramenta apresentou as atividades e como a professora planejou as atividades no decorrer do semestre. Se essas atividades estariam adequadas aos alunos, se essa lista de exercício (sempre usada) não acaba por gerar sentimento de incompetência, frustração e ansiedade. Reflexões que nos remetem aos conceitos da Teoria das Situações Didáticas, pois segundo Almouloud (2007), essa teoria busca criar um modelo da interação entre o aprendiz, o saber e o meio em que o aprendizado deve se desenrolar. Pela proximidade dos conteúdos de algoritmos e da matemática, acreditamos que seja possível estabelecer a relação nesse contexto da pesquisa com os conceitos da Teoria das Situações Didáticas. Isso porque, para Almouloud (2007), o objeto central de estudo nessa teoria não é o sujeito cognitivo, mas a situação didática na qual são identificadas as interações estabelecidas.

Por meio da aplicação da ferramenta escolhida nesta pesquisa, surgiram reflexões importantes que vão além da aplicação de um *software*, principalmente pela observação do comportamento dos alunos e do professor durante esse processo. Tais reflexões apontaram para um conceito importante originado da Teoria das Situações Didáticas, o contrato didático, o qual analisa as relações que se estabelecem entre professores e alunos, mesmo que de forma implícita, e, ainda, a influência desse no ensino da matemática.

Almouloud, baseado nas afirmações feitas por Brousseau, define contrato didático como "o conjunto de comportamentos específicos do professor esperado pelos alunos, e o conjunto de comportamento dos alunos esperado pelo professor" (2007, p. 89). Cabe, nesse ponto da discussão, repensar os papéis e as responsabilidades dos alunos e professor da disciplina de Algoritmos, como: se o professor soube estabelecer e deixar claras as

cláusulas desse contrato; se essas foram respeitadas pelos alunos, se o contrato foi renegociado e se o professor soube fazer as intervenções necessárias duramente a aplicação da ferramenta para manter o interesse dos alunos. Almouloud (2007) ainda destaca que "um contrato didático mal administrado, por parte do professor ou do aluno, pode ser fonte de dificuldades para aprendizagem de novos conhecimentos matemáticos" (p. 90).

Portanto, o que observamos foi que os alunos não aderiram ao contrato, e às cláusulas indicadas por nós, como professora. Cabe, então, repensarmos a importância de constituir essas cláusulas e estabelecer estratégias para fazer com que os alunos concordem com o contrato didático.

5.3.2 Análise do processo de utilização da ferramenta

Por meio desse processo de análise de dados, verificamos que o papel do *software* utilizado, aos poucos, foi mudando, pois, além de ser uma ferramenta para aplicar e ampliar os conhecimentos abordados na disciplina, tornou-se também um instrumento de diagnóstico, por possibilitar identificar dificuldades conceituais da construção de algoritmos, que até então não havíamos, no papel de professora-pesquisadora, observado. Com o uso dos blocos gráficos, disponíveis na ferramenta, ficou mais visível o processo utilizado pelo aluno para chegar ao resultado final, pois nem sempre um resultado final correto é realizado por meio de uma operação mental inteligente. Nesse sentido, é importante analisar as estratégias realizadas mentalmente pelos alunos ao resolver uma determinada tarefa, para, então, ser possível identificar as dificuldades e potencialidades dos processos de ensino e de aprendizagem. Em relação às potencialidades, observamos que o uso da ferramenta auxiliou alguns alunos, pois comandos que, inicialmente, não conseguiram utilizar, passaram a executar corretamente.

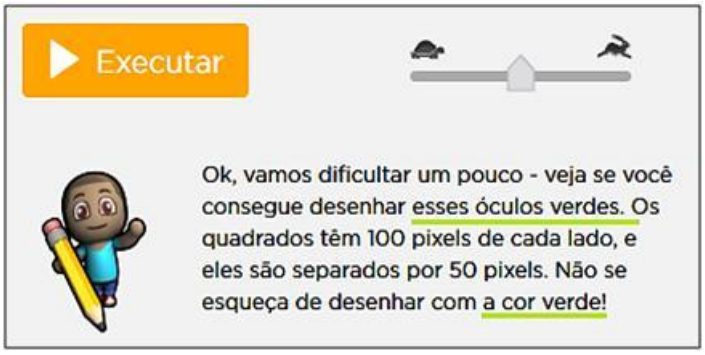
Na análise do processo vivenciado pelos alunos na utilização do *Blockly*, identificamos quatro categorias: interpretar enunciados; comandos, posicionar-se em relação ao objeto e criar representações livres.

a) Interpretar enunciados

Uma observação relevante a ser elencada trata da compreensão dos enunciados das atividades de algoritmos, pois percebemos que grande parte dos alunos não considerou as

condições estabelecidas nos enunciados da ferramenta. Observando especificamente o exemplo do passo 7 da Atividade 5, que solicita que os óculos sejam desenhado na cor verde, constatamos que dos dezoito alunos que concluíram esse passo, somente quatro construíram na cor indicada, o que equivale a 22,23 %. Isso mostra a desatenção dos alunos, pois o enunciado conforme mostra a Figura 37, dá ênfase ao requisito da cor.

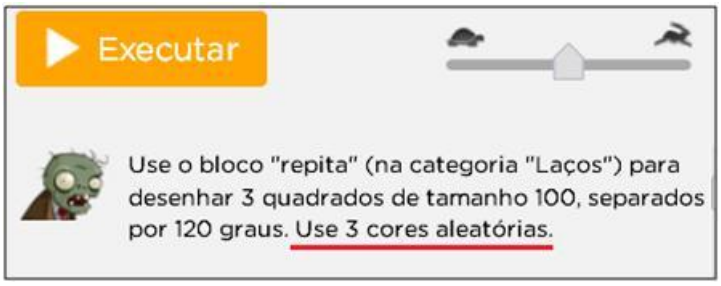
Figura 37 - Enunciado do passo 7 da atividade 5



Fonte: da pesquisa.

Outro ponto que mostra tal desatenção aos enunciados é que quase todos os alunos ignoraram a frase “Use três cores aleatórias”, contida no passo 3 da Atividade 11. A grande maioria não selecionou o comando “definir cor aleatória”, para construir cada quadrado.

Figura 38 - Enunciado do passo 3 da atividade 11



Fonte: da pesquisa.

A seguir, apresentamos dois exemplos, o primeiro em que o aluno (A_{16}) não considerou a informação sobre a cor e o segundo, no qual o aluno (A_{17}) atendeu à tal informação.

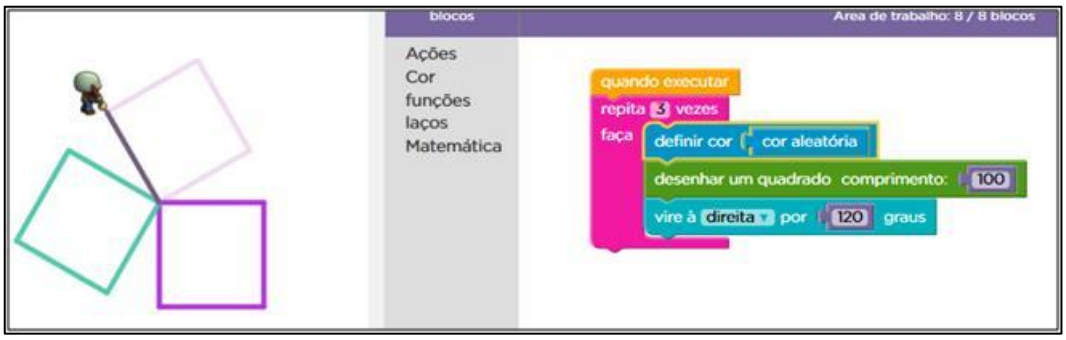
Figura 39 – Resolução do passo 3 da atividade 11 sem cor aleatória (A_{16})

Fonte: da pesquisa.

Mesmo parecendo um simples detalhe, o comando de "definir cor aleatória", demonstra a dificuldade dos alunos em extrair os dados de determinado enunciado, e de prestar atenção nesses detalhes. A diferença é que o aluno que prestou atenção à indicação da cor solicitada construiu cada um dos quadrados com uma cor diferente (Figura 40).

.

Figura 40 - Resolução do passo 3 da atividade 11 com cor aleatória (A_{17})



Fonte: da pesquisa.

A questão aqui não é avaliar o quão importante era o fato de esses quadrados estarem traçados de forma colorida ou em preto e branco, mas, sim, avaliar a capacidade dos alunos de prestar atenção aos dados dos enunciados, pois estava bem claro na instrução do passo 3 da Atividade 11, "Use 3 Cores aleatórias" (Figura 40).

Nesse ponto, faz-se necessária uma reflexão sobre os enunciados dos algoritmos das listas de exercícios propostas aos alunos, pois Gauthier et al. (2013) assinala a relação positiva e significativa entre a clareza da apresentação e o bom êxito dos alunos. Segundo o autor, os professores que dão instruções claras, explícitas, redundantes e compreendidas por todos os alunos levam esses a apresentarem uma maior aplicação nas atividades

individuais. Ele define, ainda, que apresentar claramente significa enfatizar os aspectos importantes do conteúdo; utilizar exemplos para explicar; avaliar a compreensão paulatinamente. Salienta, também, que "a clareza das perguntas formuladas pelos professores é um fator importante a ser considerado. De fato, ocorre de os alunos serem incapazes de responder às perguntas que lhe são feitas porque elas são ou demasiado vagas ou demasiado ambíguas" (GAUTHIER et al., 2013, p. 224).

O funcionamento da atenção explicada pela abordagem de Vygotsky, descrita por Oliveira (1992), dá conta de que, inicialmente, essa é baseada em mecanismos neurológicos inatos, mas vai, gradualmente, sendo submetida a processos de controle voluntário, em grande parte fundamentada na mediação simbólica. A autora menciona que o sujeito será capaz de dirigir voluntariamente sua atenção para os elementos do ambiente em que ele considerar relevante. Essa relevância está relacionada ao significado que o indivíduo constrói ao longo do seu desenvolvimento pela interação com o meio em que vive.

Se o sujeito dirige sua atenção para elementos que considera relevante, é importante refletir sobre as atividades propostas durante as aulas da disciplina, esclarecendo se propiciam o desenvolvimento da capacidade de prestar atenção intencionalmente, capacidade tão esperada por professores. Os enunciados dessas atividades seriam claros o suficiente para que o aluno se sinta seguro em iniciar o desenvolvimento de um Algoritmo?

Ainda sobre a compreensão dos enunciados das atividades de algoritmos, podemos relacionar com a 1° fase apresentada por Polya (1995) como sugestão para resolver um problema, pois, segundo o autor, a compreensão do problema é uma das fases de maior importância. O autor afirma que, pela incompreensão do problema, "alguns alunos atiram-se ao cálculo e ao desenho sem qualquer plano ou idéia geral; outros esperam desajeitadamente que surja alguma idéia e nada fazem para expressar sua aparição" (p.57).

Da mesma forma como Polya, percebemos a ansiedade dos alunos em "atirar-se" à construção dos algoritmos sem uma compreensão efetiva dos enunciados. Assim, torna-se evidente a necessidade de repensar os tipos de atividades propostas na disciplina, buscando esclarecer se essas não estariam provocando esse estado emocional nos alunos. Quanto à execução do plano, o defeito anunciado por Polya (1995) é o desleixo e a falta de paciência para verificar cada passo. O autor afirma que "o aluno contenta-se em obter uma resposta, põe de lado o lápis e não se espanta com os resultados, por mais disparatados que eles forem" (p. 59).

Assim, também ocorre na disciplina de Algoritmos, pois ao não entender a causa de o programa mostrar valores absurdos na tela, os alunos preferem chamar o professor para ajudá-los a identificar o motivo, revelando "falta de paciência" para resolver problemas que surgem no processo de aprendizagem.

Portanto, por meio dessa análise da capacidade dos alunos de prestar atenção aos dados dos enunciados, e, com base nas opiniões dos autores mencionados, torna-se necessária a reflexão acerca dos enunciados dos algoritmos nas listas de exercícios propostas aos alunos, bem como a respeito dos tipos de exercícios escolhidos na disciplina.

b) Comandos

Comando também denominado de instrução, segundo Lopes e Garcia "indica a um computador uma ação elementar a ser executada" (2002, p. 2). Portanto, comandos são instruções que definem as ações a serem executadas para resolver determinado problema. Na ferramenta gráfica utilizada, esses comandos estão representados pelos blocos que podem ser selecionados e arrastados para resolver os passos propostos nas atividades.

Observamos que a visualização dos objetos de maneira gráfica mostrou-se útil para ajudar os alunos a identificar a sequência de comando necessário para resolver determinado problema. Essa mesma percepção foi vista para os desvios definidos pelas condições dos comandos de "se" e "senão", porém, notamos que muitos apresentaram dificuldades em agrupar tais comandos, de perceber o que pode ser selecionado uma única vez e inserido em um comando de laço de repetição.

A seguir, apresentamos nossas percepções sobre os comandos utilizados pelos alunos durante a aplicação da ferramenta na turma.

Definir sequências de comandos

Podemos citar que o uso de ferramenta gráfica proporcionou meios para a turma (1M1-2014/2) elaborar um plano de ação para resolver determinados desafios. Observamos que a sequência de comandos para resolver os primeiros passos propostos, na sua grande maioria foi definida de maneira correta. Uma vez que uma das dificuldades apresentadas com a disciplina, proferida pelos alunos na turma era de pensar em como elaborar uma solução, a ferramenta os auxiliou na criação dos passos para resolver os desafios dos problemas propostos por ela.

Essa capacidade de elaborar um plano de ação é definida por Oliveira (1992, p. 26), com base nas concepções de Vygotsky de "funções psicológicas superiores ou processos mentais superiores", que nos remetem ao conceito de mediação e seus elementos mediadores: os instrumentos e os signos. Portanto, os instrumentos e os signos são os elementos do ambiente humano carregado de significado cultural, os mediadores entre os indivíduos e o mundo.

Para resolver os passos apresentados pela ferramenta, o aluno deve ser capaz de planejar o percurso e as ações a serem realizadas. No exemplo do enunciado do passo 20 da Atividade 2, solicitava-se o percurso do boneco até o girassol (Figura 41). Para elaborar esse trajeto era necessário utilizar os blocos de comando que a ferramenta já havia apresentado nos passos anteriores.

Figura 41 - Passo 9 da Atividade 2 (A_{10})

Fonte: da pesquisa.

Portanto, o que observamos é que esses passos foram resolvidos por um dos alunos (A_{10}) de maneira correta, pois conforme ilustrado na Figura 41, os alunos necessitavam usar somente três blocos de comandos, o "avance", o "vire à esquerda" ou o "vire à direita". Podemos afirmar que a ferramenta, quanto à apresentação dos passos das atividades, proporcionou os desafios de maneira que os alunos compreendessem e conseguissem definir a sequência de comandos. Essa responsabilidade, já mencionada por Polya (1995), é atribuída ao professor no que tange à escolha dos problemas propostos aos alunos. O autor, em seus estudos sobre os métodos de resolução de problemas em

matemática, já mencionava a responsabilidade do professor na escolha dos problemas propostos para os alunos, quando afirma que, "O problema deve ser bem escolhido, nem muito difícil nem muito fácil, natural e interessante, e um certo tempo deve ser dedicado à sua apresentação natural e interessante" (p. 4).

Também compreendemos a ferramenta utilizada como um instrumento mediador para estimular os alunos a elaborar um plano de ação e resolver os problemas propostos, pois foi apresentando os desafios e, ao mesmo tempo, dando pistas para atingir a zona de desenvolvimento proximal.

Utilizar comandos de seleção

Observamos que a maioria dos alunos conseguiu aplicar o comando "se" e suas condições, acreditamos que a visualização dos objetos de forma gráfica tenha facilitado esse processo. Notamos que nenhum dos alunos desistiu de continuar a usar a ferramenta nos passos em que estava utilizando tal instrução. Nesse sentido, podemos afirmar que a ferramenta se mostrou produtiva para essa turma, quanto à aplicação de conteúdos já estudados na disciplina que envolve o comando "se".

Em Algoritmos, o comando "se", também denominado de seleção simples, apresenta como variação a seleção composta, o que denominamos de "se – senão". Esse tipo de seleção é utilizado para analisar o que será executado quando a condição for verdadeira, assim como o que será executado quando a condição for falsa.

A seguir, ilustraremos dois exemplos de um dos alunos (A_{24}), o primeiro em que faz uso de dois comandos de seleção simples, e o segundo que utiliza seleção composta. No primeiro exemplo, o passo faz uso de dois comandos "se", sendo que o primeiro verifica "se houver um buraco" para acionar a instrução "preencha1", o segundo comando verifica "se há um monte" e, então, aciona a instrução remova1 (Figura 42).

Figura 42 - Exemplo do comando "se" (A_{24})

Fonte: da pesquisa.

O segundo exemplo faz uso do comando de seleção composta, o “se - senão” em que verifica “se há um monte” para acionar a instrução “remova1”, “senão” (caso-contrário) será acionada a “preencha1” (Figura 43).

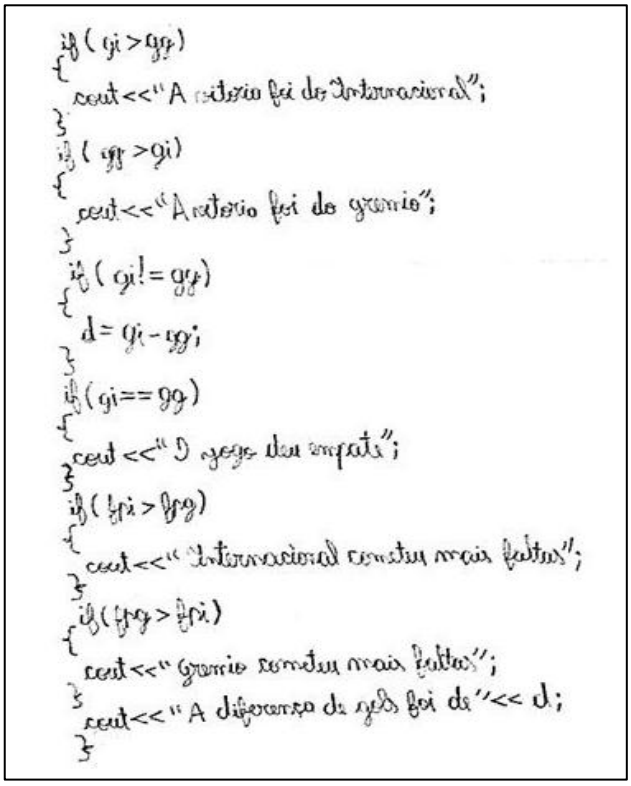
Figura 43 - Exemplo do comando "se - senão " (A_{24})

Fonte: da pesquisa.

Assim como para o comando “se”, a ferramenta mostrou-se propícia para o comando “se - senão”, pois observamos que a maioria dos alunos conseguiu executar os passos que envolviam o uso desses comandos. Com essa constatação, surgiu o questionamento: se o *software* propiciou o entendimento dos comandos “se” e “se – senão” ou os alunos dessa turma já teriam aprendido durante as aulas anteriores?

Para isso, voltamos a analisar as provas da etapa I, período em que foi abordado esse conceito e observamos que tais comandos foram utilizados de maneira correta pela maioria dos alunos. Verificamos que, dos 27 alunos da turma que finalizaram a etapa I, somente cinco (18,5%) apresentaram algum tipo de dificuldade com o uso desses comandos. A seguir, ilustramos dois exemplos de códigos construídos pelos alunos durante as provas da etapa I. Ressaltamos que os alunos utilizam o comando "se" em inglês "if" e o "se – senão" como "if – else", como é exigido pelas linguagens de programação.

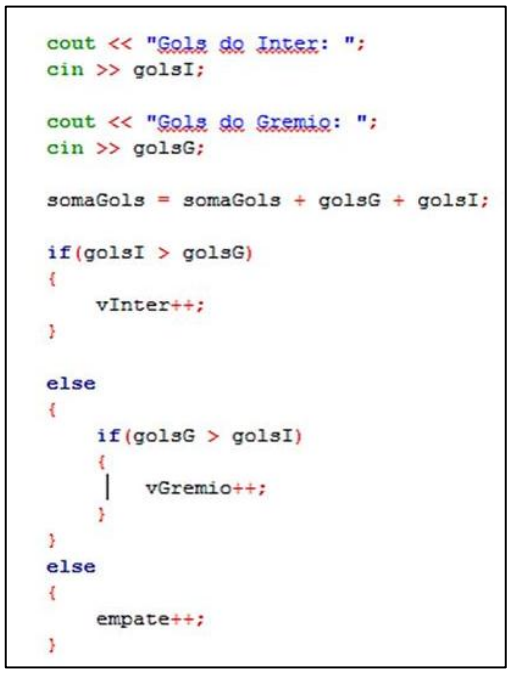
Figura 44 - Exemplo do comando "se" na prova (A_{24})



Fonte: da pesquisa.

No primeiro exemplo (Figura 44), o aluno (A_{24}) faz uso de comando de seleção simples para analisar o número de gols e o de faltas de um grenal, já no segundo exemplo (Figura 45), o aluno (A_{22}) utiliza o comando de seleção composta para resolver o mesmo exercício e verificar os gols do grenal.

