Research, Society and Development, v. 11, n. 9, e58811929683, 2022
(CC BY 4.0) | ISSN 2525-3409 | DOI: http://dx.doi.org/10.33448/rsd-v11i9.29683

Ensino de lógica de programação através de problemas abstratos com conteúdo de geometria plana

Programming logic teaching through abstract problems with plane geometry content

Enseñanza de la lógica de programación a través de problemas abstractos con contenido de geometría plana

Recebido: 27/04/2022 | Revisado: 13/05/2022 | Aceito: 13/07/2022 | Publicado: 19/07/2022

Fernanda Regebe Castro
ORCID: https://orcid.org/0000-0002-7368-4517
Instituto Federal da Bahia, Brasil
E-mail: fernandaregebe@gmail.com
Fábio Marques da Cruz
ORCID: https://orcid.org/0000-0002-5964-8149
Instituto Federal da Bahia, Brasil
E-mail: fabiomacz@gmail.com

Resumo
Este artigo relata a aplicação de uma oficina com o objetivo de propor problemas abstratos em lógica de programação a partir de conteúdos de geometria plana. A ferramenta LOGO foi utilizada para o desenvolvimento de algoritmos, pelos estudantes, que tinham por finalidade construir figuras a partir da aplicação dos postulados da geometria plana. Estas figuras eram traçadas a partir do rastro da tartaruga, que se movimenta através de uma sequência de comandos de um algoritmo dado. Os resultados obtidos mostram que não houve transferência direta dos conhecimentos da geometria para a construção de algoritmos, mas é necessária a construção de um novo conhecimento que engloba estes dois domínios de conteúdo. Observamos dificuldades na interpretação e divisão dos problemas e na capacidade de abstração. Também concluímos que precisamos focar nossa práxis pedagógica em atividades que desenvolvam o pensamento abstrato dos discentes, pois a área de programação de algoritmos exige habilidades com alto nível de abstração e este gap acaba se tornando um desafio para o desenvolvimento dos alunos.
Palavras-chave: Lógica de programação; Problemas abstratos; Transposição de conhecimento; Ensino.

Abstract
This article reports on the application of a workshop with the objective of proposing abstract problems in programming logic based on plane geometry content. The LOGO tool was used for the development of algorithms by students, whose purpose was to build figures from the application of the postulates of plane geometry. These objects were drawn from the turtle's trail, which moves through a sequence of commands from a given algorithm. The results obtained show that there was no direct transfer of knowledge from geometry to the construction of algorithms, but it is necessary to build new knowledge that encompasses these two content domains. We observed difficulties in the interpretation and division of problems and in the capacity for abstraction. We also concluded that we need to focus our pedagogical praxis on activities that develop the students 'abstract thinking, as the area of algorithm programming requires skills with a high level of abstraction and this gap ends up becoming a challenge for the students' development.
Keywords: Programming logic; Abstract problems; Knowledge transposition; Teaching.

Resumen
Este artículo reporta la aplicación de un taller con el objetivo de proponer problemas abstractos en lógica de programación basados en contenidos de geometría plana. Se utilizó la herramienta LOGO para el desarrollo de algoritmos, por parte de los estudiantes, cuyo propósito fue construir figuras a partir de la aplicación de los postulados de la geometría plana. Estas cifras se trazaron a partir del rastro de la tortuga, que se mueve a través de una secuencia de comandos de un algoritmo determinado. Los resultados obtenidos muestran que no hubo una transferencia directa de conocimiento desde la geometría hacia la construcción de algoritmos, sino que es necesario construir nuevo conocimiento que abarque estos dos dominios de contenido. Observamos dificultades en la interpretación y división de problemas y en la capacidad de abstracción. También concluimos que necesitamos enfocar nuestra praxis pedagógica en actividades que desarrollen el pensamiento abstracto de los estudiantes, ya que el área de programación de algoritmos requiere habilidades con un alto nivel de abstracción y esta brecha termina convirtiéndose en un desafío para el desarrollo de los estudiantes.
Palabras clave: Lógica de programación; Problemas abstractos; Transferencia de conocimiento, Enseñanza.

1. Introdução

O componente curricular de lógica de programação é um elemento base para os cursos na área de tecnologia da informação, ciência da computação, análise e desenvolvimento de sistemas, técnico em informática, entre outros. Qualquer que seja o nível (técnico, tecnólogo ou superior) do curso, esta disciplina é ministrada sempre no primeiro período e é responsável por grandes níveis de repetência e evasão. Um dos principais objetivos da disciplina é promover a capacidade para o aluno resolver problemas do cotidiano utilizando instruções estruturadas logicamente e executadas no computador por meio de linguagem de programação (Pereira Júnior, & Rapkiewicz, 2004). Os algoritmos são sequencias lógicas de passos para se realizar uma tarefa ou trabalho (Shitsuka, et al, 2019).

Existem muitas pesquisas que estudam o ensino de lógica de programação e as dificuldades dos alunos. Alguns atribuem à dificuldade de interpretação do problema a ser resolvido, o entendimento do enunciado, mesmo antes da dificuldade de interpretação de algum tipo de representação (Falkembach, et al, 2003) , à dificuldade de resolução de problemas (Martins & Correia, 2003), à dificuldade na construção do algoritmo (Campos, 2009), a dificuldade na aplicação das estruturas de controle dos algoritmos (Gomes, et al., 2008) e a dificuldade de assimilar as abstrações envolvidas no processo de ensino-aprendizagem de fundamentos de programação (Rodrigues, 2002). Gomes e Mendes (2014) argumentam que muitos estudantes utilizam estratégias de memorização, ou seja, leitura e observação da resolução dos exercícios, que não são suficientes para aprender programação. E defendem que é necessário o engajamento intensivo na prática de resolução de problemas, examinando as dificuldades de cada problema e procurando a solução destes. Além de tudo isso, o próprio ensino de programação de computadores não é fácil e, devido a isso, muitas universidades discutem com frequência seus currículos dos cursos da área de Computação, em busca de alternativas para diminuir o índice de evasão deste curso (Bigolin et al, 2019).

Regebe (2017) analisou a evolução das habilidades cognitivas de raciocínio lógico em tecnologia da informação aplicando 9 instrumentos (testes de conhecimento) durante um ano letivo em turmas iniciais do curso técnico em informática da modalidade integrada, classificando os itens destes testes em 1) itens de conhecimento matemático puro, 2) itens de conteúdos de estrutura de programação e 3) itens híbridos. Os itens de conhecimento matemático puro foram considerados os mais fáceis para os alunos, os itens de estrutura de programação tiveram dificuldade intermediária e os itens de conteúdo híbrido foram considerados os mais difíceis durante todo o ano. Um item de conhecimento matemático puro tornava-se muito mais difícil quando a resolução era exigida através de um algoritmo, tornando-se assim um item híbrido (Regebe (2017); (Regebe, & Amantes, 2019). Como uma das considerações, a autora sugeriu estratégias de ensino da disciplina para a condução do conteúdo. Os problemas algorítmicos exigem que os alunos tenham o raciocínio de transposição do problema ou que eles tenham conhecimento suficiente em vários conteúdos matemáticos em que eles têm dificuldade, como foi levantado por Gomes et *al.* (2006), quando relacionaram as dificuldades dos alunos ao background em conhecimentos matemáticos. Sendo assim, a autora interpretou que a solução destas questões não exige somente uma transferência direta de conceitos para outra dimensão e sim a construção de um novo conhecimento (Fensham, 1994).

Diante da dificuldade dos alunos em trabalhar com conceitos abstratos, e diante da necessidade de estratégias que envolvam o desenvolvimento da capacidade de abstração e estratégias de resolução de problemas pelo aluno para resolver problemas matemáticos, propomos estratégias que levem o aluno a trabalhar com problemas matemáticos abstratos antes mesmo das etapas de transposição do problema e implementação algorítmica. Esta estratégia seria uma forma de desenvolver o pensamento abstrato nos discentes, abstração esta necessária para a construção do conhecimento de transposição de problemas matemáticos em soluções algorítmicas.