Figura 45 - Exemplo do comando "se - senão " na Prova (A_{22})



```
cout << "Gols do Inter: ";
cin >> golsI;

cout << "Gols do Gremio: ";
cin >> golsG;

somaGols = somaGols + golsG + golsI;

if(golsI > golsG)
{
    vInter++;
}

else
{
    if(golsG > golsI)
    {
    |   vGremio++;
    }
}
else
{
    empate++;
}
```

Fonte: da pesquisa.

Assim, após a verificação das provas da etapa I, não é possível afirmar se a ferramenta conduziu os alunos para uma melhor compreensão dos comandos “se” e “se – senão”, ou se, quando fizeram uso da ferramenta, já teriam compreendido tais conceitos.

Agrupar comandos

Mais um aspecto observado durante a análise foi a dificuldade demonstrada pelos alunos de agrupar e classificar comandos. Alguns alunos não conseguiram identificar comandos que estavam sendo escritos repetitivamente e perceber que esses poderiam ser agrupados, reorganizados, buscando um melhor desempenho do código (algoritmo). A capacidade de perceber essa possibilidade e resolver determinada situação de maneira mais inteligente, na área de computação é muito valorizada, pelo fato de que os códigos ficam menores, requerem menos processamento e são mais fáceis de alterações.

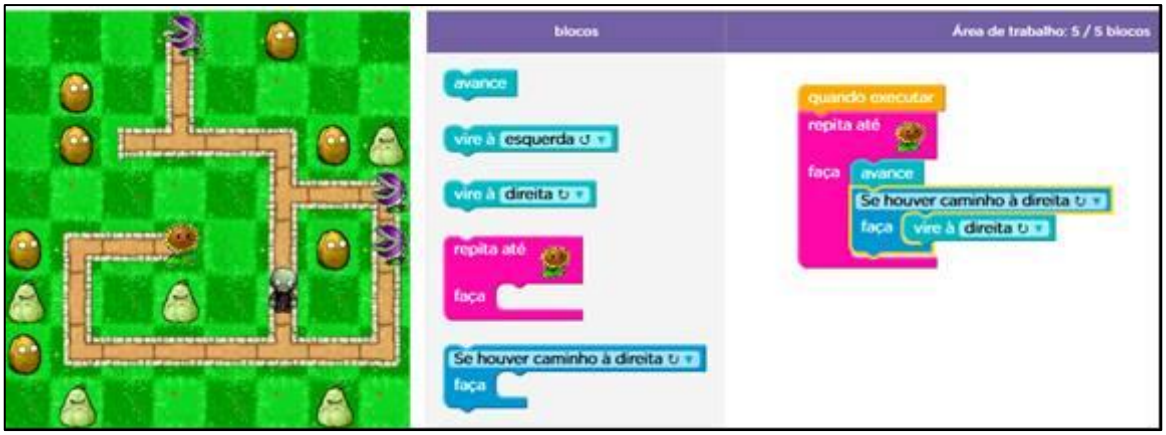
Na ferramenta gráfica, os comandos são representados pelos blocos que podem ser selecionados; no exemplo ilustrado na Figura 46, o aluno (A_{20}) não conseguiu identificar que poderia ter agrupado os comandos “avance” e “vire à direita”, ou seja, não conseguiu estabelecer esse tipo de relação, utilizou dez “avance” para resolver o passo proposto pela atividade.

Figura 46 - Passo 17 da atividade 2 (A_{20})

Fonte: da pesquisa.

Essa maneira de resolver não está totalmente errada, mas demanda muito mais processamento, pois será necessário executar dezesseis comandos, enquanto a resolução poderia ter quatro comandos, conforme resposta do colega (Figura 47), que conseguiu estabelecer esse tipo de relação. O colega (A_{23}) conseguiu perceber que o "boneco" executava o mesmo procedimento de avançar e virar à direita, caso houvesse caminho disponível.

Figura 47 - Passo 17 da atividade 2 (A_{23})



Fonte: da pesquisa.

Analisar ambas as maneiras de resolver o mesmo passo da atividade faz repensarmos sobre o tempo que cada aluno necessita para entender os conteúdos. Percebemos que enquanto um dos alunos precisou descrever as dezesseis instruções para

fazer o caminho necessário, o outro conseguiu entender a possibilidade de agrupar e economizar instruções. Geralmente, esperamos que todos aprendam em tempos parecidos, no entanto, quando, em determinada turma, os alunos apresentem essa diferença, precisamos pensar em estratégias que envolvam ambos os tempos.

Ocorrência essa que nos faz avaliar os conceitos do tempo didático da transposição didática apresentados por Pais (2008), referentes à programação de ensino. Essa programação está voltada a cumprir o programa proposto, enquanto o de aprendizagem deve respeitar o tempo necessário de cada sujeito para superar os bloqueios.

Assim, podemos refletir se a forma que ferramenta foi utilizada, de maneira individual e extraclasse priorizou o tempo de aprendizagem ou o tempo didático. Ainda, nesse sentido, conduzindo a questionamentos referentes ao planejamento das aulas da disciplina de algoritmos, indagamos se esse apresenta maior ênfase no tempo didático ou no tempo de aprendizagem.

Observamos, com esse passo da atividade, que cada aluno apresentou uma solução refletindo o seu estágio de aprendizagem, ficando explícita a relação de que alguns precisaram realizar mais atividades para abstrair e identificar quais comandos poderiam estar agrupados para não ser necessário repetir. Como já mencionado, resolver determinada situação com menor número de comandos, na área de computação, constitui prática muito valorizada, pois, devido à complexidade dos programas, pode tornar-se inviável registrar uma solução em um único arquivo.

A dificuldade em classificar e organizar os comandos está relacionada com a compreensão dos comandos de repetição, pois, se o aluno não consegue entender o que pode ser agrupado, também não conseguirá perceber o que pode ser envolvido por um comando de laço. Sobre comandos de repetição será descrito no próximo item, denominado de laços de repetição.

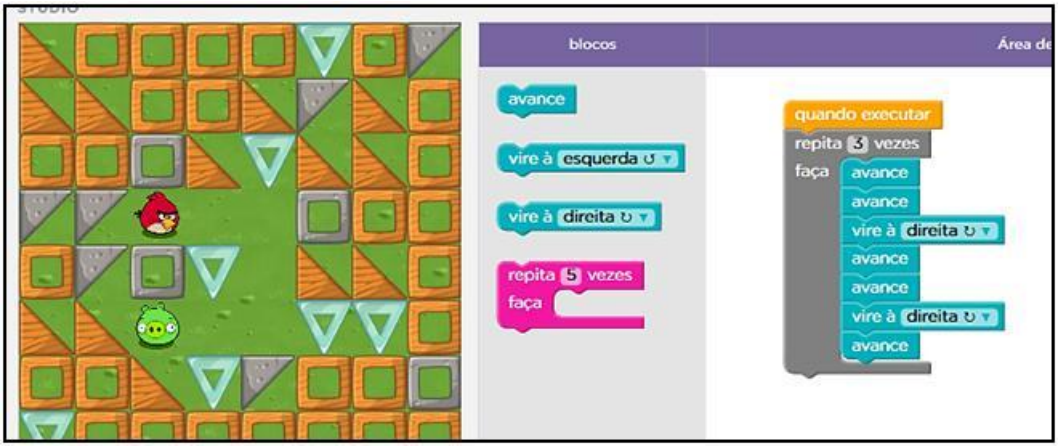
Laços de repetição

Outro aspecto que se revelou como novidade com relação ao uso de laço de repetição diz respeito à maneira como os alunos entendem o seu funcionamento. Apesar de percebermos dificuldades com relação a esse comando, entendíamos que era em relação a definir os passos necessários para resolver o problema em si, ou seja, de estabelecer o plano de ação e determinar as instruções necessárias que estariam inseridas no comando de

laço. Mas observamos que a dificuldade é quanto à compreensão da forma de execução do próprio comando de “laço”.

A dificuldade observada sobre a utilização de laços foi com o uso do comando “repita “n” vezes”, em que a variável “n” significa quantas vezes o laço irá se repetir. Observamos que muitos alunos parecem ignorar o número 3, como no exemplo da (Figura 48) de um dos alunos (A_2), e, assim, definem internamente os comandos necessários para resolver o passo como se não estivessem utilizando laços. Esse fato mostra que os alunos não entenderam que os comandos envolvidos pelo “repita 3 vezes faça” executará três vezes. Ao invés disso, selecionaram os comandos “avance” e “vire à direita” desnecessários.

Figura 48 - Passo 9 da atividade 2 envolvendo laços (A_2)



Fonte: da pesquisa.

A proposta do passo 9 da atividade 2 tratava do movimento para que o personagem (vermelho) chegasse até o (verde), utilizando o comando “repita n vezes faça”. O que constatamos é que os alunos não apresentaram dificuldades para pensar no caminho necessário, mas não souberam realizar esse caminho de maneira menos trabalhosa.

Outro passo semelhante proposto pela ferramenta que fica explícita essa dificuldade dos alunos em entender que precisa repetir “n vezes” diz respeito ao momento em que a proposta era traçar um triângulo utilizando comando de repetição (Figura 49). O mesmo processo se reproduz entre parte desses alunos, o comando “avance por 100 pixels” é selecionado três vezes, ao invés de ser colocado somente uma vez e inserido no comando “repita 3 vezes”. O que, mais uma vez, confirma a ideia de que os alunos não entenderam o que realmente significa o comando de repetição.

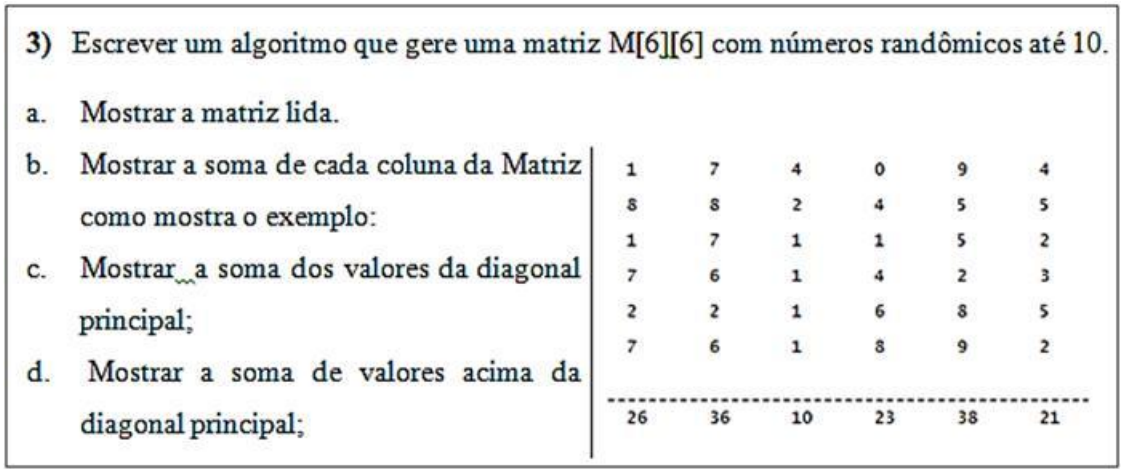
Figura 49 - Passo 4 da atividade 5 envolvendo laços (A_2)

Fonte: da pesquisa.

O conceito de laços de repetição foi abordado ainda na primeira etapa da disciplina, o que evidenciou que, na sua grande maioria, os alunos não entenderam o que é necessário repetir. Nesse sentido, é possível afirmar que o uso do *software*, da maneira em que foi utilizado nessa disciplina, não propiciou a aplicação do comando "repita n vezes".

Nesse estágio, conseguimos visualizar essas mesmas características na questão 3 da prova da etapa II (Figura 50). Para responder a essa questão, um dos alunos (A_2) construiu de forma "manual", ou seja, efetuou a adição de cada coluna de uma matriz, sem fazer uso de comandos de laços, do mesmo modo que definiu na ferramenta todos os comandos para traçar o triângulo.

Figura 50 - Questão 3 da Prova da etapa II



3) Escrever um algoritmo que gere uma matriz M[6][6] com números randômicos até 10.

a. Mostrar a matriz lida.

b. Mostrar a soma de cada coluna da Matriz como mostra o exemplo:

c. Mostrar a soma dos valores da diagonal principal;

d. Mostrar a soma de valores acima da diagonal principal;

1	7	4	0	9	4
8	8	2	4	5	5
1	7	1	1	5	2
7	6	1	4	2	3
2	2	1	6	8	5
7	6	1	8	9	2
26	36	10	23	38	21

Fonte: da pesquisa.

Para adicionar os elementos de cada coluna, o aluno (A_2) especificou de maneira fixa as linhas e as colunas, como mostra o algoritmo da Figura 51. Assim, para realizar essa operação com os valores que estão na coluna "0" da matriz "M[linha][coluna]" definiu

uma variável inteira denominada “c1”. Podemos observar nessa fórmula “c1 = M[0][0] + M[1][0] + M[2][0] + M[3][0] + M[4][0]”, que somente alterou o primeiro índice referente à linha, o segundo da coluna sempre repetiu “0”.

Percebemos que, assim como na ferramenta, o aluno entendeu o processo, o que era necessário fazer para adicionar cada coluna da matriz, mas não soube propor uma solução que envolvesse comandos de laço de repetição, e, sabemos, propor soluções envolvendo esses comandos é um dos procedimentos fundamentais exigidos na disciplina de Algoritmos.

Figura 51 - Código da questão 3 (A_2)

```
#include<iostream>
#include <windows.h>
#include <time.h>

using namespace std;
main()
{
    srand(time(NULL));
    int M[6][6], l, c;

    for(l=0;l<6;l++)
    {
        for(c=0;c<6;c++)
        {
            M[l][c] = rand()%10;
            cout<<M[l][c]<<"\t";
        }
        cout<<"\n\n";
    }
    cout<<"\n\n";

   int c1, c2, c3, c4, c5, c6;
   c1 = M[0][0] + M[1][0] + M[2][0] + M[3][0] + M[4][0] + M[5][0];
   cout<<"Coluna 1: "<<c1<<" ";

   c2 = M[0][1] + M[1][1] + M[2][1] + M[3][1] + M[4][1] + M[5][1];
   cout<<"Coluna 2: "<<c2<<" ";

   c3 = M[0][2] + M[1][2] + M[2][2] + M[3][2] + M[4][2] + M[5][2];
   cout<<"Coluna 3: "<<c3<<" ";

   c4 = M[0][3] + M[1][3] + M[2][3] + M[3][3] + M[4][3] + M[5][3];
   cout<<"Coluna 4: "<<c4<<" ";

   c5 = M[0][4] + M[1][4] + M[2][4] + M[3][4] + M[4][4] + M[5][4];
   cout<<"Coluna 5: "<<c5<<" ";

   c6 = M[0][5] + M[1][5] + M[2][5] + M[3][5] + M[4][5] + M[5][5];
   cout<<"Coluna 6: "<<c6<<" ";
}
```

Fonte: da pesquisa.

Por sua vez, o aluno (A_8) resolveu o mesmo exercício da prova, fazendo o cálculo de cada coluna dentro de laços (Figura 52), e de maneira mais eficiente. Isso não quer dizer que para cada algoritmo só existe uma única maneira de resolvê-lo, mas existem tipos de pensamentos mais elaborados.

Figura 52 - Código questão 3 (A_8)

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
using namespace std;
main()
{
    srand(time(NULL));
    int M[6][6], SC[6]= {0,0,0,0,0,0}, somaDiag = 0, somaAcimaDiag = 0;
    int l, c;

    for(l=0; l<6; l++)
    {
          for(c=0; c<6; c++)
         {
               M[l][c] = 0;
               M[l][c] = rand()%10;
               cout<<M[l][c]<<"\t";
          }
        cout<<"\n";
    }

    somaDiag = 0;

    for(c=0; c<6; c++)
    {
        for(l=0; l<6; l++)
        {
           SC[c] = SC[c] + M[l][c];
        }
    }
    cout<<"\n";

    for(int i=0; i<6; i++)
    {
             cout<<SC[i]<<"\t";
    }

    cout<<"\n\nSoma da Diagonal Principal: "<<somaDiag;
    cout<<"\n\nSoma acima da Diagonal Principal: "<<somaAcimaDiag;

}
```

Fonte: da pesquisa.

Ainda com relação a laços, outra dificuldade identificada é o comando “Enquanto (verdade)”, um número maior de alunos demonstraram dificuldades com o uso, evidenciando que não o entenderam como um comando de laço, e, sim, como um de seleção. O exemplo ilustrado na Figura 53 mostra que o aluno, mesmo utilizando o

comando “enquanto (há um monte)”, inseriu outro comando, o “repita 4 vezes” para se certificar que os demais comandos “avance” e “remova 1” estariam sendo executados mais vezes.

Essa constatação foi relevante, pois ao explanar esses conceitos, nossa preocupação como professora-pesquisadora sempre foi de explicar o critério de controle que finalizaria o comando do laço. No exemplo citado, que usa o comando “enquanto (há um monte)” (Figura 53), a maior preocupação era de esclarecer em que momento a expressão “(há um monte)” tornar-se-ia falsa. Percebemos que a dificuldade antecede a esse critério de verdadeiro ou falso no controle de laço, pois os alunos demonstraram que não entenderam o “enquanto” como um comando laço, e, sim, como um de seleção. Um comando de seleção que somente verifica se a expressão “(há um monte)” é verdadeira ou não.

Figura 53- Passo 6 da atividade 9 com uso do “Enquanto”

Fonte: da pesquisa.

Com o auxílio da ferramenta, foi possível observar que poucos alunos entenderam que os comandos incluídos em um laço, executarão enquanto o critério for verdadeiro, “Enquanto (verdade)”, o que mostra a necessidade de repensarmos as analogias utilizadas durante as explicações desses. Geralmente, eram utilizados em aula exemplos, tais como, “Enquanto (é dia)”, “Enquanto (resposta == S)”, “Enquanto (sexo <> F)”. Ou ainda, se existe a necessidade do uso de analogias, pois Almouloud (2007), ao se referir aos efeitos do contrato didático e seus fatores para a aprendizagem, aponta o uso abusivo de analogia

como uma das atitudes ou práticas que é considerada como verdadeira ruptura do contrato didático. Para o autor, prática que favorece a memorização, sendo útil para fazer compreender o significado de um conceito, mas que, quando utilizada de maneira abusiva, pode descaracterizá-lo.

Nesse sentido, é necessária a reflexão sobre esses comandos, pois analisando a produção das atividades dos alunos no ambiente gráfico, percebemos que esses não entenderam o conceito de laço, que, de fato, não têm, generalizados em suas mentes, o que é um laço. Assim, é necessário repensar a apresentação desses comandos para que os seus significados sejam apropriados pelos alunos, relacionando com o significado e o sentido da palavra desses comandos.

Na concepção de Vygotsky, segundo Palangana "o desenvolvimento da linguagem coloca-se como paradigma para explicar a formação de todas as demais operações que envolvem o uso de signos" (2001, p. 102). O autor também acredita que é da relação entre a fala e a inteligência prática ou ainda, da combinação entre o instrumento e o signo que emergem as funções cognitivas superiores. Oliveira também descreve a análise de Vygotsky das relações entre pensamento e linguagem e afirma que:

> A questão do significado ocupa um lugar central. O significado é um componente essencial da palavra, e é ao mesmo tempo, um ato do pensamento, pois o significado de uma palavra já é em si uma generalização. Isto é, no significado da palavra é que o pensamento e a fala se unem em pensamento verbal (1992, p. 48).

Para reforçar a importância do significado das palavras, Oliveira afirma que "do ponto de vista da psicologia, o significado de cada palavra é uma generalização de um conceito. E como as generalizações e os conceitos são inegavelmente atos do pensamento, podemos considerar o significado como um fenômeno do pensamento" (1992, p. 48).

Como o significado das palavras é modificado ao longo da história social e individual e continuam se transformando, Vygotsky distingue dois componentes do significado das palavras: o significado propriamente dito e o sentido. O significado propriamente dito consiste em um núcleo relativamente estável de compreensão da palavra compartilhado por todas as pessoas que a utilizam. O sentido refere-se ao significado da palavra para cada indivíduo (OLIVEIRA, 1992).

Sendo a pesquisadora a própria professora de Algoritmos e refletindo sobre a forma em que apresenta os conceitos da disciplina aos alunos, percebemos que nossa atenção se volta para priorizar a representação e as formas de resolver o algoritmo. São raras as

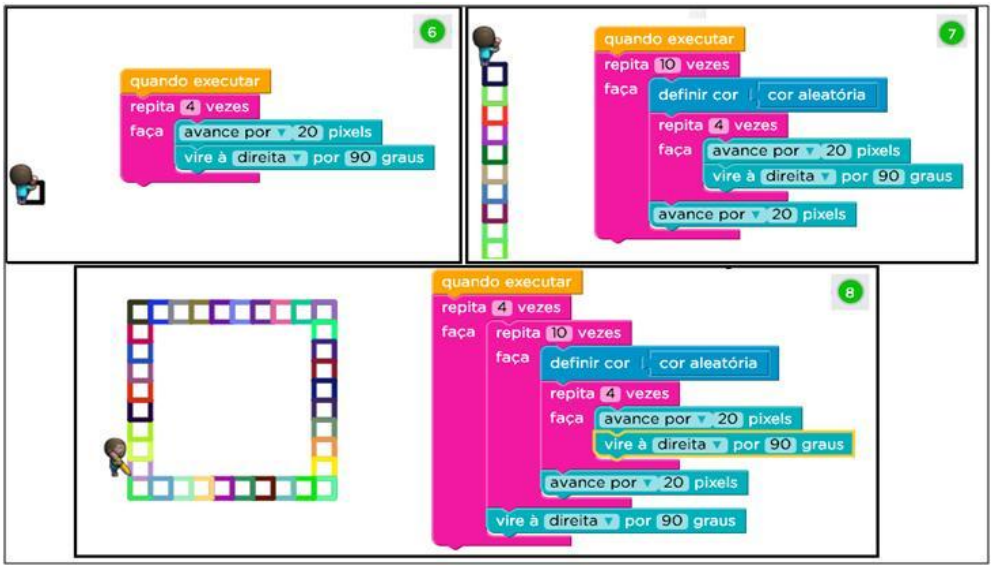
ocasiões em que nos preocupamos com o significado das palavras, com o significado e o sentido de cada um dos comandos utilizados, a função e a forma de execução desses estão sempre em destaque. Isso se justifica pelo entendimento de que não percebíamos que alunos de graduação poderiam ter dificuldades em entender o significado de "palavras" desses comandos.

Portanto, esse entendimento sobre o significado dos comandos que a professora esperava que os alunos já tivessem, podem ser comparados com as "noções paramatemáticas" apresentadas por Chevallard (2000, p. 59), segundo o autor são conhecimentos supostamente obtidos pelo aluno durante a sua formação anterior; noções essas que, geralmente, são apresentadas aos alunos por meio de demonstração sem uma reflexão sobre o seu significado.

Laços encadeados

Outro aspecto observado sobre os conceitos dos quais os alunos manifestaram dificuldades ou conseguiram aplicá-los com o auxílio da ferramenta gráfica foi com o uso do comando "repita x vezes". Observamos que muitos alunos, inicialmente, apresentavam dificuldades, mas superaram com o avanço nos passos das atividades, pois dos quinze alunos pertencentes ao grupo G2, treze deles conseguiram realizar corretamente a sequência de passos, o que evidencia o entendimento do seu uso, conforme ilustrado na Figura 54.

Figura 54 - Evolução dos laços encadeados



Fonte: da pesquisa.

A ferramenta mostrou-se eficiente para o entendimento de laços encadeados, pois ao observarmos a evolução dos passos 6, 7 e 8 da Atividade 7 (Figura 54), que propõe no passo 6 o uso de um "repita 4 vezes" para desenhar um pequeno quadrado. No passo seguinte (7) indica que se construa uma linha vertical de pequenos quadrados, para o que foi necessário o uso do comando "repita 10 vezes", envolvendo o comando "repita 4 vezes", e, assim, traçar dez pequenos quadrados na direção vertical. Por fim, o passo 8 propõe a construção de um quadrado maior, formado de quatro linhas de pequenos quadrados, para o que foi preciso envolver os dois laços já construídos num outro "repita 4 vezes", para então traçar quatro linhas de pequenos quadrados.

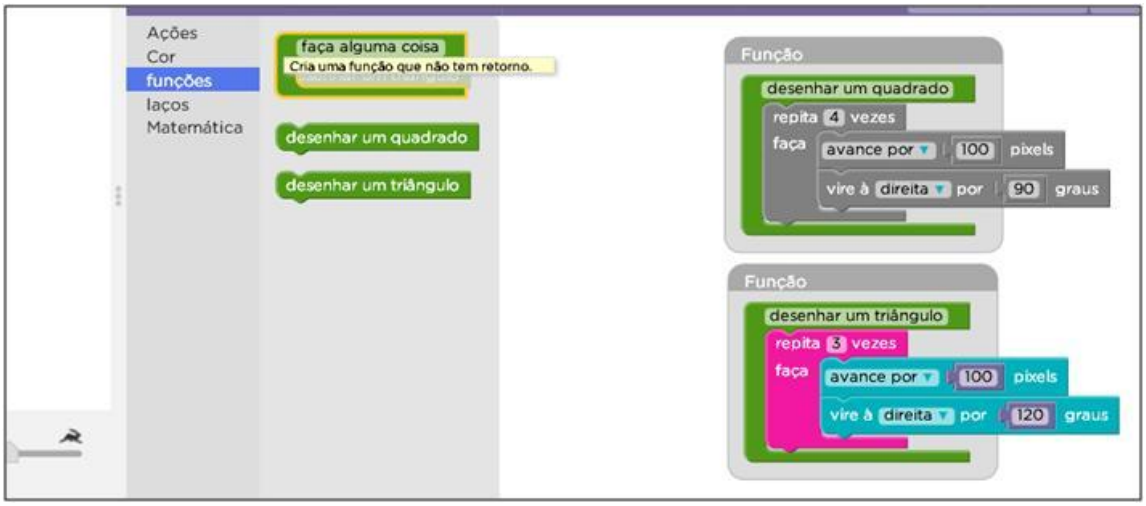
Conforme observamos, os alunos (A_{12}, A_{13}, A_{14}, A_{15}, A_{16}, A_{17}, A_{18} , A_{19}, A_{20} , A_{21}, A_{22}, A_{23} A_{24}) que seguiram realizando as atividades propostas com o uso de laços, aparentemente, entenderam o uso do comando repita de maneira encadeada. Fato que indica que a ferramenta gráfica pode auxiliar os alunos na aplicação de conceitos de laços encadeados.

Essa constatação nos reporta aos conceitos de situação adidática introduzida por Brousseau, que Freitas caracteriza essencialmente por "representar determinados momentos do processo de aprendizagem nos quais os alunos trabalham de maneira independente, não sofrendo nenhum tipo de controle direto do professor relativamente ao conteúdo matemático em jogo" (FREITAS, 2008, p. 44). Ainda sobre situações adidáticas, o autor apresenta que essas representam os momentos mais importantes da aprendizagem, pois o sucesso do aluno nela significa que ele por seu próprio mérito, conseguiu sintetizar algum conhecimento.

Definir Função

Apesar das dificuldades observadas com alguns alunos em agrupar comandos, anteriormente descritas, outro aspecto que observamos no qual a ferramenta mostrou-se eficiente foi para a aplicação do conceito de função. A ferramenta dispõe da opção para separar e reorganizar os comandos e, assim, criar uma função. Entre as categorias de comandos disponíveis (Figura 55), há a de funções e, ao clicarmos nessa, aparecem os comandos de selecionar as já criadas e o "faça alguma coisa", usado para criar uma nova função. Cada função é representada por um quadrado cinza que apresenta como primeira instrução a sua identificação e, após, os comandos que a compõem.

Figura 55 - Passo 2 da atividade 15 propõe Criar Funções

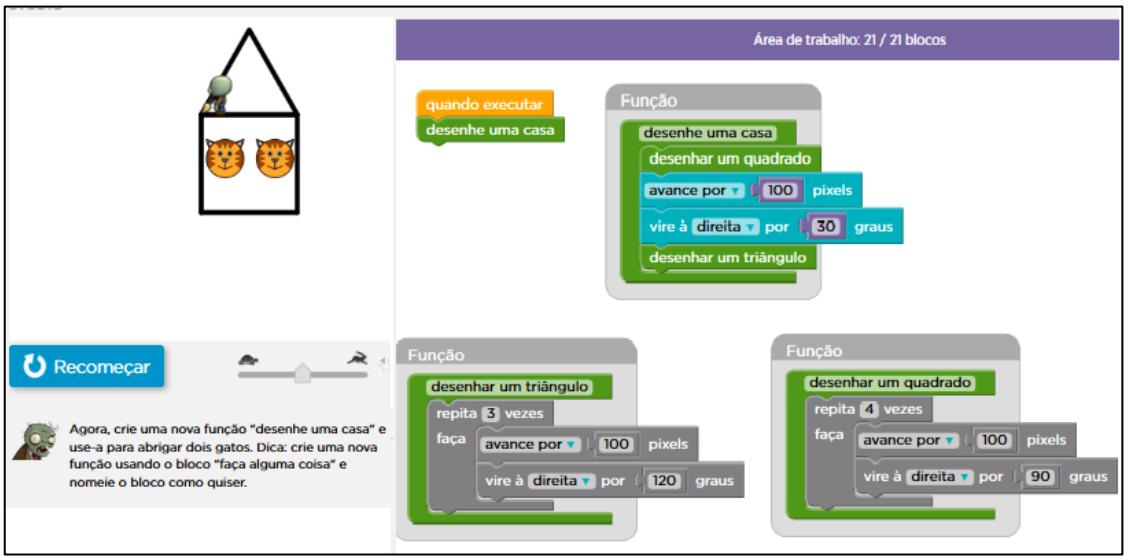


Fonte: da pesquisa.

Portanto definir uma função pode ser relacionado com a atividade de elaborar um plano de ação, que é apresentada por Polya (1995) como uma das fases sugeridas para resolver um problema. Essa consiste em relacionar os dados do problema à pergunta feita e procurar achar uma estratégia para que se possa chegar à solução. O autor também salienta que é importante que o professor auxilie o aluno por meio de discretas sugestões e indagações que conduzam ao caminho certo.

Esse plano de ação é formado pelos comandos necessários para apresentar uma solução aos desafios propostos pela ferramenta. Expomos como exemplo o passo 5 da Atividade 15 (Figura 56), em que ao lançar o desafio para que o aluno criasse uma nova função “desenhar uma casa” e a usasse para abrigar dois gatos, dá a dica de como criar uma nova função: “use o bloco “faça alguma coisa” e nomeie o bloco como quiser”. Ao lançar esse desafio, a ferramenta apresenta duas funções já construídas, a “desenhar um triângulo” e a “desenhar um quadrado”, conduzindo, assim, o aluno à construção correta da sua nova função denominada “desenhe uma casa”.

Figura 56 - Passo 5 da atividade 15 (A_{22})



Fonte: da pesquisa.

Observamos, nesse exemplo da solução ao desafio proposto (Figura 56), que o aluno A_{22} logo abaixo do comando "quando executar" selecionou o comando "desenhe uma casa". Esse comando, por sua vez, é uma função. Ao observarmos essa função percebemos que referenciou outras duas funções a "desenhar um triângulo" e a "desenhar um quadrado".

A função desenhe uma casa proposta pelo aluno (A_{22}) é composta por quatro comandos: o 1° "desenhar um quadrado"; logo após pelo 2° "avance por 100 pixels", necessário pelo fato de que o "boneco" finaliza o traço do quadrado no canto inferior e esquerdo do primeiro gatinho, portanto, era necessário avançar 100 pixels para o boneco posicionar-se corretamente para o início do traço do triângulo; o 3° comando é o "vire à direita por 30 graus" para iniciar no sentido correto o traço do triângulo; e, por fim, o 4° comando "desenhar um triângulo", formando, então, o traço de uma casa. Por meio da observação desse processo construído pelo aluno podemos confirmar a ideia de que ele entendeu o conceito de construir funções, e, assim, a ferramenta para o conceito de função com esse aluno se mostrou eficiente.

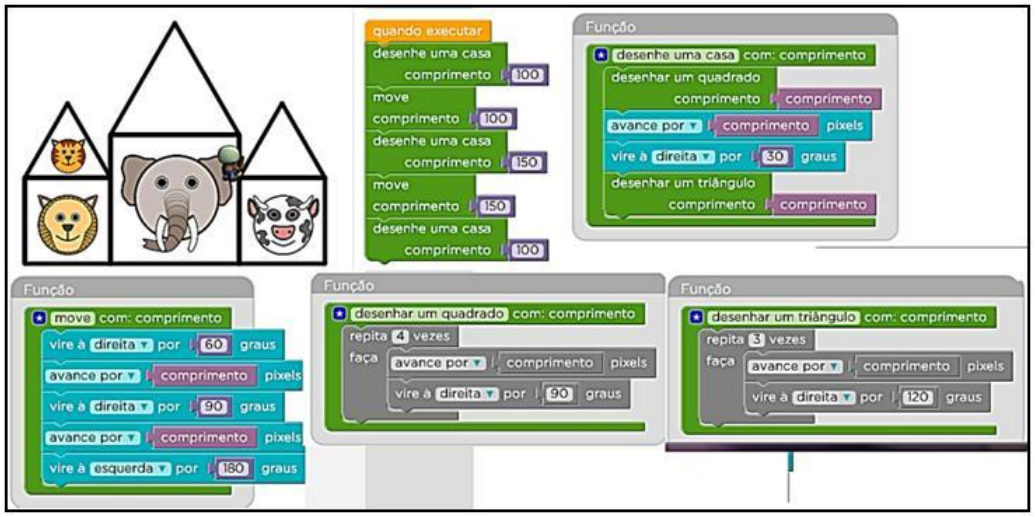
Poucos alunos conseguiram chegar até as atividades que envolviam função, alguns iniciaram e fizeram determinados passos (A_{16}, A_{17}, A_{18}, A_{25}, A_{26}), principalmente os que abrangiam somente a chamada de funções já elaboradas. Somente seis alunos conseguiram realizar praticamente todos os passos que envolviam esse conceito (A_{19}, A_{20}, A_{21}, A_{22}, A_{23}, A_{24}), dos quais, somente dois apresentaram dificuldades com a passagem de parâmetros

para as funções. Com os demais (quatro) alunos, o que observamos foi que a ferramenta os auxiliou a entender a função bem como a passagem de parâmetros dessa.

Para Fiscina, “parâmetros são canais pelos quais se estabelece uma comunicação bidirecional entre um subalgoritmo e o algoritmo chamador. Dados são passados pelo algoritmo chamador ao subalgoritmo, ou retornados por este ao primeiro por meio de parâmetros” (2013, p. 69). Ainda podemos dizer que os parâmetros são variáveis que carregam os valores que desejamos passar como informação para as funções, ou ainda enviar como retorno para o programa principal.

No próximo exemplo de função (Figura 57), a proposta era de que essa recebesse por parâmetro o tamanho e, ao invés de desenhar uma, seriam desenhadas três casas. Nesse exemplo, fica visível o reaproveitamento de comandos, pois com a mesma função “desenhe uma casa com comprimento”, foram desenhadas três casas de tamanhos diferentes, o que foi possível variando o parâmetro “comprimento” e, assim, foi modificado o tamanho das casas traçadas.

Figura 57 – Passo 8 da atividade 15 com funções e parâmetros (A_{22})



Fonte: da pesquisa.

Analisando a construção desse passo, observamos que o aluno (A_{22}) mostrou ter entendido o conceito de passagem de parâmetro, pois construiu o passo da atividade corretamente. Percebemos que, ao construir a função, utilizou a variável “comprimento” como parâmetro e, assim, alterou o tamanho das casas. Por meio dessas observações, podemos afirmar que, para o conceito de funções e seus parâmetros com esse aluno, a ferramenta também se mostrou eficiente.