Neste artigo, apresentamos a aplicação de uma oficina com o objetivo de propor problemas abstratos em lógica de programação envolvendo conteúdos e problemas também abstratos de geometria plana. A oficina fez parte da programação da Semana Nacional de Ciência e Tecnologia (SNCT) de 2019 do Instituto Federal de Ciência e Tecnologia (IFBA) Campus

Camaçari.

Do ponto de vista metodológico, o artigo enquadra-se no gênero de relato de experiência, que é uma narração detalhada de experiências vividas sob o ponto de vista de quem relata (narrador). Os resultados são narrados de forma descritiva, interpretados e discutidos à luz de um marco teórico. (Grollmus; Tarrés, 2015).

A ferramenta LOGO foi utilizada para o desenvolvimento de algoritmos, pelos estudantes, que tinham por finalidade formar figuras a partir da aplicação dos postulados da geometria plana. Os algoritmos deveriam conter sequências de comandos que controlavam o movimento de uma tartaruga (cursor). Através do rastro deixado pela tartaruga as figuras geométricas seriam desenhadas. No entanto, para que os algoritmos pudessem ser criados pelos alunos, a etapa anterior ao desenho (algoritmo com conjunto de comandos que formam o desenho) seria descobrir todas as medidas e ângulos da figura a ser desenhada através de deduções baseadas nos postulados matemáticos da geometria plana.

2. Metodologia

A oficina de Ensino de Lógica de Programação através de problemas abstratos com conteúdos de Geometria Plana foi oferecida na programação da Semana Nacional de Ciência e Tecnologia (SNCT) do ano de 2019 do Campus Camaçari. A SNCT é uma ação nacional proposta pelo Ministério da Ciência, Tecnologia, Inovações e Comunicações e que tem como objetivo a divulgação e popularização da ciência na sociedade. Durante a semana, que ocorre de forma concomitante em universidade, institutos, e escolas em todo o país, as portas destas instituições se abrem para toda a comunidade e são apresentados trabalhos, pesquisas, oferecidas oficinas, minicursos, entre outros. Nosso curso foi aberto a todo público interno e externo e tinha como requisito ter estudado o assunto de geometria plana, no entanto, só se inscreveram e cursaram alunos do próprio campus Camaçari das modalidades integrada e superior.

As inscrições foram efetuadas 20 minutos antes do início do curso e a turma foi formada por 16 alunos, sendo 13 da modalidade integrada e 3 da modalidade superior (Tabela 1). Apesar de ter sido requisito para o curso ter estudado geometria plana, 3 alunos ainda se inscreverem sem terem visto o conteúdo, então foram retirados da amostra (alunos 4 e 6 da modalidade integrada - curso de TI, e aluno 13 da licenciatura em matemática). Os alunos do curso integrado ainda cursavam o primeiro ano e por isto não tinham visto geometria plana. Já o aluno do superior apesar de ter concluído o ensino médio (EM) no município de Camaçari, afirmou não ter visto o assunto nem na licenciatura nem mesmo na época que cursou o EM. Este fato mostra a deficiência da rede pública quando muitas vezes faltam professores para as disciplinas básicas, acarretando falta de aulas, não cumprimento do programa curricular ou até mesmo desvio de função de professores de outras áreas que são obrigados a lecionar disciplinas que não tem familiaridade e acabam por evitar assuntos em que não se sintam confortáveis.

Tabela 1- Dados dos alunos matriculados na oficina.