Essa segunda categoria estabelecida na análise dos dados da pesquisa, denominada “sobre comandos”, é ampla, pois algoritmos são formados por comandos, de modo que é possível listarmos seis subcategorias: definir sequência de comandos; utilizar comandos de seleção; agrupar comandos; laços de repetição; laços encadeados; definir função. Subcategorias essas que nos direcionaram para muitos conceitos, tais como de resolução de problemas, ZDP, transposição didático, tempos didático e de aprendizagem, contrato didático, signos e significados, situações adidáticas e planos de ação. Conceitos importantes e que vieram ao encontro com a finalidade desta pesquisa: qualificar a prática da professora com relação à disciplina de Algoritmos.

c) Posicionar-se em relação ao objeto

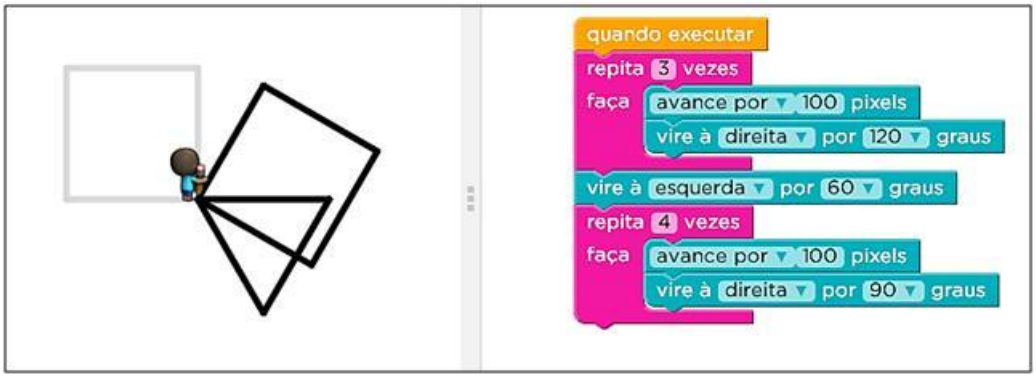
Outra questão que ficou evidente no material de análise, foi da incapacidade de o aluno se posicionar em relação ao objeto, no caso do *software* o “boneco”. Percebemos que os alunos têm dificuldade em se pôr no lugar do “boneco”, para, então, decidir se esse deve girar 60° ou 90°, precisa virar para a esquerda ou para a direita e, ainda, se avança 20 pixels ou 100 pixels. Assim, ao relacionarmos com as atividades realizadas nas aulas de algoritmos, antes mesmo de utilizarmos a ferramenta, percebemos essa mesma dificuldade no momento de simular a interação entre o computador e o usuário desse programa (algoritmo). Para construir um algoritmo, é preciso ter essa habilidade de se colocar no lugar do algoritmo, para definir o que o usuário deverá informar, em que variáveis serão armazenadas essas informações e quais são os cálculos que o algoritmo deve realizar para apresentar os resultados. Podemos relacionar essa mesma dificuldade de se posicionar em relação ao objeto, no caso do *software* o “boneco”, com a de se colocar no lugar do algoritmo.

Podemos citar como exemplo um exercício clássico utilizado em sala de aula, em que solicitava-se aos alunos que construíssem um algoritmo para calcular o peso ideal de qualquer pessoa, com base nas informações da altura e sexo, utilizando as seguintes expressões (para homens é PesoIdeal = altura * 72.7 - 58 e para mulheres é PesoIdeal = altura * 62.1 - 44.7). Percebemos que, para resolver esse exercício, os alunos apresentam dificuldade em definir quais são as informações que devem ser solicitadas ao usuário, nesse caso, a altura e o sexo, assim, como acabam solicitando ao usuário o “Pesoideal”, valor que o algoritmo deve calcular para então apresentar ao usuário.

Ainda em relação aos passos, que com o uso da ferramenta indicaram a dificuldade anteriormente mencionada do aluno de se posicionar em relação ao objeto, observamos que envolveram os conceitos de direção, sentido e ângulo. Apresentamos alguns exemplos das instruções que abrangeram o "virar (à direita/esquerda) por X graus", e o "avance por X pixels".

Observamos que alguns alunos (A_6, A_8, A_{11}, A_{14}, A_{15}, A_{16}, A_{18}, A_{25}) no total de 8 enfrentaram dificuldades com a noção de ângulo, pois o imperativo de posicionar-se no lugar do "boneco" e "virar por 60° ou 90°" fez com que o passo da atividade não fosse executado corretamente. A seguir, na Figura 58, apresentamos um exemplo dessa dificuldade no passo 6 da Atividade 5.

Figura 58 - Passo 6 da atividade 5 (A_{18})

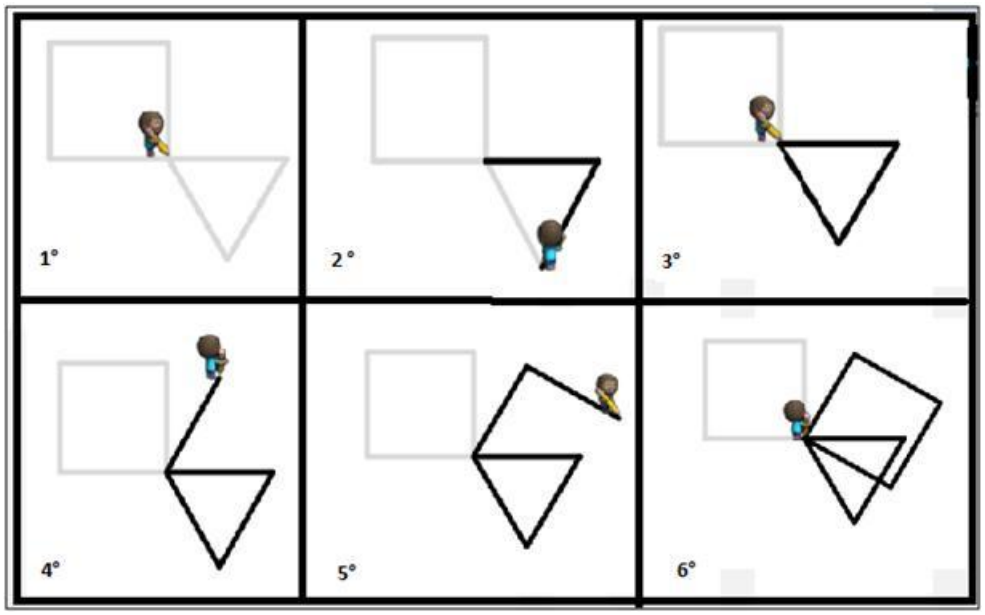


Fonte: da pesquisa.

Nesse exemplo, o que entendemos é que o aluno A_{18} não soube quantos graus era preciso girar para a esquerda para desenhar o quadrado no local solicitado. A noção de grau foi um obstáculo que o impediu de continuar com os passos propostos.

Assim, ao acompanharmos o processo de execução do passo 6 da Atividade 5 (Figura 59), observamos que, inseridos no comando de laço "repita 3 vezes", existem os comandos "Avance por 100 pixels" e o comando "vire à direita por 120 graus" que resultam no desenho do triângulo. Após finalizar o laço "repita 3 vezes", o boneco fica com o lápis apontado para a sua direita (conforme ilustrado no 3° quadrado da Figura 59), e para traçar o quadrado no local indicado (traço cinza claro), era necessário o comando "vire à esquerda por 180 graus", assim, o boneco iniciaria o traço posicionado corretamente. Como o aluno (A_{18}) selecionou o comando "vire à esquerda por 60 graus", o quadrado não foi traçado corretamente no local solicitado (conforme ilustrado nos quadrados 4°, 5° e 6° da Figura 59).

Figura 59 - Execução do passo 6 da atividade 5 (A_{18})



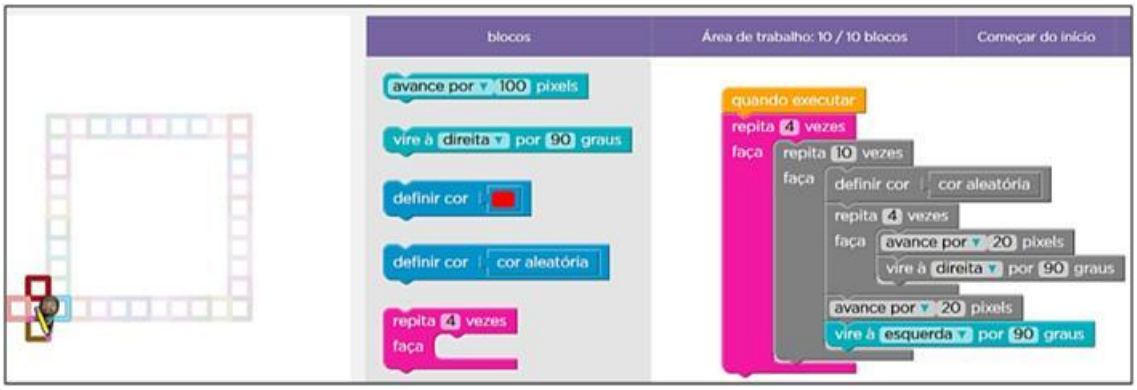
Fonte: da pesquisa.

Foi possível observar que a dificuldade não foi estabelecer os passos para traçar um triângulo e um quadrado, mas, sim, a noção de graus necessários para girar e desenhar esses objetos no lugar indicado. O que demonstra a dificuldade dos alunos de se colocarem no lugar do personagem (o boneco), do *software*.

Outra noção que parece ser simples, mas que se apresentou como obstáculo para o aluno continuar a utilizar a ferramenta foi a de direção e o sentido (esquerda e direita). Para Silva (2015), podemos atribuir o termo direção sempre que estivermos tratando de: direção horizontal, vertical e circular. Já o termo sentido, para o autor, estará sendo empregado em: da direita para a esquerda; ou da esquerda para a direita; de baixo para cima; de cima para baixo; sentido anti-horário e sentido horário.

Observamos que na proposta de selecionar a opção de sentidos diferentes, de virar à "esquerda" ao invés de à "direita", como ilustrado na Figura 60, o aluno A_{11} não conseguiu executar corretamente o passo 8 da Atividade 7, interrompendo o processo.

Figura 60 - Passo 8 da atividade 7 (A_{11})



Fonte: da pesquisa.

Para que o passo 8 fosse executado corretamente, a linha do comando "vire à esquerda por 90 graus" deveria estar posicionada abaixo do comando "repita 10 vezes" para que fossem desenhados dez pequenos quadrados na direção vertical e só então fosse executado o comando de virar 90 graus. O comando deveria ter selecionado o sentido inverso, ou seja, ao invés de esquerda, era necessário virar à direita para que iniciasse o desenho de dez pequenos quadrados na direção horizontal.

Percebemos também que o aluno A_5 interrompeu as atividades no passo em que necessitava usar um comando de direção (Figura 61), no qual era necessário virar à "direita" para resolver de forma correta. Nesse caso, não é possível afirmar que o aluno apresenta dificuldade em direcionar o "lápis", mas pode-se supor que a desistência seja pela dificuldade em distinguir, na ferramenta gráfica qual o sentido do caminho a ser percorrido.

Figura 61 - Passo 1 da atividade 5 (A_5).

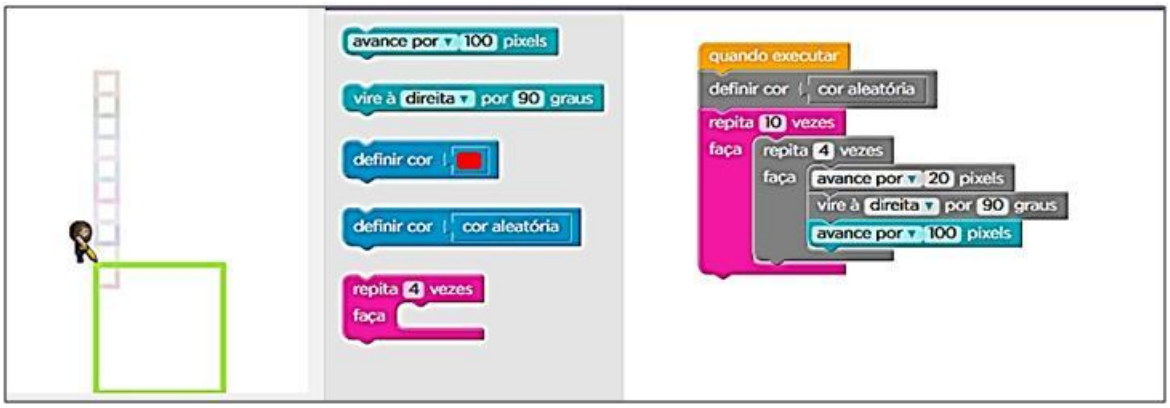
Fonte: da pesquisa.

Verificamos, ainda, que os alunos abandonaram as atividades que exijam um procedimento "diferente" dos que já lhes eram familiares. A ação mental de colocar-se na posição do personagem e decidir se o lápis na ferramenta deveria virar para a direita ou

para a esquerda e completar os desenhos propostos nos passos foi motivo para alguns alunos interromperem as atividades.

A noção de pixel foi outra dificuldade de interpretação que pode ter impedido que os alunos continuassem a realizar os passos. Pixel pode aqui ser interpretado como o quanto deveria ser avançado, pois o aluno $A_{10,}$ ao invés de selecionar "avance por 20 pixel", que é o tamanho definido pelo passo 7 da Atividade 7, selecionou "avance por 100 pixel" representando, assim, um quadrado de lado de maior medida (Figura 62).

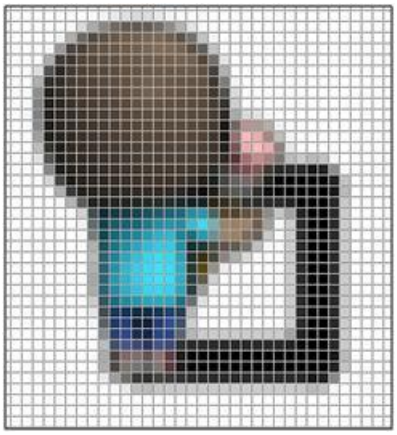
Figura 62 - Passo 7 da atividade 7 (A_{10})



Fonte: da pesquisa.

Em ferramenta de edição de imagem, se configuramos para que exiba como plano de fundo as linhas da grade, ao ampliarmos o zoom será possível visualizar os quadradinhos (pixels) que compõem a imagem (Figura 63).

Figura 63 - Imagem ampliada com pixels



Fonte: da pesquisa.

Mesmo o aluno não conhecendo o significado de pixel, poderia ter visualizado na execução que o tamanho ficou muito maior do que o proposto (Figura 62) e, assim,

diminuir o número estabelecido no comando “avance” para “avance por 20 pixels”. A ferramenta permite ao aluno a correção dos passos propostos durante o processo, portanto, era possível ter a percepção sobre o número de pixels que deveriam ser avançados nesse passo para resolver o desafio proposto.

Ao observarmos o passo da atividade proposta pelo aluno A_{10} (Figura 62) e ao confrontarmos essa com a resolução proposta do mesmo passo pelo aluno A_{12} (Figura 64), o que percebemos é a dificuldade de posicionar os comandos, ou seja, se o comando de “avance por 20 pixels” estaria inserido antes de finalizar o comando “repita 4 vezes” ou após ter finalizado esse. Como podemos notar na figura 64, o comando “avance por 20 pixels” foi inserido após finalizar o comando de “repita 4 vezes”, o que faz com que, após traçar um quadrado com lados de tamanhos de 20 pixels, o boneco avance mais 20 pixels para repetir todos os comandos inseridos no comando “repita 10 vezes”, construindo com esses uma linha na direção vertical de pequenos quadrados.

Figura 64 - Passo 7 da atividade 7 (A_{12})

Fonte: da pesquisa.

Já a escolha de inserir o comando “avance por 20 pixels” antes de finalizar o comando de “repita 4 vezes” (Figura 62) fez com que o traço do quadrado apresentasse lados de tamanhos de 120 pixels, pois o boneco avança mais 100 pixels, ainda incluso no laço responsável pelo traço do quadrado. Assim, são construídos dez quadrados no mesmo lugar e não dez quadrados na direção vertical, conforme enunciado do passo da atividade.

A ferramenta com seus comandos na forma de blocos e seus personagens representariam os objetos concretos, dos quais na percepção da professora auxiliariam aos alunos estabelecer as reflexivas necessárias para ensino e aprendizagem de algoritmos. Com base em Piaget, para que o sujeito consiga “interiorizar” essas ações que constituem o

pensamento, é necessário aprender primeiramente a executá-las materialmente para, em seguida, ser capaz de construí-las em pensamento.

Para Piaget, o sujeito compreende o que lhe é externo por meio da sua interação com o objeto, e por meio da abstração empírica retira as propriedades do objeto e estabelece relações. É tarefa do aluno desenvolver sua estratégia, e do professor, agir de maneira a criar as situações necessárias para que isso ocorra, pois, a descoberta promove a autonomia do sujeito. Segundo o autor "o ideal da educação não é aprender ao máximo, maximizar os resultados, mas antes de tudo é aprender a aprender; é aprender a se desenvolver e aprender a continuar a se desenvolver depois da escola" (PIAGET, 1978, p. 225).

Assim, ao refletirmos sobre essas dificuldades, supomos que podem vir da incapacidade do aluno de se posicionar em relação ao objeto. Verificamos que os alunos apresentam dificuldade em colocar-se no lugar do "boneco", para, então, decidir se esse deveria girar 60° ou 90°, se necessitava virar para a esquerda ou direita e ainda se esse deveria avançar 20 pixels ou 100 pixels. Desse modo, podemos relacionar essa mesma dificuldade de se posicionar em relação ao objeto, no caso do *software* o "boneco" com a de se colocar no "lugar do algoritmo".

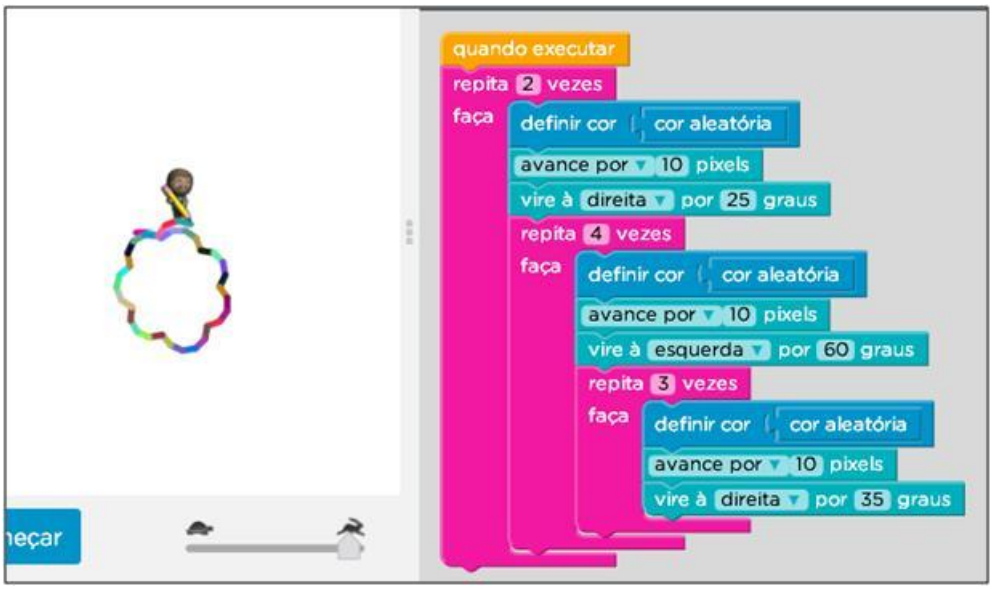
Portanto, esses apontamentos e estudos direcionam para uma necessária ponderação sobre os conteúdos envolvidos na disciplina, e na forma como esses são apresentados e cobrados dos alunos. Estariam sendo utilizadas, nas aulas, atividades e reflexões que preveem a habilidade de colocar-se no "lugar do algoritmo", para decidir quais seriam as variáveis necessárias para entrada de dados, quais os cálculos necessários, e quais seriam as saídas necessárias? Ou ainda, indica-se a possibilidade de colocar-se no "lugar do algoritmo" para estabelecer o plano de ação que, ao ser executado, apresente os resultados solicitados pelos enunciados apresentados nas atividades da disciplina?

d) Criar representações livres

Alguns dos passos propostos nas atividades da ferramenta solicitavam que os alunos criassem uma imagem com uso dos comandos já conhecidos e recentemente apresentados, ou, ainda, que alterassem os códigos já inseridos como sugestão para criar figuras. Porém, observamos que esses passos que exigiam do aluno esse tipo de criação, foram praticamente ignorados, pois entre os quinze alunos que passaram pelo passo 5.10 (Figura 65), somente sete criaram uma imagem diferente, três desses copiaram comandos

de outros passos e cinco não inseriram qualquer comando para alterar o desenho. Esses dados demonstram que mais de 53,3%, dos quinze alunos da turma, enfrentaram dificuldade de criar.

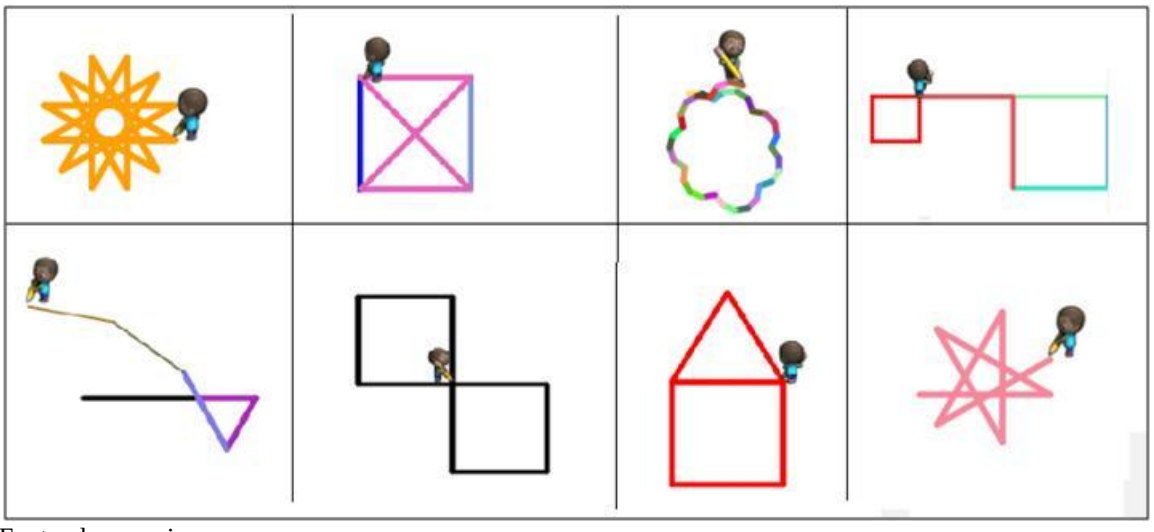
Figura 65- Passo 10 da Atividade 5 (A_{22})



Fonte: da pesquisa.

A ferramenta propôs aos alunos em alguns dos passos das atividades que esses exercitassem sua criatividade, que com o uso de comandos criassem imagens. Podemos demonstrar, na Figura 66, as imagens criadas pelos oito alunos dessa turma, porém, temos que considerar que em nenhum outro momento anterior ao uso da ferramenta, os alunos tiveram atividades com esse propósito.

Figura 66 - Imagens criadas



Fonte: da pesquisa.

D'Ambrosio (1989), enquanto descrevia a típica aula de matemática que leva os alunos a acreditarem que fazer matemática é seguir e aplicar regras transmitidas pelo professor, sinaliza que, ao aluno, não é dado, em nenhum momento, a oportunidade de criar nada, fazendo com que o aluno passe a acreditar que, na aula de matemática, o seu papel é passivo e desinteressante.

> Em nenhum momento no processo escolar, numa aula de matemática geram-se situações em que o aluno deva ser criativo, ou onde o aluno esteja motivado a solucionar um problema pela curiosidade criada pela situação em si ou pelo próprio desafio do problema. Na matemática escolar o aluno não vivencia situações de investigação, exploração e descobrimento. O processo de pesquisa matemática é reservado a poucos indivíduos que assumem a matemática como seu objeto de pesquisa. É esse processo de pesquisa que permite e incentiva a criatividade ao se trabalhar com situações problemas (D'AMBROSIO, 1989, p. 16).

Ainda sobre esse processo de criar Góes e Cruz (2006) ao interpretar Vygotsky em relação à cognição e imaginação na infância e na adolescência, afirmam que:

> [...] a imaginação é fundamental no alcance de um nível elevado do pensamento, porque torna possível operar com imagens que não estão na realidade percebida. Nenhuma cognição acurada da realidade é possível sem certo elemento de imaginação e, por outro lado, a criação artística ou a invenção demandam a participação tanto da imaginação quanto do pensamento realista (GÓES; CRUZ, 2006, p. 42).

Para os autores, a imaginação enriquece o conhecimento da realidade e enriquece a atuação sobre ela, o que demonstra a importância de atividades que envolvam esse tipo de ação para o integral desenvolvimento do aluno. Esses apontamentos dos autores geram questionamentos referentes às atividades propostas na disciplina de Algoritmos, como se essas deveriam também abordar as questões relacionados à criação.

No início da análise dos dados da pesquisa, procuramos identificar quais eram “os comportamentos” que se assemelhavam entre os alunos que resolveram os exercícios utilizando a ferramenta gráfica. Ou ainda, quais os exercícios dos quais os alunos apresentaram resoluções parecidas, para então refletir sobre o que essas poderiam representar no processo de ensino aprendizagem na disciplina de Algoritmos.

Depois de refletir sobre as resoluções, essas foram relacionadas com outras atividades realizadas pelos alunos nas atividades de sala de aula que não envolviam a

ferramenta gráfica e os conceitos da disciplina. Essas reflexões nos direcionaram para muitos conceitos, tais como Contrato didático, motivação, tempos didático e de aprendizagem, resolução de problemas, planos de ação, transposição didática, signos e significados, situações adidáticas e Zona de Desenvolvimento Proximal.

Hoje, percebemos a necessidade da atenção que se deve dispensar para as atividades realizadas pelos alunos, o imperativo de refletir sobre as resoluções apresentadas, deixando de, simplesmente, apresentar uma resolução pronta. Podemos afirmar que essas reflexões e esses conceitos foram importantes, pois contribuíram com a finalidade desta pesquisa: buscar formas de qualificar a prática docente em relação à disciplina de Algoritmos.

6 CONSIDERAÇÕES FINAIS

A inquietação acerca das dificuldades apresentadas pelos alunos e os altos índices de reprovação e evasão na disciplina de Algoritmos nos levaram a repensar a prática adotada até então ao ministrar essa disciplina. Um dos principais problemas percebidos durante a prática docente foi identificar quais eram as reais dificuldades dos alunos e essa problemática implicou reflexões sobre as estratégias de ensino utilizadas em mais de dez anos de sala de aula como ministrante dessa disciplina. Essas inquietações instigaram esta pesquisa que trouxe à tona, não somente as dificuldades dos alunos, bem como alguns dos problemas que envolvem o processo de ensino-aprendizagem na disciplina de Algoritmos.

No início desta pesquisa, ainda com certa visão "reduzida" do ambiente da sala de aula, começamos a pesquisar ferramentas gráficas utilizadas para o ensino de Algoritmos, com a expectativa de que, por meio de seu uso, seria possível auxiliar os alunos na aplicação ou ampliação dos conteúdos. Com base nas dificuldades do processo de ensino-aprendizagem na disciplina de Algoritmos e na falta de pesquisas que pudessem orientar esse processo, definimos como tema de pesquisa: A relação entre o estudo de Algoritmos e o uso de ferramentas gráficas, como decorrência desse, definimos como pergunta principal: Em que medida o uso de uma ferramenta gráfica de programação pode qualificar os processos de ensino e de aprendizagem na disciplina de Algoritmos?

Os participantes desta pesquisa são os alunos da disciplina Algoritmos (AL) do curso do TSPI, da turma da manhã (1M1/2014-2), do Curso Superior de Tecnologia em Sistemas para a Internet (TSPI) do Instituto Federal Sul-Rio-Grandense (Ifsul) do campus Passo Fundo.

Elegemos como opção metodológica a abordagem qualitativa para analisar o processo envolvido na utilização da ferramenta gráfica *Blockly* e refletir sobre seu potencial. O material de análise para nossa investigação foi produzido pelas atividades da ferramenta que gravou a resolução de todos os passos executados por cada aluno, pelas provas aplicadas na turma, pelo material produzido nas aulas e por nossas próprias observações. Portanto, após a análise dos dados, podemos expor algumas considerações acerca da pesquisa.

Por meio desse processo, o papel do *software* utilizado, aos poucos, foi mudando, além de ferramenta para os alunos aplicar e ampliar os conhecimentos abordados na disciplina, tornou-se um instrumento de diagnóstico. Com sua aplicação surgiram reflexões importantes para além do uso de um *software*, pela observação do "comportamento", das

atitudes e procedimentos adotados pelos alunos e, por nós, no papel de professora, durante esse processo. O material de análise produzido nos permitiu identificar dificuldades do processo da construção de algoritmos em relação aos conteúdos já abordados anteriormente na disciplina que, até então, como professora-pesquisadora não tínhamos conseguido observar.

Um conceito importante originado da Teoria das Situações Didáticas, o contrato didático, que analisa as relações que se estabelecem entre professores e alunos, em função do conhecimento, conceito aponta que "a ilusão de que existe um contrato é indispensável para que a relação aconteça e seja eventualmente bem-sucedida. Cada um - professor e o aluno - imagina o que o outro espera dele, o que cada um pensa que o outro pensa..." (BROUSSEAU, 2008, p. 89). Conceito importante para repensar os papéis, as responsabilidades dos alunos e da professora da disciplina de Algoritmos. Cabe a nós, professores, refletirmos se são feitas as intervenções necessárias duramente a aplicação da ferramenta para manter o interesse dos alunos, deixando clara as cláusulas iniciais do contrato e identificar os efeitos desse contrato didático que podem ocorrer durante o processo de ensino-aprendizagem.

Observando o desempenho geral da turma na realização das atividades na ferramenta, com relação ao interesse demonstrado pelos alunos, percebemos que não foi muito produtivo a sua aplicação, pois não manteve os alunos motivados em utilizá-la. Com base no número de participantes que começaram as atividades, que foi de 26 alunos, somente cinco finalizaram todos os passos, o que corresponde a 19,23% da turma. Nesse sentido, podemos inferir que nem todos aderiram às cláusulas estabelecidas pelo contrato didático, sendo, assim, nos cabe repensar a forma e o período da aplicação da ferramenta.

A opção de a ferramenta ser utilizada na sua maior parte extraclasse, demonstrou não ser eficaz, ao menos tanto quanto como esperávamos, pois, essa foi vinculada à ideia de ser uma atividade sem muita importância, apesar de essa opção ter sido adotada pela professora por ponderar o tempo de aprendizagem dos alunos. A intenção foi de respeitar o tempo necessário de cada sujeito para superar as dificuldades, fundamentada nos conceitos de transposição didática que Chevallard (2000), aponta como variáveis fundamentais quanto ao planejamento de ensino, o tempo didático e o tempo de aprendizagem.

Ao analisarmos de forma individual o processo de desenvolvimento das atividades dos alunos com a ferramenta, procurando compreender o que poderia ter causado as dificuldades apresentadas, ainda com base no conceito de transposição didática, identificamos o conceito de "noções paramatemáticas". Consideradas "noções

ferramentas", que se referem a conhecimentos que se supõe que o aluno já tenha obtido durante a sua formação anterior (Chevallard,2000), podemos citar como exemplo de conhecimentos que a ferramenta demanda que o aluno tenha: a noção de ângulo; de posição, de direção e sentido.

No decorrer das aulas, também pudemos identificar que supomos, como professora, que o aluno já teria conhecimentos considerados como "noções ferramentas", para as quais não foi dispensada a atenção necessária. Conceitos foram apresentados sem uma reflexão sobre o seu significado, podemos citar como exemplo e que foram abordadas nas primeiras listas de exercícios as seguintes noções: juros, percentagem, média ponderada, consumo médio de combustível, conversão entre moedas. Fato que aponta também para a necessidade de rever o formato das "listas de exercícios" utilizadas, pois essas envolvem conceitos apresentados sem significado para o aluno, o que pode tornar a disciplina de Algoritmos exaustiva e ser um dos possíveis motivos de desistências e reprovações na turma.

Outra reflexão que emerge nessa etapa é referente aos conteúdos que são "cobrados" nas atividades de sala de aula e avaliações da disciplina de Algoritmos. Cabe uma reavaliação quanto à forma de apresentação desses conteúdos considerados de "noções paramatemáticas", se esses estariam recebendo atenção necessária nas aulas, considerando a importância dispensada nas avaliações da disciplina.

Refletindo sobre a forma como são apresentados os conceitos da disciplina aos alunos com relação ao significado e o sentido das palavras, percebemos a necessidade de priorizar o significado dos signos, ou seja, das palavras, de cada um dos comandos utilizados, procurando evitar a simples apresentação de sua função ou uso excessivo de analogias nas suas aplicações.

Podemos afirmar que a percepção de responsabilidade por parte da professora-pesquisadora diante da aprendizagem e desempenho dos alunos após esses estudos aumentou consideravelmente. Polya ao referir-se aos métodos de resolução de problemas em matemática, já mencionava a responsabilidade do professor na escolha dos problemas propostos para os alunos, alegando que "se lhes faltar compreensão e interesse, isto nem sempre será culpa sua" (1995, p. 4). Ainda, acerca de como os problemas da disciplina de Algoritmo estavam sendo apresentado aos alunos, citamos Gauthier et al., que apontam para a importância da clareza das perguntas formuladas pelos professores, quando afirmam que "ocorre de os alunos serem incapazes de responder às perguntas que lhe são feitas porque elas são ou demasiado vagas ou demasiado ambíguas" (2013, p. 224).

Sobre a importância da clareza da formulação das perguntas apontada por Gauthier et al. (2013), cabe-nos repensar os enunciados dos problemas propostos, se esses não estariam "demasiado vagos ou demasiado ambíguos", dificultando para aluno o processo inicial e mais importante da construção de algoritmos que é a compreensão do problema, primeira das quatro fases de resolução de um problema, propostas por Polya (1995).

Observando a forma como a ferramenta apresenta os conceitos de algoritmos, bem como o processo de resolução das atividades realizadas pelos alunos envolvidos na construção, foi possível analisar o modo como os conteúdos da disciplina de Algoritmo estavam sendo apresentados. Portanto, diante da pergunta inicial "em que medida uma ferramenta gráfica de programação pode qualificar os processos de ensino e de aprendizagem na disciplina de Algoritmos", podemos afirmar que, mesmo diante do pequeno espaço de tempo do uso, e do período escolhido para aplicação, o processo nos possibilitou compreender a ferramenta como um instrumento mediador. Instrumento utilizado para estimular os alunos a elaborar os planos de ações e resolver os desafios, ao mesmo tempo em que dá pistas para incidir na Zona Desenvolvimento Proximal. Assim como também nos permitiu identificar elementos importantes que vêm ao encontro de uma das principais finalidades desta pesquisa: qualificar a prática da professora com relação à disciplina de Algoritmos.