ALUNO	MODALIDADE	CURSO	SÉRIE	GOSTA DE MATEMÁTICA?	VOCÊ TEM FACILIDADE EM MATEMÁTICA?	GOSTA DE RACIOCÍNIO LÓGICO?	VOCÊ TEM FACILIDADE EM RACIOCÍNIO LÓGICO?	VIU GEOMETRIA PLANA?	GEOMETRIA EXIGE RACIOCÍNIO LÓGICO?	JÁ PROGRAMOU?	LINGUAGENS
1	INTEGRADO	TI	4	GOSTO POUCO	TENHO POUCA	GOSTO	INDIFERENTE	SIM	EXIGE	SIM	JAVA C# HTML SQL
2	SUPERIOR	COMPUTAÇÃO	3	GOSTO	TENHO	GOSTO	TENHO	SIM	EXIGE	SIM	C SQL RUBY
3	SUPERIOR	COMPUTAÇÃO	3	NÃO GOSTO	NÃO TENHO	GOSTO	INDIFERENTE	SIM	EXIGE	SIM	C SQL JAVA PHP
4	INTEGRADO	ELETRO	1	GOSTO POUCO	TENHO POUCA	INDIFERENTE	NÃO TENHO	NÃO		NÃO	NÃO SE APLICA
5	INTEGRADO	TI	1	GOSTO	TENHO	GOSTO	TENHO POUCA	SIM	EXIGE	SIM	PORTUGOL E PASCALZIM
6	INTEGRADO	TI	1	GOSTO POUCO	TENHO POUCA	GOSTO	TENHO	NÃO		NÃO	NÃO SE APLICA
7	INTEGRADO	TI	4	GOSTO	TENHO	GOSTO MUITO	TENHO	SIM	EXIGE MUITO	SIM	JAVA C# PHP PORTUGOL
8	INTEGRADO	TI	4	GOSTO MUITO	TENHO	GOSTO	TENHO POUCA	SIM	EXIGE	SIM	C# PHP JAVA JAVASCRIPT PORTUGOL
9	INTEGRADO	TI	4	GOSTO	TENHO	GOSTO	TENHO	SIM	POUCO	SIM	JAVA C#
10	INTEGRADO	TI	4	GOSTO POUCO	NÃO TENHO	GOSTO	TENHO POUCA	SIM	EXIGE MUITO	SIM	PASCAL JAVA SQL HTML C#
11	INTEGRADO	TI	4	GOSTO	TENHO POUCA	GOSTO POUCO	TENHO POUCA	SIM	EXIGE MUITO	SIM	PASCAL JAVA C# SQL HTML
12	INTEGRADO	TI	4	GOSTO POUCO	TENHO POUCA	GOSTO MUITO	TENHO	SIM	EXIGE	SIM	JAVA C# SQL
13	SUPERIOR	MATEMÁTICA	5	GOSTO	NÃO TENHO	GOSTO	TENHO	NÃO	EXIGE	SIM	PASCAL
14	INTEGRADO	TI	4	GOSTO	TENHO POUCA	GOSTO	TENHO POUCA	SIM	EXIGE MUITO	SIM	JAVA E C#
15	INTEGRADO	TI	4	INDIFERENTE	INDIFERENTE	GOSTO	INDIFERENTE	SIM	EXIGE	SIM	C# JAVA PPASCAL
16	INTEGRADO	TI	2	GOSTO MUITO	TENHO MUITA	GOSTO MUITO	TENHO	SIM	EXIGE	SIM	JAVA PZIM MY SQL

Fonte: Dados da pesquisa.

Após a retirada dos 3 alunos que não estabeleciam ao critério inicial (ter estudado geometria plana), nossa amostra final passou a ser formada por 13 alunos, conforme pode ser visto na Tabela 2. Destes alunos, 11 são da modalidade integrada (Curso Técnico em TI) e 2 da modalidade superior (Ciência da Computação).

Tabela 2 – Amostra final após a retirada dos alunos que não tinham visto o assunto de geometria plana.