Ao iniciar esta pesquisa, a principal preocupação era a dificuldade de os alunos construírem os Algoritmos e o quanto a utilização de uma ferramenta gráfica poderia auxiliá-los nesse processo. Porém, durante o desenvolvimento desta investigação, surgiram elementos importantes que possibilitaram identificar dificuldades próprias da professora-pesquisadora, com relação à sua prática e à sua habilidade de sentir-se parte do processo ensino-aprendizagem de algoritmos. Entendemos que esse processo de qualificar a prática não termina aqui, terá correções todos os semestres, pois conforme Brousseau "o ensino e a aprendizagem acontecem por meio de processos que nunca estão em um equilíbrio estável. Devem ser entendidos como uma sucessão de "correções" pontuais que não podem ser justificadas isoladamente" (2008, p. 89). Assim, também, devemos entender o que Gauthier et al. (2013) apontam sobre a pesquisa na área de ensino, quando trazem que os enunciados oriundos da pesquisa ajudam os professores a analisar e compreender a situação, dando-lhes mais possibilidades de intervenções potencialmente pertinentes, bem como ajudar os envolvidos a formalizar seus saberes a respeito da ação.

Especificamente sobre a ferramenta *Blockly*, podemos afirmar que o seu uso apresenta contribuições para o ensino e aprendizagem da disciplina de Algoritmos e que,

por meio desta primeira aplicação, apontaram para formas diferentes de aproveitamento, que servirão para futuras análises, novas possibilidades, visando auxiliar o processo ensino-aprendizagem de Algoritmos.

E para finalizar, nossas percepções indicam que o uso de um software com o intuito de auxiliar no processo de ensino-aprendizagem de algoritmos aponta para um indispensável entendimento sobre os "saberes" necessários para ensinar. Para reforçarmos o que percebemos sobre a importância de estudos acerca desses "saberes", retomamos Gauthier et al. (2013), que afirmam "a responsabilidade ética para com os alunos provoca no professor a necessidade de conhecer os melhores meios para instruir e educar, ou seja, ele não pode ignorar o que a pesquisa diz a respeito das melhores práticas" (p. 399).

Enfim, como últimas palavras, ponderamos que esta pesquisa nos direcionou para muitos conceitos importantes que vieram ao encontro da sua finalidade, de qualificar a prática docente com relação à disciplina de Algoritmos, proporcionando formação continuada e qualificação profissional.

REFERÊNCIAS

ALICE. Versão 3.x. Carnegie Mellon University, 2008. Disponível em: <http://www.alice.org>. Acesso em: 28 dez. 2014.

ALMOULOUD, Saddo A. *Fundamentos da didática da matemática.* Curitiba: UFPR, 2007.

ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi de. *Fundamentos da programação de computadores*: algoritmos, PASCAL, C/C++(padrão ANSI) e JAVA. 3. ed. São Paulo: Pearson, 2012.

AURELIANO, Viviane Cristina Oliveira; TEDESCO, Patrícia Cabral de Azevedo Restelli. Utilizando o Scratch para Apoiar o Processo de Ensino-aprendizagem de Programação para Iniciantes na EaD. In: XXI WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO, 2013, Maceió. *Anais*... Maceió.

BERLINSKI, David. *O Advento do algoritmo*: a idéia que governa o mundo. São Paulo: Globo, 2002.

BOGDAN, Robert C.; BIKLEN, Sari Knoop. Características da investigação qualitativa. In: *Investigação qualitativa em educação*. Porto: Porto Editora, 1994. p. 47-51.

BRANDÃO, Leonidas de Oliveira; BRANDÃO, Anarosa Alves Franco; RIBEIRO, Romenig da Silva. *iVProg*–Uma Ferramenta de Programação Visual para o Ensino de Algoritmos. In: WORKSHOPS DO CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 1, 2012, Rio de Janeiro, *Anais* ... Rio de Janeiro.

BROOKSHEAR, Glenn. *Ciência da Computação:* uma visão abrangente. 7. ed. Porto Alegre: Artmed, 2005.

BROUSSEAU, Guy. *Introdução ao estudo da teoria das situações didáticas:* conteúdos e métodos de ensino. São Paulo: Ática, 2008.

CARDOSO, Ruth. Corrêa Leite; SAMPAIO, Helena. *Estudantes universitários e o trabal*ho. Revista Brasileira de Ciências Sociais, São Paulo, v. 9, n. 26, out. 1994. Disponível em: <http://www.anpocs.org.br/portal/publicacoes/rbcs_00_26/rbcs26_03.htm> . Acesso em: 10 fev. 2016.

CHAN, Iana. Ferramentas gratuitas para você aprender a programar. *Revista Super Interessante*. 2013. Disponível em:<http://super.abril.com.br/blogs/superlistas/tag/servicos/>. Acesso em: 10 set. 2015.

CHEVALLARD, Yves. *La transposición didáctica: del saber sabio al saber enseñado.* Trad. Claudia Gilman. 3. ed. Buenos Aires: Aique, 2000.

CODE.ORG. 2015. Disponível em: < https://code.org/about/>. Acesso em: 28 mar. 2015.

COSTA, Elis Regina da. Revisão de literatura sobre a motivação de alunos para aprender : implicações para o ensino . *Revista eletrônica do curso de Pedagogia do campus Jatai - UFG***,** v.1, n.14, p.1-16, 2013. Disponível em: <http://h200137217135.ufg.br/index.php/ritref/article/view/24150>. Acesso em: 10 ago. 2015.

CURY, Helena Noronha. *Análise de erros:* o que podemos aprender com as respostas dos alunos. Belo Horizonte: Autêntica, 2007.

D'AMBROSIO, Beatriz S. Como ensinar matemática hoje*? Temas e Debates*. SBEM. Ano II. n. 2. Brasília. 1989. p. 15-19. Disponível em: <http://educadores.diaadia.pr.gov.br/arquivos/file/2010/artigos_teses/matematica/artigo_beatriz.pdf>. Acesso em: 10 out. 2015.

DUVAL, Raymond. Registros de representação semióticas e funcionamento cognitivo do pensamento. Trad. Méricles Thadeu Moretti. In: *REVEMAT*. Florianópolis, v. 7, n.2, p. 266-267, 2012.

DUVAL, Raymond. Raymond Duval e a Teoria dos registros de representação semióticas. In: *RPEM*. Campo Mourão, PR, v. 2, n.3, p. 10-34, 2013.

DUVAL, Raymond. Registros de representação semióticas e funcionamento cognitivo da compressão em matemática. In: MACHADO, Silvia Dias Alcântara (Org.). *Aprendizagem em Matemática*: Registros de representação semiótica**.** Campinas: Papirus, 2003. p. 7-33.

ESTEBAN, Maria Paz Sandín. *Pesquisa qualitativa em educação:* Fundamentos e tradições. Porto Alegre: AMGH, 2010.

ECHEVERRÍA, Maria Del Puy Perez; POZO, Juan Ignacio (Org.). Aprender a resolver problemas e resolver problemas para aprender. In: *A solução de problemas***:** aprender a resolver, resolver para aprender. Porto Alegre: ArtMed, 1998, p.13-42.

ECCHELI, Simone Deperon. *A motivação como prevenção da indis*ciplina. Educar em Revista, n. 38, Curitiba: 2008. Disponível em: < http://dx.doi.org/10.1590/S0104-40602008000200014> Acesso em: 10 set. 2015.

FISCINA, Fabrizio Leandro Fonseca; BORGES, Marcio. *Algoritmos: licenciatura em ciência da computação*. Salvador: UNEB/GEAD, 2013.

FONSECA, Luís Carlos Costa *et al*. Proposta de um Ambiente Online para a Resolução e Correção Automática de Algoritmo. In: XX CONGRESO INTERNACIONAL DE INFORMÁTICA EDUCATIVA, 2015, Santiago. *Anais*... Chile. Disponível em: < http://www.tise.cl/volumen11/TISE2015/492-497.pdf> Acesso em: 10 jan. 2016.

FORBELLONE, André Luiz Villar; EBERSPÄCHER, Henri Frederico. *Lógica de Programação:* a construção de algoritmos e estruturas de dados. 3. ed. São Paulo: Pearson, 2005.

FREITAS, José Luiz Magalhães. Teoria das Situações. In: MACHADO, Silvia Dias Alcântara (Org.). *Educação Matemática*: uma (nova) introdução**.** 3. ed**.** São Paulo: Educ, 2008. p. 77-111.

GAUTHIER, Clermont *et al*. *Por uma teoria da Pedagogia*: pesquisas contemporâneas sobre o saber docente. Trad. Francisco Pereira. 3. ed. Ijuí: Unijuí, 2013.

GOÉS, Maria Cecilia Rafael de; CRUZ, Maria Nazaré da. *Sentido, significado e conceito*: notas sobre as contribuições de Lev Vigotski. Pró-Posições, v.17, n. 2, mai/ago. 2006.

GOOGLE DEVELOPERS (2015) *Blockly* : A visual programming editor. Disponível em: <https://developers.google.com/blockly>. Acesso em: 10 jun. 2015.

GRANDO, Neiva Ignês. Transposição didática e educação matemática**.** In: RAYS , Oswaldo Alonso (Org.). *Educação e ensino*: constatações, inquietações e proposições. Santa Maria: Pallotti, 2000.

_____. Dificuldades e obstáculos em educação matemática**.** In: *ESPAÇO PEDAGÓGICO.* Passo Fundo, v. 2, n.1, p. 109-122, dez. 1995.

_____; MOREIRA, Mariano; SILVA, Elcio Oliveira da. O contrato didático e o currículo oculto: um duplo olhar sobre o fazer pedagógico**.** In: *ZETETIKÉ*. Campinas, v. 4, n. 6, p. 5-7, jul/dez. 1996.

IFSUL. *Projeto do curso superior de tecnologia em sistemas para Internet*. Disponível em <http://painel.passofundo.ifsul.edu.br/uploads/arq/20150609165220135 3593 9.pdf>. 2014a. Acesso em: 12 jul. 2015.

IFSUL. *Projeto Pedagógico Institucional*. Disponível em:<www.ifsul.edu.br/index.php? option=com_docman&task=doc_download&gid=4048&Itemid=81>. 2014b. Acesso em: 15 jul. 2015.

JESUS, Andreia de; BRITO, Glaucia Silva. Concepção de ensino-aprendizagem de algoritmos e programação de computadores: a prática docente. 2009. Disponível em: < http://e-revista.unioeste.br/index.php/variascientia/article/view/2632/3107>. Acesso em: 4 fev. 2014.

JORDÃO, Fabio. *Google Blockly:* uma linguagem de programação baseada em quebra-cabeças. 2012. Disponível em: < http://www.tecmundo.com.br/programacao/24947-google-blockly-uma-linguagem-de-programacao-baseada-em-quebra-cabecas.htm>. Acesso em: 4 mai. 2014.

LOPES, Anita; GARCIA Guto. *Introdução à programação*. Rio de Janeiro: Campus, 2002.

MINAYO, Maria Cecília de Souza. *Pesquisa social:* teoria, método e criatividade. 29. ed. Petrópolis: Vozes, 2010.

MIRANDA, Elisangela Maschio de. *Uma ferramenta de apoio ao processo de aprendizagem de algoritmos.* Dissertação (Mestrado em Ciência da Computação) Universidade Federal de Santa Catarina, Centro Tecnológico. Florianópolis, 2004. Disponível em: <http://repositorio.ufsc.br/xmlui/handle/123456789/86766>. Acesso em: 13 jun. 2014.

NETO, Valter dos Santos Mendonça. A utilização da ferramenta Scratch como auxílio na aprendizagem de lógica de programação. 2013. *Congresso Brasileiro de Informática na Educação – CBIE 2013*. Disponível em: <http://www.br-ie.org/pub/index.php/wcbie/ article/view/2675/2329>. Acesso em: 5 jan. 2015.

OLHARDIGITAL. Bill Gates e Zuckerberg ensinarão programação em projeto educacional.*Revista Online.* 2013. Disponível em: <http://olhardigital.uol.com.br/ pro/noticia/38236/38236>. Acesso em: 15 ago. 2015.

OLIVEIRA, Marta Kohn de. *Vygotsky*: aprendizado e desenvolvimento um processo sócio-histórico. São Paulo: Scipione, 1992.

PAIS, Luiz Carlos. Transposição Didática. In: MACHADO, Silvia Dias Alcântara (Org.). E*ducação Matemática:* uma (nova) introdução. 3. ed**.** São Paulo: EDUC, 2008. p. 11- 48.

PALANGANA, Isilda Campaner. *Desenvolvimento e aprendizagem em Piaget e Vygotsky*: a relevância social. 3. ed. São Paulo: Sammus, 2001.

PAPADOPOULOS, George S. Aprender para o século XXI. In DELORS, Jaques. (Org) *A educação para o século XXI*: questões e perspectivas. Trad. Fátima Murad. Porto Alegre: Artmed, 2005. p. 19 - 34.

PAPERT, Seymour. *A máquina das crianças:* repensando a escola na era da informática. Porto Alegre: A

PASSOS, Claudio Cesar Manso; TEIXEIRA, Paulo Jorge Magalhães. Um pouco da teoria das situações didáticas (tsd) de Guy Brousseau. In: *ZETETIKÉ* . Campinas, v. 21, n. 39, Jan/Jun. 2013.

PIAGET, Jean. *A epistemologia genética*: sabedoria e ilusões da filosofia / Problemas da epistemologia genética. São Paulo: Abril Cultural, 1978.

POLYA, George. *A arte de resolver problemas*: um novo aspecto do método matemático. Tradução de Heitor Lisboa de Araújo. Rio de Janeiro: Interciência, 1995.

PUGA, Sandra; RISSETI, Gerson. *Lógica de Programação e estrutura de dados* com aplicações em java. 2. ed**.** São Paulo: Pearson, 2009.

SANTOS, Renato Manuel Simões. *Ensino da programação através de programação visual*. Dissertação (Mestrado em Ensino de Informática) – Universidade de Lisboa, Lisboa, 2013. Disponível em: < http://repositorio.ul.pt/handle/10451/913>. Acesso em: 15 dez. 2014.

SETTI, Mariangela de Oliveira Gomes. *Processo de Discretização do Raciocínio Matemático na Tradução para o Raciocínio Computacional:* um Estudo de Caso no Ensino Aprendizagem de Algoritmos. Tese (Doutorado em Educação) - Setor de Educação, Universidade Federal do Paraná, Curitiba, 2009. Disponível em: <www.ppge.ufpr.br/teses/D09_setti.pdf>. Acesso em: 3 abr. 2014.

SILVA, Domiciano Correa Marques da. *Noções Importantes de direção e sentido*; Brasil Escola. 2015. Disponível em: <http://brasilescola.uol.com.br/fisica/nocoes-importantes.htm>. Acesso em: 20 dez. 2015.

SPINA, Carli. Learn computer programming and Web. *Revista American Library Association*. 2013. Disponível em: <http://crln.acrl.org/content/74/10/522.short>. Acesso em: 11 jul. 2014.

VALASKI, Joselaine; PARAISO, Emerson Cabrera. Limitações da Utilização do Alice no Ensino de Programação para Alunos de Graduação. In: *XXIII SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO*, 2012, Rio de Janeiro, *Anais*... Rio de Janeiro: SBIE, 2012.

VIEGAS, Thaís R. *et al*. Uso das TICs no processo de ensino-aprendizagem de programação. In: XX CONGRESO INTERNACIONAL DE INFORMÁTICA EDUCATIVA, 2015, Santiago. *Anais*... Chile. Disponivel em: <http://www.tise.cl/volumen11/TISE2015/780-785.pdf>. Acesso em: 10 jan. 2016.

VIGOTSKI, Lev Semenovich. *A formação social da mente***:** o desenvolvimento dos processos psicológicos superiores. 6. ed. São Paulo: Martins Fontes, 1998.