ALUNO	MODALIDADE	CURSO	SÉRIE	GOSTA DE MATEMÁTICA?	VOCÊ TEM FACILIDADE EM MATEMÁTICA?	GOSTA DE RACIOCÍNIO LÓGICO?	VOCÊ TEM FACILIDADE EM RACIOCÍNIO LÓGICO?	VIU GEOMETRIA PLANA?	GEOMETRIA EXIGE RACIOCÍNIO LÓGICO?	JÁ PROGRAMOU?	LINGUAGENS
1	INTEGRADO	TI	4	GOSTO POUCO	TENHO POUCA	GOSTO	INDIFERENTE	SIM	EXIGE	SIM	JAVA C# HTML SQL
2	SUPERIOR	COMPUTAÇÃO	3	GOSTO	TENHO	GOSTO	TENHO	SIM	EXIGE	SIM	C SQL RUBY
3	SUPERIOR	COMPUTAÇÃO	3	NÃO GOSTO	NÃO TENHO	GOSTO	INDIFERENTE	SIM	EXIGE	SIM	C SQL JAVA PHP
5	INTEGRADO	TI	1	GOSTO	TENHO	GOSTO	TENHO POUCA	SIM	EXIGE	SIM	PORTUGOL E PASCALZIM
7	INTEGRADO	TI	4	GOSTO	TENHO	GOSTO MUITO	TENHO	SIM	EXIGE MUITO	SIM	JAVA C# PHP PORTUGOL
8	INTEGRADO	TI	4	GOSTO MUITO	TENHO	GOSTO	TENHO POUCA	SIM	EXIGE	SIM	C# PHP JAVA JAVASCRIPT PORTUGOL
9	INTEGRADO	TI	4	GOSTO	TENHO	GOSTO	TENHO	SIM	POUCO	SIM	JAVA C#
10	INTEGRADO	TI	4	GOSTO POUCO	NÃO TENHO	GOSTO	TENHO POUCA	SIM	EXIGE MUITO	SIM	PASCAL JAVA SQL HTML C#
11	INTEGRADO	TI	4	GOSTO	TENHO POUCA	GOSTO POUCO	TENHO POUCA	SIM	EXIGE MUITO	SIM	PASCAL JAVA C# SQL HTML
12	INTEGRADO	TI	4	GOSTO POUCO	TENHO POUCA	GOSTO MUITO	TENHO	SIM	EXIGE	SIM	JAVA C# SQL
14	INTEGRADO	TI	4	GOSTO	TENHO POUCA	GOSTO	TENHO POUCA	SIM	EXIGE MUITO	SIM	JAVA E C#
15	INTEGRADO	TI	4	INDIFERENTE	INDIFERENTE	GOSTO	INDIFERENTE	SIM	EXIGE	SIM	C# JAVA PPASCAL
16	INTEGRADO	TI	2	GOSTO MUITO	TENHO MUITA	GOSTO MUITO	TENHO	SIM	EXIGE	SIM	JAVA PZIM MY SQL

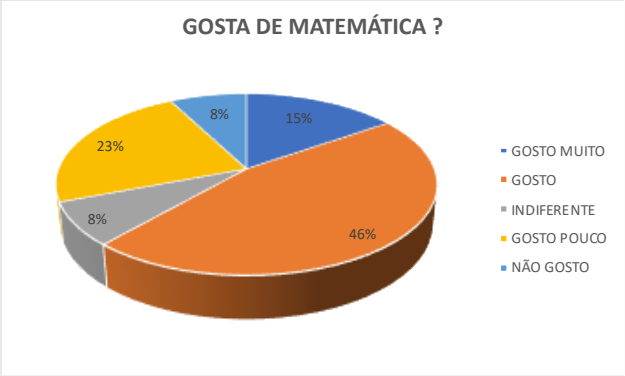
Fonte: Dados da pesquisa.

Estes dados sobre o curso e série dos alunos, se já teriam visto o conteúdo de geometria plana, foram respondidos em um questionário sem identificação dos nomes antes do início da oficina. Neste mesmo instrumento, pudemos averiguar questões como gosto e facilidade em matemática e raciocínio lógico e experiência em programação (Tabela 1 e Tabela 2).

De acordo com as respostas dos estudantes, com relação ao atributo de gostar de matemática, quase a metade da amostra (46%) declarou gostar e 15% declarou gostar muito (Figura 1). Já com relação à facilidade com matemática (Figura 2), somente um aluno (8%) declarou ter muita facilidade e cinco alunos (38%) declararam ter facilidade, representando uma

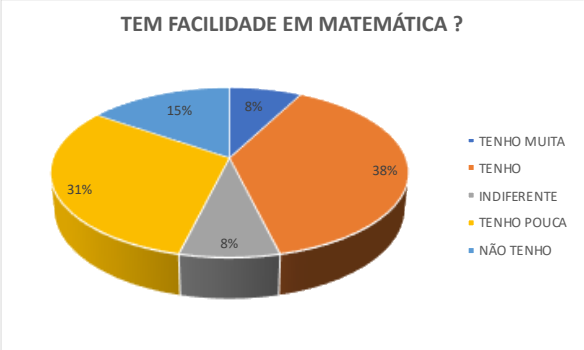
soma de 46%. O restante, 31% declararam que tem pouca facilidade, 15% declararam não ter facilidade e 8% declararam serem indiferentes. Portanto, percebemos que o gosto por matemática não implica em facilidade.

Figura 1- Resposta dos alunos em relação ao gosto por matemática.



Fonte: Dados da pesquisa.

Figura 2 - Resposta dos alunos em relação à facilidade em matemática.



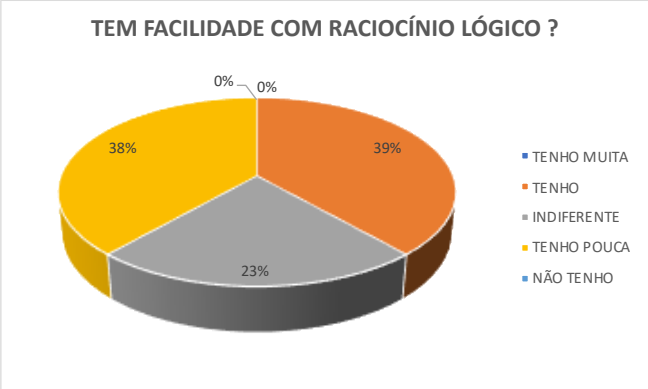
Fonte: Dados da pesquisa.

Com relação ao gosto por raciocínio lógico, a grande maioria tem identificação pois 69% declararam gostar e 23% declararam gostar muito. Nenhum aluno se declarou indiferente ou que não gosta (Figura 3). Já com relação à facilidade em raciocínio lógico, as respostas aconteceram de forma análoga à matemática pois nenhum aluno afirmou ter muita facilidade, 23% se mostraram indiferentes e 38% afirmaram ter pouca facilidade. Apenas 39% afirmaram ter facilidade e nenhum aluno se classificou como tendo pouca facilidade (Figura 4).

Figura 3 - Resposta dos alunos com relação ao gosto por raciocínio lógico.

Fonte: Dados da pesquisa.

Figura 4 - Resposta dos alunos com relação à facilidade em raciocínio lógico.



Fonte: Dados da pesquisa.

3. Desenvolvimento

3.1 A Oficina

A oficina teve duração de 3 horas e foi ministrada por dois docentes da área de TI. O objetivo foi fazer observações sobre a resolução de problemas abstratos pelos alunos. Durante todo o desenvolvimento e tentativas de resolução dos problemas, os estudantes tiveram o apoio dos docentes em sala. As primeiras dicas eram oferecidas ao apresentar o problema no quadro e as dicas necessárias para chegar a uma solução eram dadas em conversas individuais com os alunos. O apoio se torna importante pois os alunos não alcançam o nível máximo em suas habilidades todas as vezes em que irão desenvolver uma tarefa. O desempenho nas habilidades depende do suporte, do contexto e do estado emocional (Fischer & Bidell, 1998). Existe um intervalo entre os níveis funcional e ótimo, ou seja, o delta de desempenho na execução de uma tarefa com e sem apoio contextual, chamado alcance (Schwartz & Fischer, 2004).

3.2 Os problemas propostos na oficina

Os problemas propostos na oficina consistiram em redesenhar os modelos dados, pelos docentes, de figuras que representavam ou eram compostas por figuras geométricas através algoritmos formados por uma sequência de comandos do programa de computador LOGO. A linguagem de programação LOGO, desenvolvida na década de 60, é uma linguagem interpretada e apropriada para crianças e seu funcionamento consistia em desenhar na tela do computador desafios impostos pelo sistema (Ramos & Moraes, 2020). Para que os algoritmos pudessem ser criados pelos alunos, a etapa anterior ao desenho (conjunto de comandos que forma o desenho - algoritmo) seria descobrir todas as medidas e ângulos da figura a ser desenhada através de deduções baseadas nos postulados matemáticos da geometria plana.