VYGOTSKY, Lev Semenovitch. *Pensamento e linguagem*. 2 ed**.** São Paulo: Martins Fontes, 1998.

APÊNDICES

APÊNDICE A – Questionário aplicado aos alunos

Curso Superior de Tecnologia em

Sistemas para Internet

ALGORITMO (AL)

Prof. Carmem Scorsatto

Data : 01/09/2014

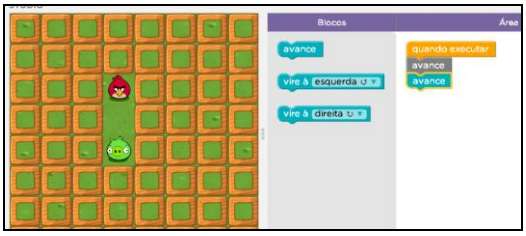
1) Você está encontrando dificuldades na disciplina ?

( ) Não ( ) Sim

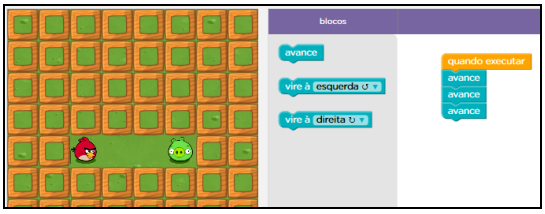
Justifique:

APÊNDICE B – Atividades realizadas pelo aluno A_{24}

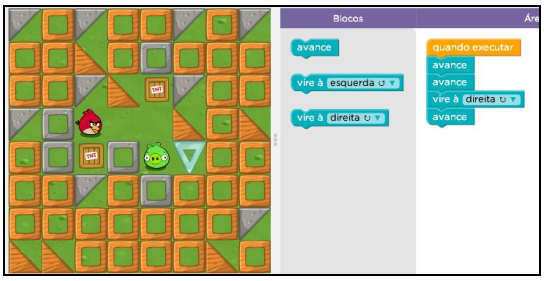
Consegues ajudar-me a apanhar o porquinho malvado? Empilha alguns blocos "mover para a frente" e depois clica em "Executar" para me ajudares a chegar até ele.



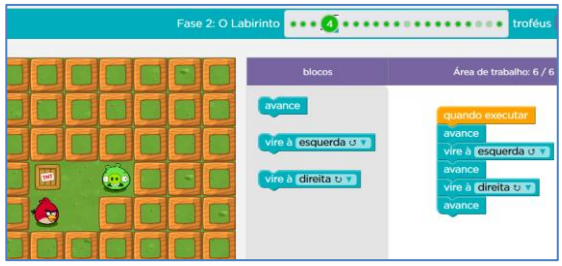
Este porco está a enervar-me. Ajuda-me a encontrá-lo!



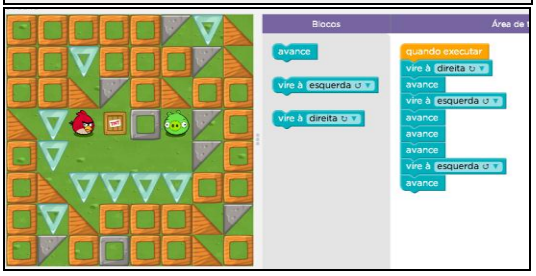
Traça o caminho e leva-me até ao porquinho tonto. Não choques com as dinamites, se não ficas sem penas!



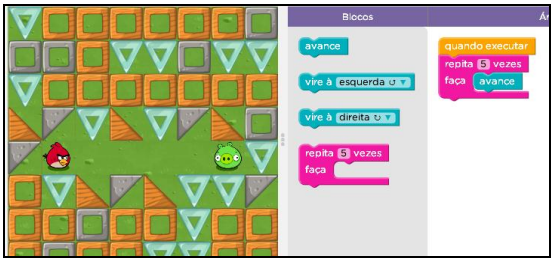
Leva-me até ao maldito verde! (Tem cuidado com a dinamite)



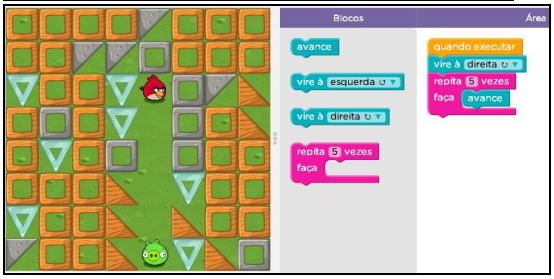
Fique calmo e ajude-me a encontrar o porco mau. Senão eu vou ficar zangado!



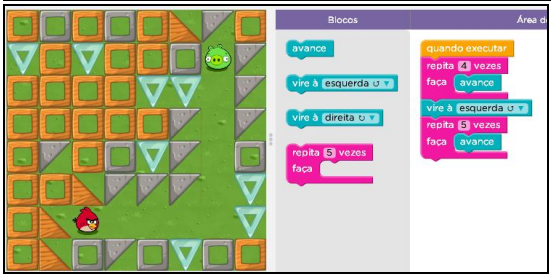
Há uma maneira de chegar ao porco bobalhão usando apenas 2 blocos. Você consegue descobrir como?



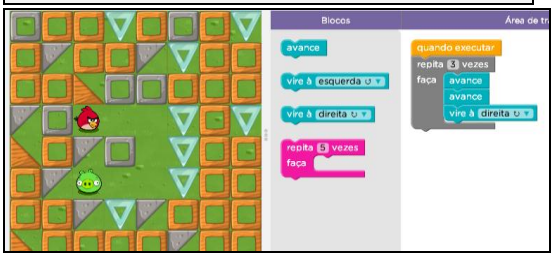
Tente me levar até o intruso verde usando apenas três blocos.



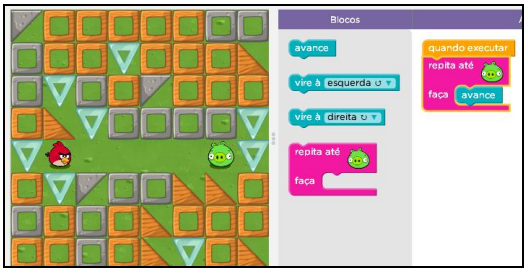
Ajude-me a banir esse porquinho mau usando o menor número de blocos possível. Tente usar mais de um bloco "repita".



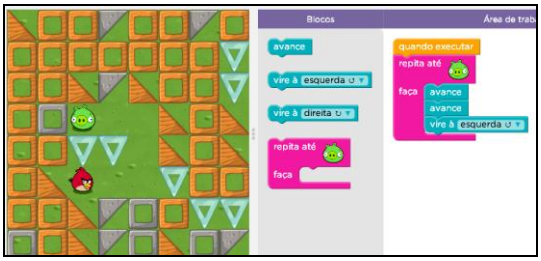
Quando um bloco estiver cinza, significa que você não pode excluí-lo. Resolva esse desafio usando o bloco "repita" que faz 3 repetições. Tente colocar esses 3 blocos dentro do bloco "repita" em cinza: avance, avance, vire.



Ok, use o novo bloco "repita até" - ele vai se repetir até que eu chegue naquele porco irritante.



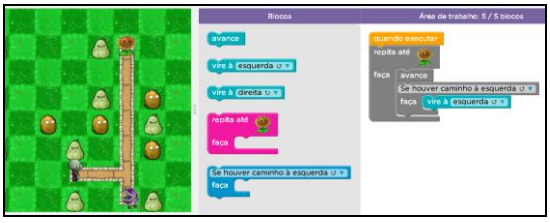
Ok, uma última vez para praticar - consegues resolver este um usando apenas 4 blocos?



Querida pessoa. Eu zombie. Eu com fome. Tenho... ir... girassol... Consegues colocar-me lá com apenas 5 blocos?

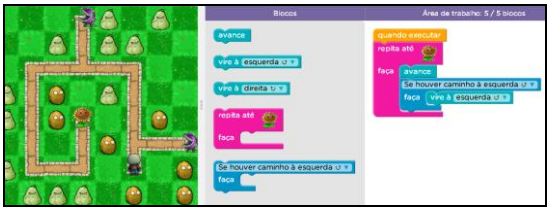
Ok, isto é semelhante, mas ligeiramente diferente. Consegues fazê-lo com apenas 5 blocos?

Usa o novo bloco "se" para me deixar decidir quando virar. Dica: só precisas de mais um bloco, mas aprende como podemos configurá-lo para que possas fazer sozinho na próxima vez.



Ok, isto é como o último quebra-cabeças, mas precisas de te lembrar de como usaste o bloco "se" e o bloco "repetir" juntos.

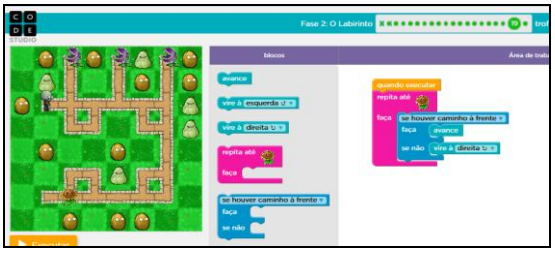
Eu querer girassol! Usa um bloco "se" para levar-me lá, com o menor número de blocos.



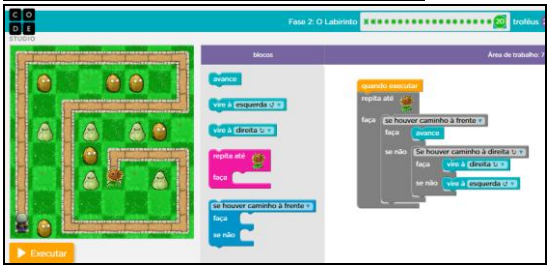
Ok, vamos praticar mais uma vez. Essa parte não é muito diferente, mas fique de olho nos comilões!

O bloco "se-senão" verifica uma condição e, em seguida, faz uma coisa OU outra. Para me levar ao girassol, tente usar esse novo bloco.

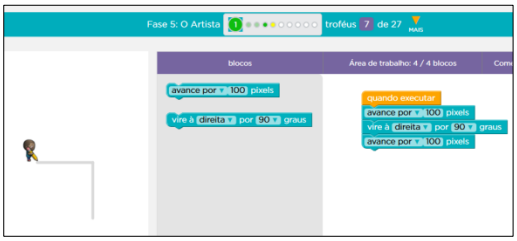
Vamos praticar o uso do bloco "se-senão" mais uma vez, você consegue na primeira tentativa?



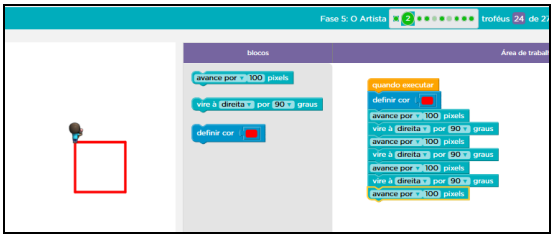
Você consegue usar apenas 3 blocos para me ajudar a percorrer um labirinto mais complexo? Se fizer isso direito, eu posso percorrer qualquer caminho cheio de curvas, não importa o comprimento.



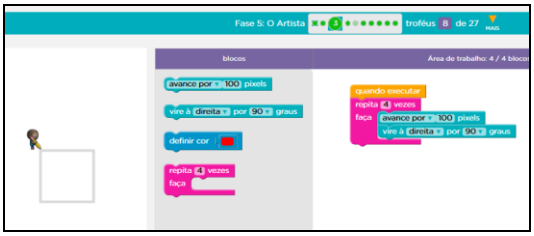
Olá, eu sou um artista! Você pode escrever códigos para me fazer desenhar quase tudo. Use alguns blocos para me fazer desenhar sobre as linhas cinza da figura.



Agora, desenhe um quadrado. NOTA: escolha sua cor preferida no novo bloco "defina a cor".

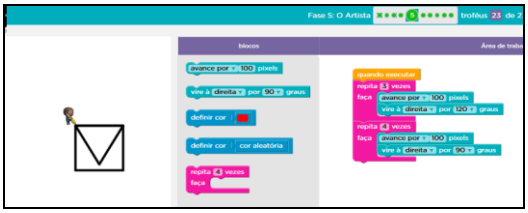


Faça um quadrado usando apenas 3 blocos.

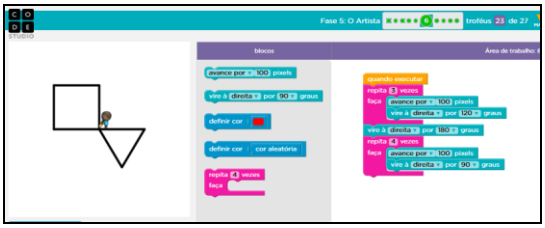


Desenhe um triângulo cujos lados têm cores diferentes, usando a opção "cor aleatória" que escolhe uma cor diferente a cada vez que é executado. Dica: você terá que descobrir o quanto virar, selecionando um número no bloco vire.

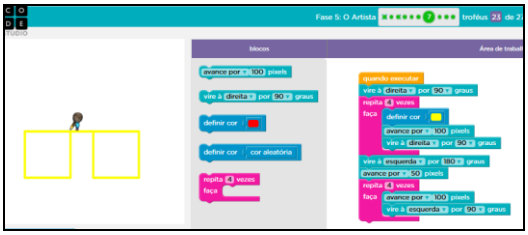
Agora, para praticar, desenhe um envelope usando um triângulo e um quadrado.



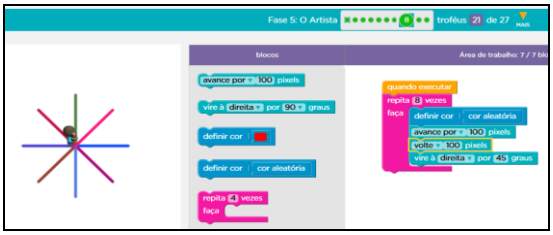
Você consegue descobrir como desenhar esse triângulo e esse quadrado? Dica: faça o triângulo primeiro e, em seguida, descubra o quanto virar antes de desenhar o quadrado.



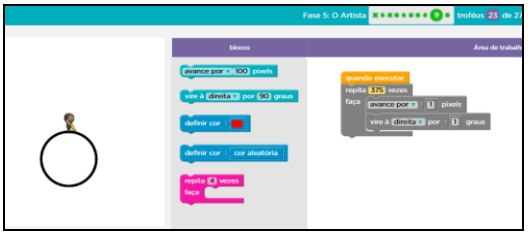
Ok, vamos dificultar um pouco - veja se você consegue desenhar esses óculos verdes. Os quadrados têm 100 pixels de cada lado, e eles são separados por 50 pixels. Não se esqueça de desenhar com a cor verde!



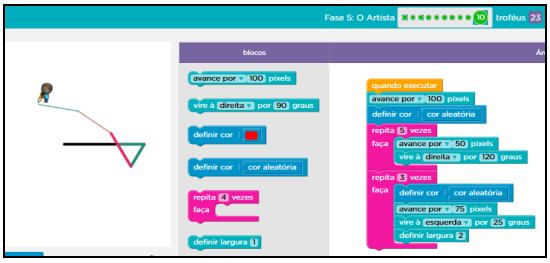
Ok, tente descobrir o que acontece se você executar esse código (ou clique em "Executar" para ver). Em seguida, execute-o várias vezes para completar o desenho. As cores serão diferentes toda vez que o código for executado.

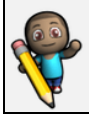


Você consegue descobrir qual número precisa usar para substituir os pontos de interrogação e desenhar um círculo?

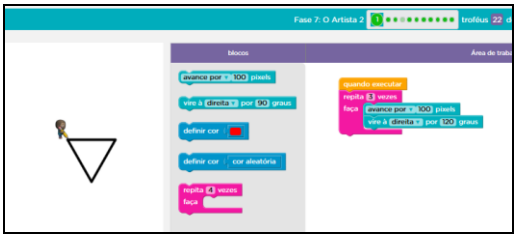


Você conseguiu! Agora, desenhe o que quiser. Algumas ideias divertidas: um boneco, um floco de neve ou uma espiral. Use também o novo bloco "defina largura". Divirta-se!

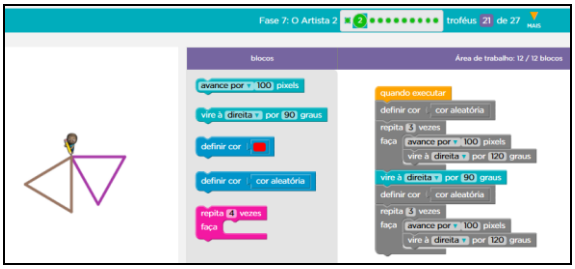




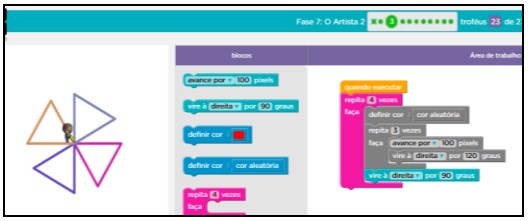
Você pode desenhar um triângulo (com bordas de 100 pixels) com apenas 3 blocos? Dica: use um bloco "repita".



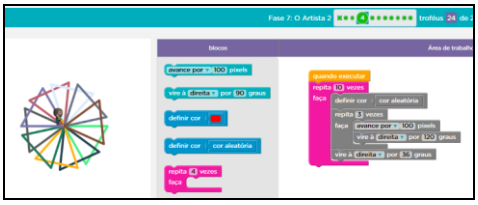
Acrescente um bloco "vire à direita 90 graus" em algum lugar no meio do programa, na área de trabalho, para desenhar esses triângulos.



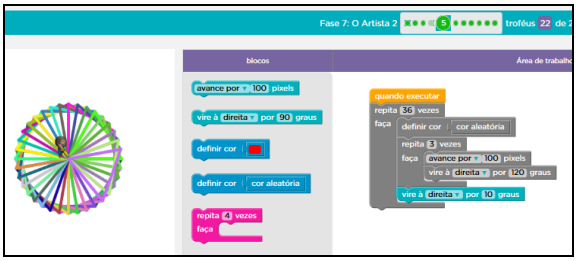
Muito bem, este é código que você escreveu para desenhar um único triângulo. Você consegue adicionar um bloco "repita" e um "vire" para fazer uma linda flor?



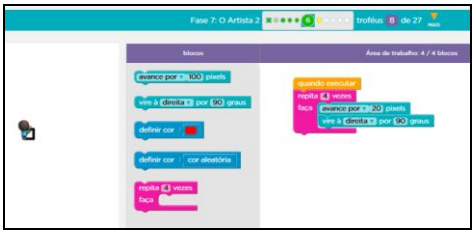
Este é o mesmo código do desafio anterior, mas virando apenas 36 graus depois de desenhar cada triângulo. Quantas vezes essa ação precisa se repetir? (Dica: depois de 360 graus, o desenho será um círculo completo)



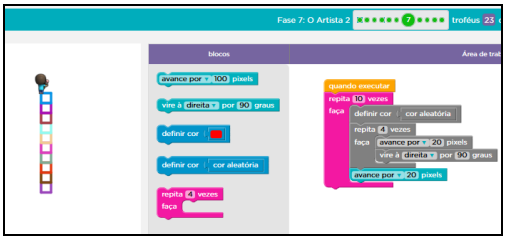
Este é o mesmo código do desafio anterior, mas repetindo cada virada 36 vezes. Qual deve ser a inclinação de cada virada em graus? (Dica: após 360 graus, o desenho será um círculo completo)



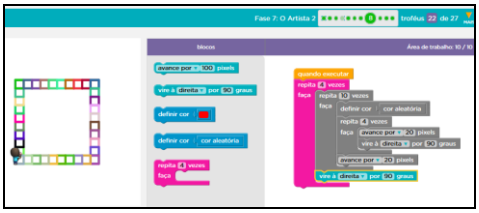
Usando apenas 3 blocos, você pode desenhar um quadrado com arestas de 20 pixels?



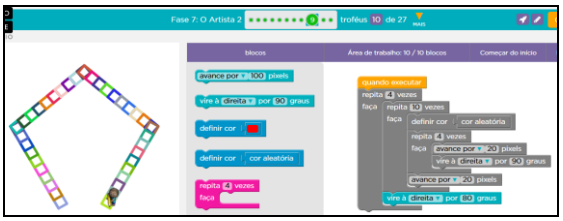
Este é o código do quadrado desenhado no último desafio. Você consegue repeti-lo para desenhar 10 quadrados adjacentes como uma escada? Dica: você só precisa de mais 2 blocos.



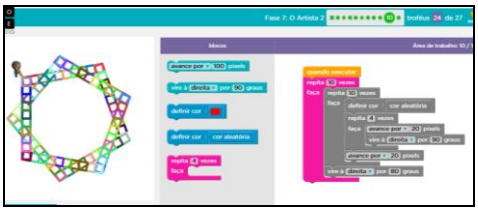
Esta é a solução do desafio anterior. Você consegue adicionar apenas mais 2 blocos para completar o desenho?



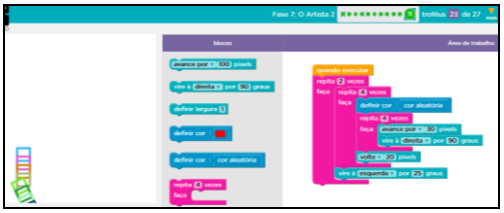
Esta é a solução do desafio anterior. Quantos graus você deve virar para completar o desenho? (Você provavelmente vai precisar chutar o valor algumas vezes)



Esta é a solução do desafio anterior. Quantas vezes você deve repetir para completar o desenho?

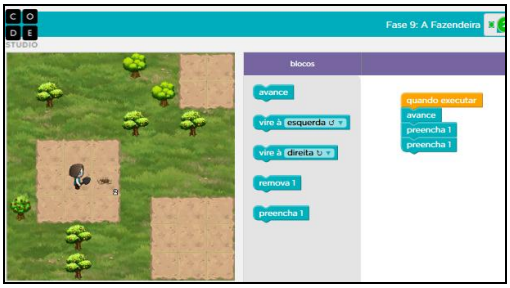


Altere os números nos blocos "vire" e "repita" para criar padrões diferentes. Ou então, experimente mudar o resto do código para desenhar o que quiser.



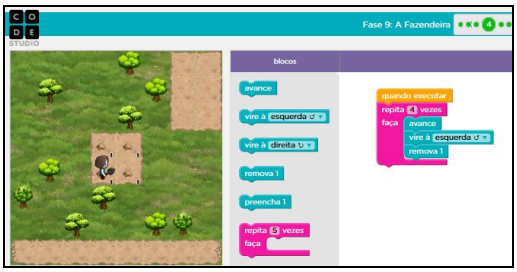
Olá, eu sou uma fazendeira! Preciso de sua ajuda para aplanar o terreno da minha fazenda, para prepará-lo para o plantio. Leve-me até o monte de terra e use o bloco "remova" para removê-lo.

Em seguida, leve-me para perto do buraco e preencha-o com DUAS pás cheias de terra, usando o bloco "preencha".



Leve-me até o monte de terra e me diga quantas pás devo remover, usando o menor número de blocos possível.

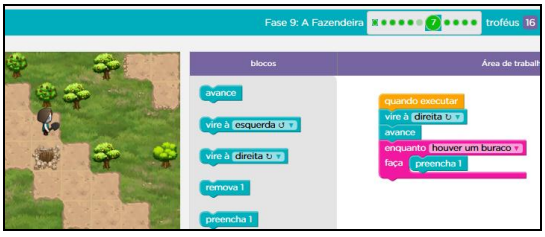
Você pode me ajudar a remover todos os quatro montes de terra? Dica: se puder, use um bloco de repetição.



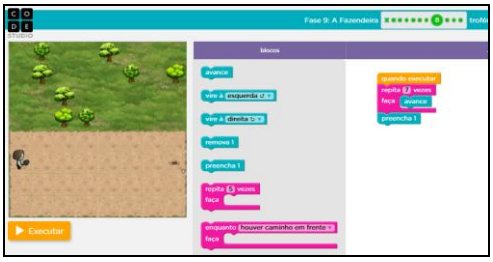
Ajude-me a encher todos esses buracos com 5 pás de terra. Dica: você pode usar um bloco "repita" dentro de um bloco "enquanto".

Remova todas as pilhas de terra usando o menor número de blocos possível. As novas opções de menu no bloco "enquanto" mostram se eu estou em uma pilha de terra ou em um buraco.

Uau, eu encontrei um buraco muito fundo! Não sei quantas pás de terra vamos precisar. Você pode escrever um programa para preenchê-lo completamente?



Ajude-me a preencher o buraco no final do terreno, usando o menor número de blocos possível.

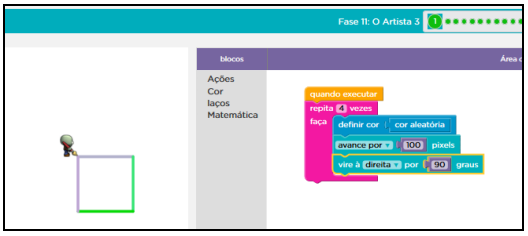


Faça com que eu remova todos esses montes de terra. Tente usar o menor número de blocos possível. Dica: tente usar um bloco "enquanto".

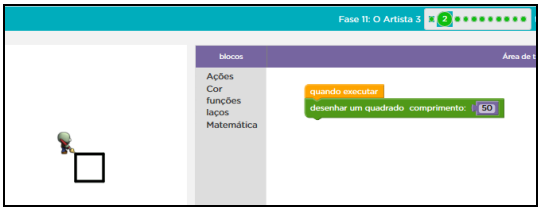
Estou a trabalhar até tarde e como tal está muito escuro e eu não consigo saber o tamanho dos montes de terra. Agora não tenho todas as opções que eu costumava ter no bloco "enquanto". Move-me ao longo do campo e se houver um monte de terra, remove-o.

Ainda está escuro lá fora. Mova-me ao longo do terreno. Se houver um monte de terra, remova-o, e se houver um buraco, preencha-o. Dica: use o menu de opções no bloco "se".

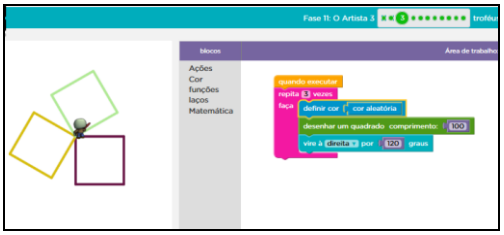
Olá! Mim zumbi artista. Mim gostar de desenhar! Me ajude a desenhar um quadrado com uma cor especial. Importante: você tem todos os blocos de antes, mas agora eles estão divididos em categorias.



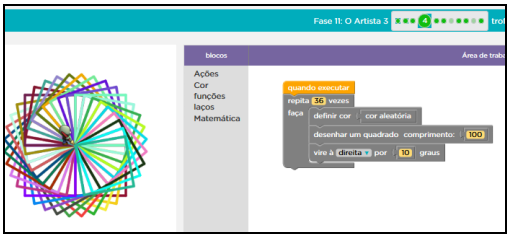
Bem-vindo ao uso de funções, as quais permitem que você defina novos blocos! Use o novo bloco "desenhe um quadrado", encontrado na categoria "Funções", para desenhar um pequeno quadrado verde de 50x50.



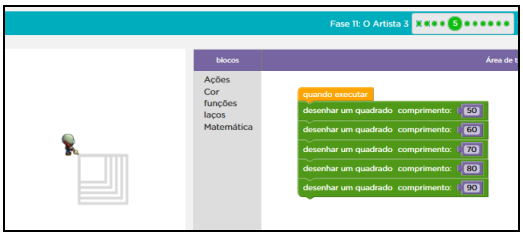
Use o bloco "repita" (na categoria "Laços") para desenhar 3 quadrados de tamanho 100, separados por 120 graus. Use 3 cores aleatórias.



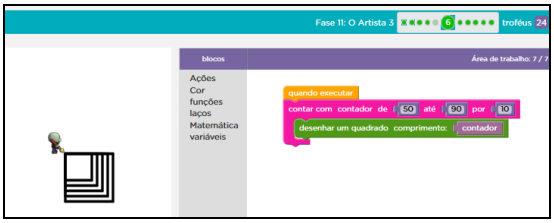
Agora vamos começar a ser extravagantes. Altera o código para desenhar 36 quadrados separados com 100 pixeis de largura separados por 10 graus. Dica: podes querer usar o controlo deslizante para me fazer ir mais rápido.



Desenha quadrados com lados de 50, 60, 70, 80 e 90 pixeis. Vais precisar de usar a função "desenhar um quadrado" cinco vezes.

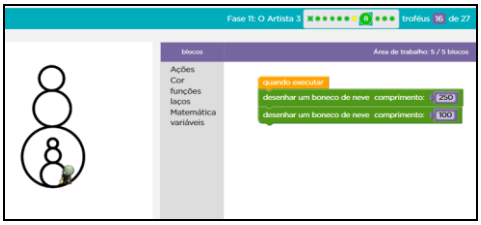


Bem, este programa irá utilizar um contador para desenhar os mesmos quadrados da última vez. Queres que o quadrado tenha o mesmo tamanho que o contador, então usa o bloco de "contador". Dica: É na categoria de variáveis.

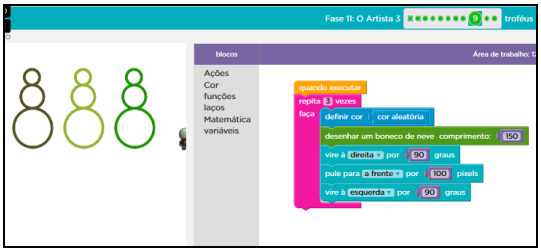


Aqui está um programa para desenhar uma espiral, mas os blocos estão desativados e como tal não vão correr. Faz um novo programa usando um bloco de "contar com" (na categoria Ciclos) e o bloco "contador" (na categoria Variáveis) para desenhar a mesma espiral.

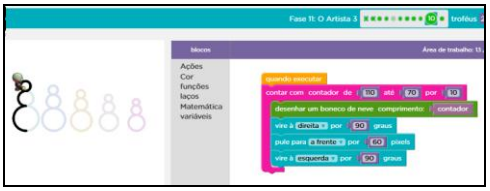
Há uma nova função "desenhar um boneco de neve" (na categoria de Funções). Desenha dois bonecos de neve, de 250 e 100 de altura.



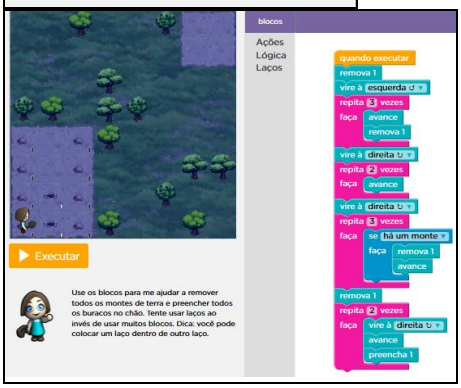
Este é um pouco complicado. Usa a função "desenhar um boneco de neve" e o bloco novo "salto para a frente" (na categoria de Ações). Desenha 3 bonecos de neve com cores diferentes, separados por 100 pixeis.



Usa um ciclo "contar com" para desenhar uma família de bonecos de neve, com alturas de 100, 110, 90, 80 e 70. Os bonecos de neve devem estar separados por 60 pixeis.



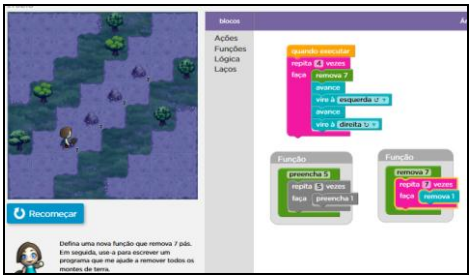
Tenta usar os blocos para me ajudar a remover todos os montes de terra e preenche todos os buracos no chão. Tenta usar ciclos ao invés de usar demasiados blocos. Dica: podes colocar um ciclo dentro de outro ciclo.



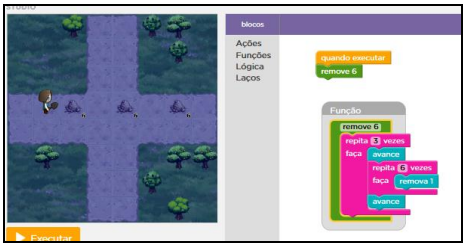
Definimos o nosso próprio bloco chamado "encher 5", que agora está na categoria de Funções. Usa-o para me ajudar a preencher este buraco.

Usa o bloco de Função "encher 5" para eu preencher todos os buracos. Terás que arrastar o bloco "encher 5" para fora da categoria de Funções.

Define uma nova função que remove 7 pazadas. Em seguida, usa-a para escrever um programa que me ajude a remover todas os montes.

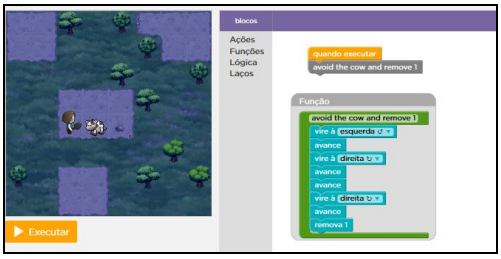


Cria uma nova função que remove 6 pazadas de uma pilha e usa-a para me ajudar a nivelar todos os montes.

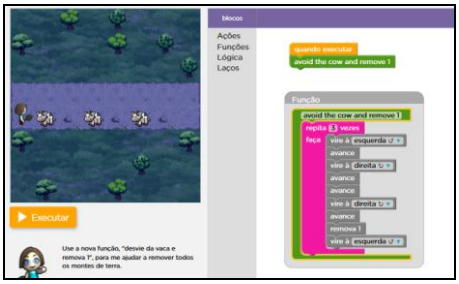


Usa duas novas funções, "preencher 8" e "remover 8", para ajudar-me a nivelar o monte e preencher o buraco.

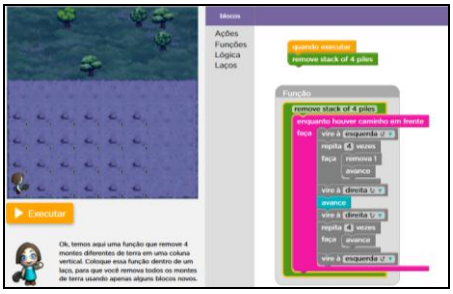
Há uma vaca no meu campo! Escreve uma nova função que me ajude a evitar a vaca e remover o monte. Coloca todos os blocos dentro da nova função.



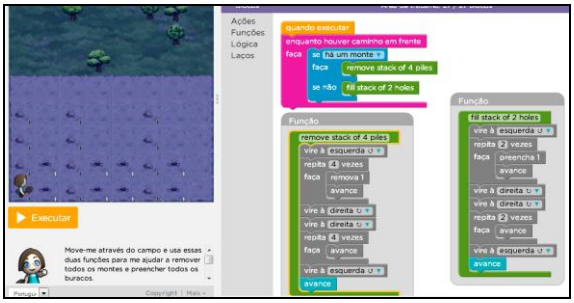
Usa esta nova função, "evitar a vaca e remover 1", para me ajudar a remover todos os montes.



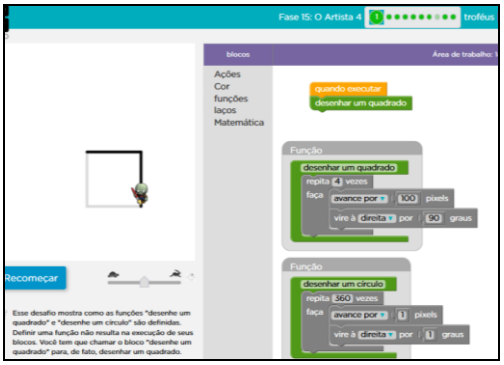
Ok, aqui está uma função que remove 4 montes diferentes de terra numa coluna vertical. Tente colocar essa função dentro de um ciclo, para remover todos os montes de terra usando apenas alguns novos blocos.



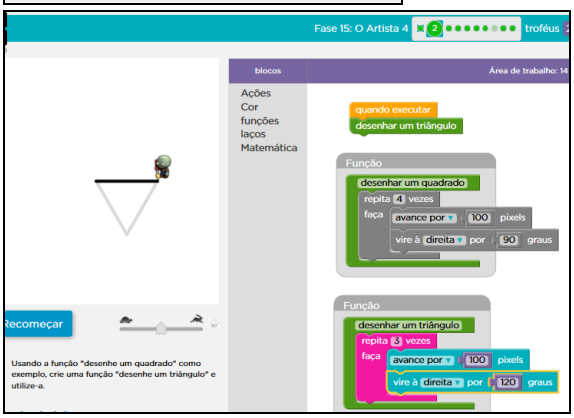