> “A linguagem LOGO foi criada no final da década de 60. Papert, que foi um dos criadores da linguagem LOGO, trabalhou com Piaget e foi influenciado pelas teorias do construtivismo. A proposta da linguagem LOGO era colocar a criança para comandar um robô ou uma representação de robô na tela do computador. [...] A partir de comandos como "parafrente 100" (pontos), e "giredireita 45" (graus), movimenta-se o robô pelo espaço. Quando o robô está representado na tela do computador, a tartaruga deixa um rastro por onde passa, criando assim um desenho. O termo LOGO é derivado do grego "logos" que significa "palavra", uma referência aos comandos da linguagem.” (Pimentel, 2019)

A linguagem LOGO possui uma lista de comandos que permite que o usuário configure para onde a tartaruga vai andar (ângulo), a direção (para frente e para trás), a distância e se vai deixar rastro ou não, através dos comandos *uselápis* e

useborracha (Quadro *1*). Os comandos também têm suas versões abreviadas, ao invés de *parafrente* podemos usar *pf*, *pt* ao invés de *paratrás*, *pd* ao invés de *paradireita* e *pe* ao invés de *paraesquerda*.

Quadro 1- Comandos do LOGO na versão SuperLogo e XLogo.

SuperLogo	xLogo	Ação
parafrente	parafrente	move a tartaruga *n* passos para a frente
paratrás	paratrás	move a tartaruga *n* passos para trás
paradireita	paradireita	gira a tartaruga *n* graus para a direita
paraesquerda	paraesquerda	gira a tartaruga *n* graus para a esquerda
tartaruga	limpedesenho	limpa a janela gráfica e posiciona a tartaruga no centro da tela e voltada para cima
usenada	usenada	a tartaruga passa a não desenhar o caminho que vier a percorrer na tela (não deixa o rastro)
uselápis	uselápis	a tartaruga passa a desenhar o caminho que vier a percorrer na tela
useborracha	useborracha	a tartaruga passa a apagar o que estiver no caminho que vier a percorrer na tela
desapareçatat	escondetat	a representação da tartaruga não será apresentada na tela (ficará invisível)
apareçatat	mostretat	a representação da tartaruga será apresentada na tela (ficará visível)
mudecl	mudecordolápis	muda a cor usada no traçado do caminho que vier a percorrer na tela. A cor é passada como parâmetro (um número que representa a cor numa tabela de cores)
mudecp	mudecordolápis	muda a cor que será usada no comando "pinte"
pinte	pinte	pinta a região sob a tartaruga
repita	repita	repete *n* vezes uma lista de comandos (representada entre colchetes)
aprenda	aprenda	cria um procedimento (um novo comando que a tartaruga terá "aprendido" a realizar)
atribua	atribua	atribui a uma variável um dado
sorteienúmero	sorteie	retorna um número aleatório entre 0 e (*n*-1)

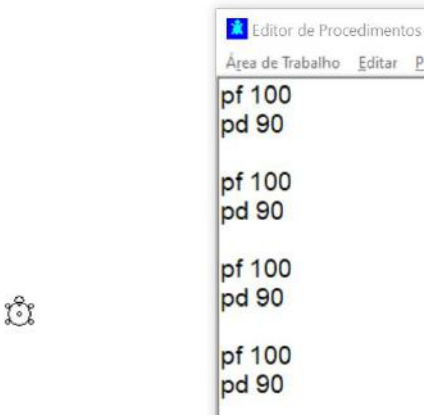
Fonte: Pimentel (2019).

A partir da explicação do funcionamento da ferramenta LOGO, demonstramos o primeiro exemplo que foi desenhar um quadrado com lado medindo 100 unidades e em seguida propusemos que os alunos desenhassem as próximas figuras, sempre os incitando a utilizar as regras por trás dos desenhos da figura, ou seja, os postulados da geometria plana. Na

Figura **5** temos a tartaruga na posição inicial do logo e os comandos que serão utilizados para desenhar um quadrado. Notemos que para desenhar um quadrado é preciso que a tartaruga execute quatro vezes os passos de andar 100 para a frente e depois virar 90 graus para a direita.