Move-me através do campo e usa essas duas funções para me ajudar a remover todos os montes e preencher todos os buracos.



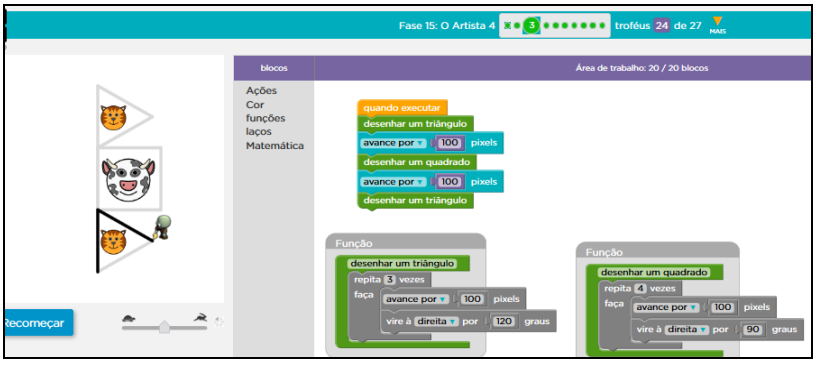
Este quebra-cabeças mostra como as funções "desenhar um quadrado" e "desenhar um círculo" são definidas. Definir uma função não executa os seus blocos. Você tem que puxar o bloco "desenhar um quadrado" para realmente desenhar um quadrado.



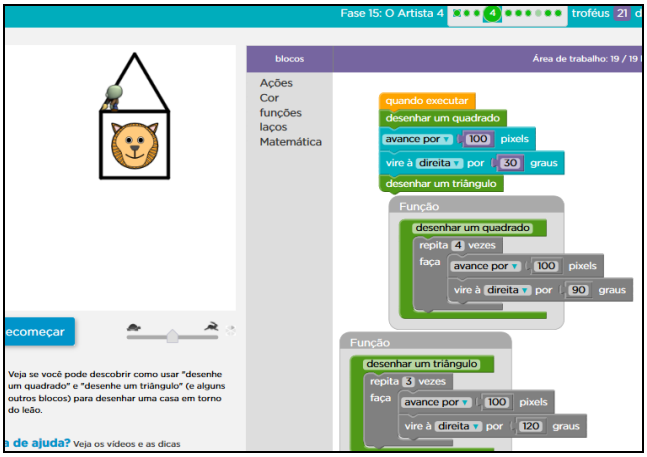
Usando a função "desenhar um quadrado" como um exemplo, cria uma função de "desenhar um triângulo" e usa-a.



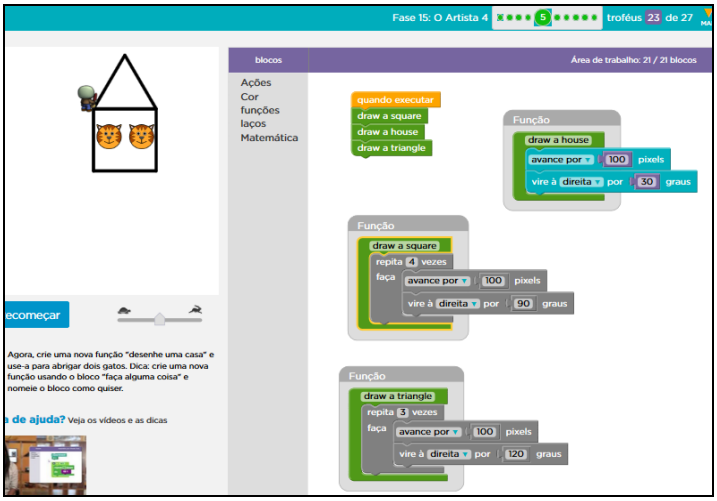
Desenha cercas triangulares em torno dos gatos e uma cerca quadrada em torno da vaca. Dica: testa o programa à medida que vais avançando.



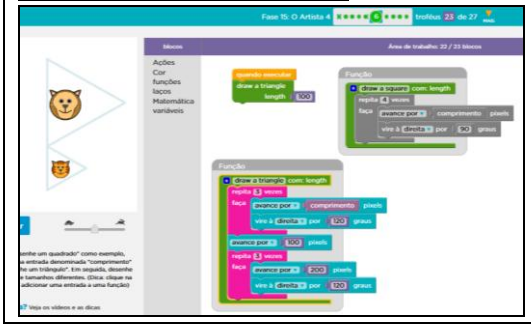
Tenta descobrir como usar "desenhar um quadrado" e "desenhar um triângulo" (e alguns outros blocos) para desenhar uma casa em torno do leão.



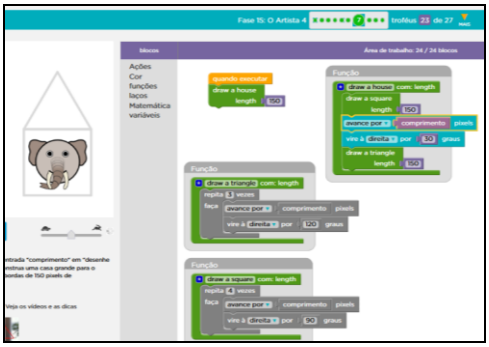
Agora cria uma nova função "desenhar uma casa" e usa-a para hospedar dois gatos. Dica: Cria uma nova função usando o bloco "fazer algo" e digite o teu próprio texto para o nome do bloco.



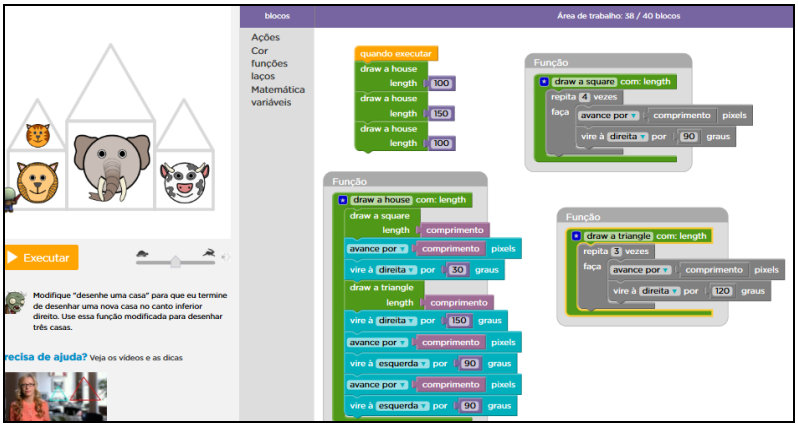
Usando "desenhar um quadrado" como exemplo, adiciona uma entrada denominada "comprimento" para "desenhar um triângulo". Em seguida, desenha triângulos em tamanhos diferentes. (Dica: clica na estrela para adicionar uma entrada para uma função)



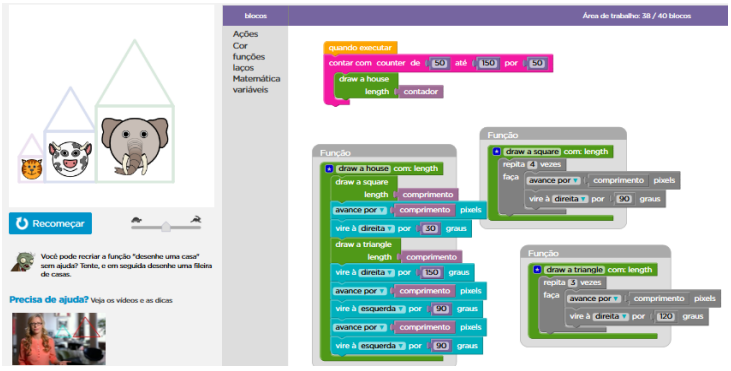
Adiciona uma entrada chamada "altura" para "desenhar uma casa" e constrói uma casa grande para o elefante (com arestas de 150 pixeis de comprimento)



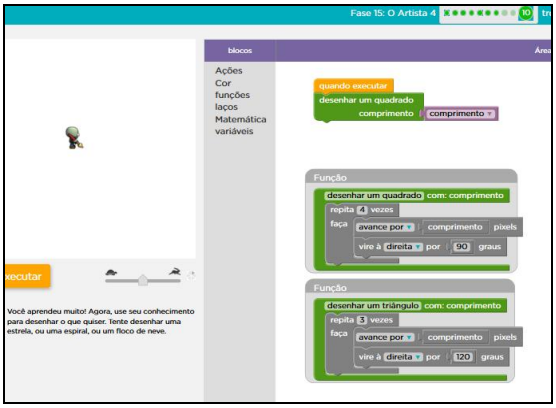
Modifique "desenhar uma casa" para que eu termine de desenhar uma nova casa no canto inferior direito. Use essa função modificada para desenhar três casas.



Podes recriar a função "desenhar uma casa" sem ajuda? Tenta, e depois desenha uma fileira de casas.



Aprendeste muito! Agora, já consegues desenhar o que quiseres. Tenta desenhar uma estrela, ou uma espiral ou um floco de neve extravagante.



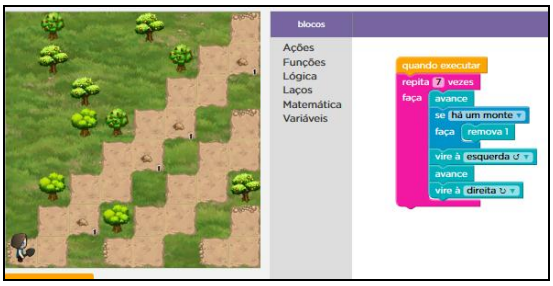
Bem-vindo à depuração! O meu código não funciona bem. Consegues detectar o problema e corrigi-lo para que eu possa ficar com o meu campo plano e pronto para começar a plantar?

O que há de errado com este programa? Como deves corrigi-lo para que eu possa remover aquele monte?

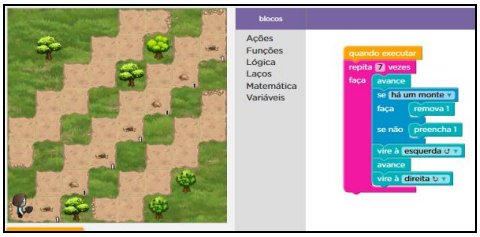
Corrige este programa por mim para que o meu campo fique plano e bonito.

Como deverás alterar este programa para me ajudar a preencher o buraco?

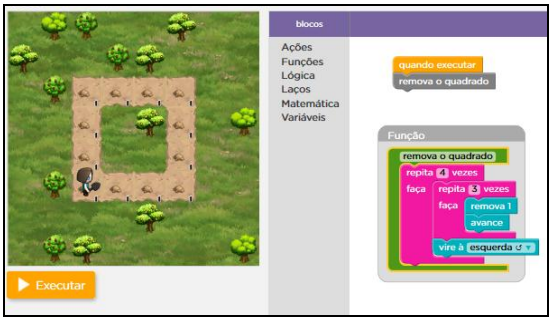
Ajuda! Corrige o programa para que o meu campo fique completamente plano.



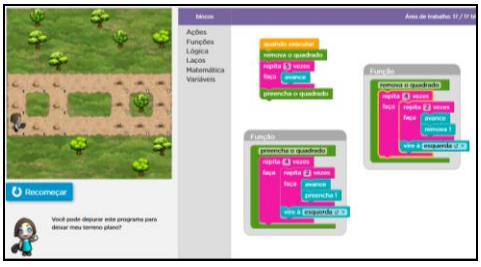
Corrige este programa para que possa tapar os buracos e remover todas os montes.



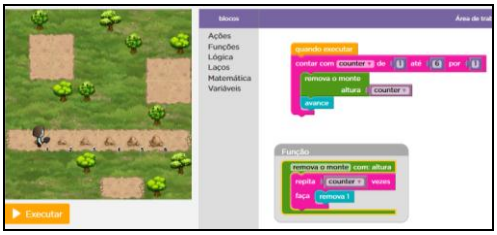
O que deverás mudar neste programa para me ajudar a remover todos os montes?



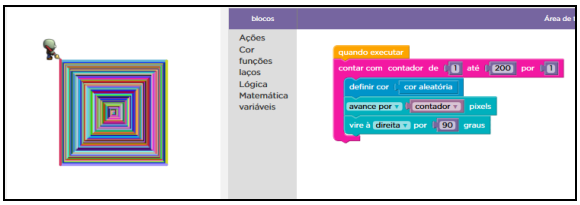
Podes corrigir este programa por mim para que eu possa ficar com o campo plano?



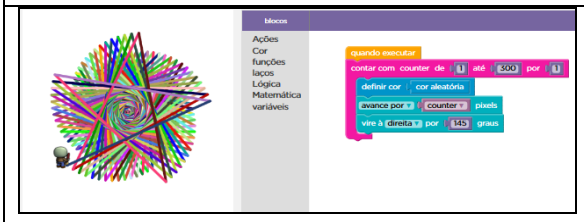
Edita este programa para que eu possa remover todas os montes do meu campo.

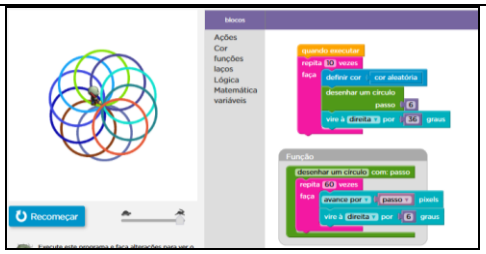


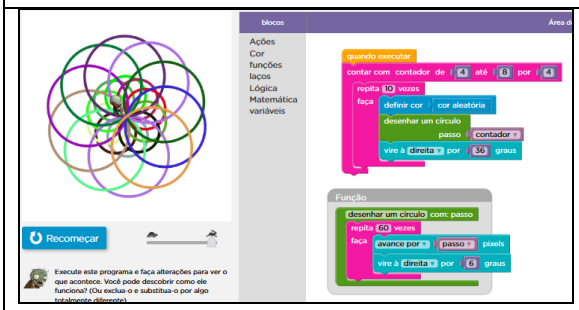
Tenta executar este programa e faz as alterações para ver o que acontece. Podes descobrir como funciona? (Ou apagá-lo e substituí-lo por algo totalmente diferente)

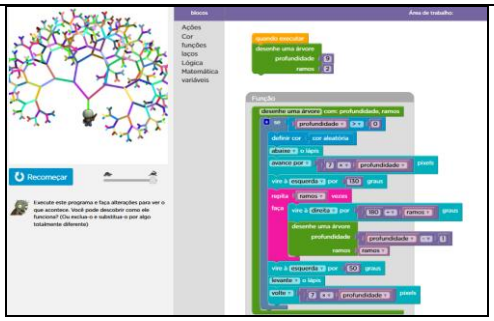


Tenta executar este programa e faz as alterações para ver o que acontece. Podes descobrir como funciona? (Ou apagá-lo e substituí-lo por algo totalmente diferente)









ANEXOS

ANEXO A – Autorização da instituição participante

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
Instituto Federal de Educação, Ciência e Tecnologia Sul-rio-grandense

CONSENTIMENTO PARA O ESTUDO

Pelo presente Termo de Consentimento, declaro que fui informado de forma clara e detalhada dos objetivos da metodologia e da finalidade do Projeto de Pesquisa intitulada: **ANÁLISE DA POTENCIALIDADE DE FERRAMENTAS GRÁFICAS NO ENSINO NA DISCIPLINA DE ALGORITMOS** desenvolvida pela Professora Carmen Vera Scorsatto Brezolin , como parte do trabalho de produção da dissertação de mestrado no PPGEDU/UPF, na linha de pesquisa **Processos Educativos e Linguagem**, sob a orientação da Profa. Dra. Neiva Inês Grando.

Tenho o conhecimento de que receberei resposta a qualquer dúvida sobre os procedimentos e outros assuntos relacionados com esta pesquisa. Também estou ciente que os participantes do estudo o farão de forma voluntária e terão as suas identidades reveladas mediante consentimento individualizado.

Portanto, concordo com a participação do IFSul – Campus Passo Fundo – neste estudo, bem como autorizo, para fins exclusivamente desta pesquisa, a utilização de informações e imagens realizadas nesta instituição.

Passo Fundo – RS, 29 de setembro de 2014.

Alexandre Pitol Boeira
Diretor Geral
IFSul Campus Passo Fundo

ANEXO B – Termo de consentimento livre e esclarecido (folha 1)

UNIVERSIDADE DE PASSO FUNDO
Faculdade de Educação
Mestrado em Educação

Você está sendo convidado(a) a participar de uma pesquisa de dissertação de mestrado sobre “análise da potencialidade de ferramentas gráficas no ensino na disciplina de algoritmos”, de responsabilidade da pesquisadora Carmen Scorsatto Brezolin , sob orientação da Profa. Dra. Neiva Ignês Grando.

Esta pesquisa se justifica pelas dificuldades apresentadas pelos alunos na transcrição do raciocínio matemático para o ambiente computacional. Dessa forma, observa-se que a disciplina de algoritmos apresenta um alto índice de desistência e reprovação.

O objetivo principal do estudo é investigar o potencial de ferramentas gráficas para o ensino da disciplina de algoritmos.

A sua participação na pesquisa dar-se-á nas seguintes modalidades: utilização de um *software* gráfico direcionado a aprendizagem de algoritmos e outra para responder avaliações quanto ao seu aprendizado na disciplina

Entende-se que a sua participação na pesquisa não causara danos a sua integridade, se for identificado algum sinal de desconforto psicológico da sua participação na pesquisa, o pesquisador compromete-se em orientá-lo(a) e encaminhá-lo(a) para os profissionais especializados na área. Ao você participar da pesquisa irá contribuir na produção de conhecimento para entender, as dificuldades apresentadas na disciplina de algoritmos em sala de aula, sendo você e todos os demais envolvidos no âmbito escolar beneficiados diretamente pelos resultados da pesquisa.

Você terá a garantia de receber esclarecimentos sobre qualquer dúvida relacionada à pesquisa e poderá ter acesso aos seus dados em qualquer etapa do estudo. Sua participação nessa pesquisa não é obrigatória e você pode desistir a qualquer momento, retirando seu consentimento. Caso tenha alguma despesa relacionada à pesquisa, você terá o direito de ser ressarcido(a) e você não receberá pagamento pela sua participação no estudo.

ANEXO B – Termo de consentimento livre e esclarecido (folha 2)

As suas informações serão gravadas e posteriormente destruídas. Os dados relacionados à sua identificação não serão divulgados. Os resultados da pesquisa serão divulgados, mas você terá a garantia do sigilo e da confidencialidade dos dados. Caso você tenha dúvidas sobre o comportamento do pesquisador ou sobre as mudanças ocorridas na pesquisa que não constam no TCLE, e caso se considerar prejudicado(a) na sua dignidade e autonomia, você poderá entrar em contato com a pesquisadora Carmen Scorsatto Brezolin, pelo telefone (54) 99668627, ou com o curso Mestrado em Educação, ou também pode consultar o Comitê de Ética em Pesquisa da UPF, pelo telefone (54) 3316-8157, no horário das 08h às 12h e das 13h30min às 17h30min, de segunda a sexta-feira. Desde já, agradecemos a sua colaboração e solicitamos a sua autorização, que será assinada em duas vias, sendo que uma ficará com você e outra com a pesquisadora.

Dessa forma, concordo em participar da pesquisa, de acordo com as explicações e orientações citadas.

Passo Fundo, ____ de ____ de 2014.

Nome do (a) participante: _____

Assinatura: _____

Nome do pesquisador: Carmen Scorsatto Brezolin.

Assinatura: _____

ANEXO C – Parecer substanciado do Comitê de Ética (folha 1)

UNIVERSIDADE DE PASSO FUNDO/ PRÓ-REITORIA DE PESQUISA E PÓS-

Plataforma Brasil

PARECER CONSUBSTANCIADO DO CEP

DADOS DO PROJETO DE PESQUISA

Título da Pesquisa: ANÁLISE DA POTENCIALIDADE DE FERRAMENTAS GRÁFICAS NO ENSINO NA DISCIPLINA DE ALGORITMOS

Pesquisador: Carmen vera Scorsatto Brezolin

Área Temática:

Versão: 2

CAAE: 37528214.0.0000.5342

Instituição Proponente: Universidade de Passo Fundo/Vice-Reitoria de Pesquisa e Pós-Graduação

Patrocinador Principal: Financiamento Próprio

DADOS DO PARECER

Número do Parecer: 860.081

Data da Relatoria: 09/11/2014

Apresentação do Projeto:

O projeto de dissertaçao de Mestrado em Educação apresenta como problema de investigação "avaliar as implicações do uso da ferramenta gráficas como estratégia para potencializar e qualificar o processo de aprendizado dos alunos da disciplina de Lógica e Algoritmos (LA) do Curso Superior de Tecnologia em Sistemas para a Internet (TSPI), oferecido pelo Instituto Federal Sul-Rio-Grandense (IFSUL) do campus Passo Fundo." A constatação da autora é que os alunos encontram muitas dificuldades para a transcrição do raciocínio matemático para o ambiente computacional e o indice de reprovação nesta disciplina é bastante alto. O objetivo é avaliar o potencial da ferramentas gráficas como meio de superação das dificuldades de aprendizagem. Trata-se de um estudo exploratório-descritivo e qualitativo, que terá como público-alvo alunos do 1° nível do referido curso. Os dados empíricos serão coletados por meio de avaliações participativas e não-participativas e posteriormente analisados qualitativamente

Objetivo da Pesquisa:

Avaliar o potencial das ferramenta gráficas como estratégia de ensino-aprendizagem para o

Endereço: BR 285- Km 171 Campus I - Centro Administrativo
Bairro: Divisão de Pesquisa / São José **CEP:** 99.010-970
UF: RS **Município:** PASSO FUNDO
Telefone: (54)3316-8157 **E-mail:** cep@upf.br

ANEXO C – Parecer substanciado do Comitê de Ética (folha 2)

ensino da disciplina de Lógica e Algoritmos.

Avaliação dos Riscos e Benefícios:

Entende-se que a analise dos dados empíricos coletados por meio de avaliações participativas e não-participativas não causara danos a integridade
dos participantes da pesquisa.

Benefícios:

A pesquisa irá gerar conhecimento para entender, podem auxiliar no processo de aprendizagem dos alunos da disciplina de Algoritmos.

Comentários e Considerações sobre a Pesquisa:

Trata-se de um estudo exploratório-descritivo e qualitativo, que terá como público-alvo alunos do 1° nível do Curso Superior de Tecnologia
Tecnologias em Sistemas para a Internet (TSPI) do Instituto Federal Sul-Riograndense (IFSUL) do campus Passo Fundo. Os dados empíricos serão coletados por meio de avaliações participativas e não-participativas e posteriormente analisados qualitativamente

Considerações sobre os Termos de apresentação obrigatória:

Os direitos fundamentais do(s) participante(s) foi(ram) garantido(s) no projeto e no TCLE. O protocolo foi instruído e apresentado de maneira completa e adequada. Os compromissos do (a) pesquisador (a) e das instituições envolvidas estavam presentes. O projeto foi considerado claro em seus aspectos científicos, metodológicos e éticos.

Recomendações:

Após o término da pesquisa, o CEP UPF solicita:

a) A devolução dos resultados do estudo ao(s) sujeito(s) da pesquisa ou a instituição que forneceu os dados;

b) Enviar o relatório final da pesquisa, pela plataforma, utilizando a opção, no final da página, "Enviar Notificação" + relatório final.

ANEXO C – Parecer substanciado do Comitê de Ética (folha 3)

UNIVERSIDADE DE PASSO FUNDO/ PRÓ-REITORIA DE PESQUISA E PÓS-

Plataforma Brasil

Continuação do Parecer: 860.081

Conclusões ou Pendências e Lista de Inadequações:

Diante do exposto, este Comitê, de acordo com as atribuições definidas na Resolução n. 466/12, do Conselho Nacional da Saúde, Ministério da Saúde, Brasil, manifesta-se pela aprovação do projeto de pesquisa na forma como foi proposto.

Situação do Parecer:

Aprovado

Necessita Apreciação da CONEP:

Não

Considerações Finais a critério do CEP:

PASSO FUNDO, 05 de Novembro de 2014

Assinado por:
Nadir Antonio Pichler
(Coordenador)

CIP – Catalogação na Publicação

B848c Brezolin, Carmen Vera Scorsatto
Contribuições da ferramenta gráfica Blockly no processo ensino-apredizagem na disciplina de algoritmos / Carmen Vera Scorsatto Brezolin. – 2016.
158 f. : il., color. ; 30 cm.

Orientadora: Profa. Dra. Neiva Ignês Grando.
Dissertação (Mestrado em Educação) – Universidade de Passo Fundo, 2016.

1. Algoritmos – Estudo e ensino. 2. Aprendizagem. 3. Tecnologia educacional. I. Grando, Neiva Ignês, orientadora. II. Título.

CDU: 37:004

Catalogação: Bibliotecária Schirlei T. da Silva Vaz - CRB 10/1364