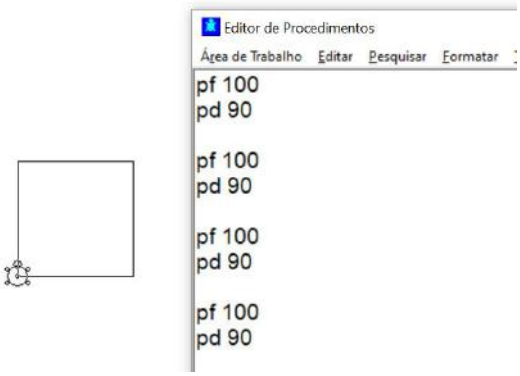
Na Figura 6 temos o quadrado já desenhado, ou seja, os comandos já foram executados. Porém, notamos que se trata de uma repetição dos mesmos comandos para desenhar um quadrado. Com a utilização de estruturas de repetição, que é uma estrutura utilizada em lógica de programação, é possível indicar que a tartaruga execute aquele mesmo conjunto de comandos 4 vezes seguidas. Na Figura 7, temos a tartaruga criada em um código bastante enxuto, utilizando estruturas de repetição.

Figura 5 – Tartaruga na posição inicial e lista de comandos executados para desenhar um quadrado.



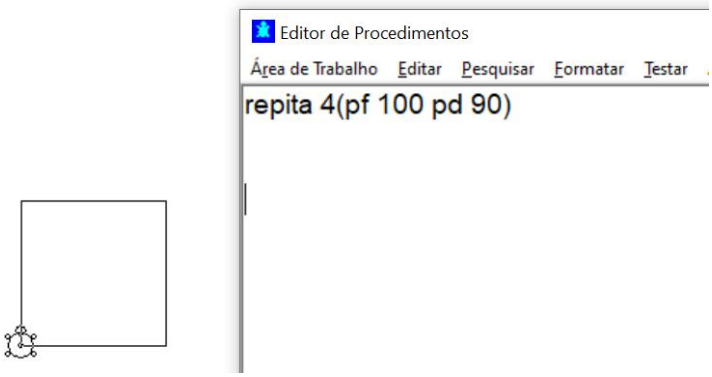
Fonte: dados da pesquisa.

Figura 6- Tartaruga na posição final após execução da lista de comandos executados para desenhar um quadrado.



Fonte: Dados da pesquisa.

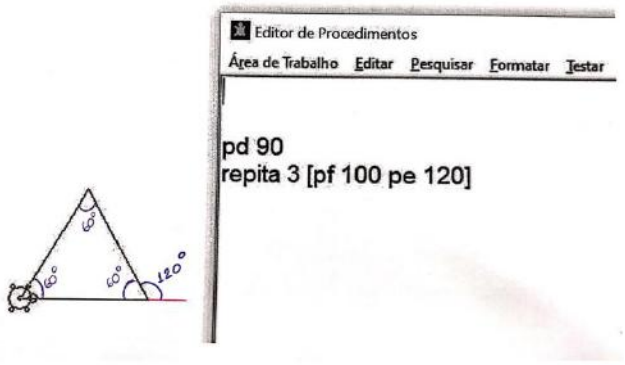
Figura 7- Comandos otimizados utilizando a estrutura de repetição para desenhar o mesmo quadrado.



Fonte: Dados da pesquisa.

O segundo exercício foi o desenho de um triângulo equilátero, sendo que eles já tinham a experiência de desenhar um quadrado. Primeiramente, é necessário girar a tartaruga 90 graus para direita e andar para frente 100 unidades, depois girar 120 graus à esquerda para deixar a tartaruga em posição de riscar o próximo lado. Foi necessário girar 120 graus pois esta é a medida do ângulo complementar dos 60 graus internos de cada ângulo de um triângulo equilátero (3 lados e 3 ângulos iguais com soma interna de 180 graus) (Figura 8). Notemos a posição final da tartaruga ao terminar de desenhar o triângulo.

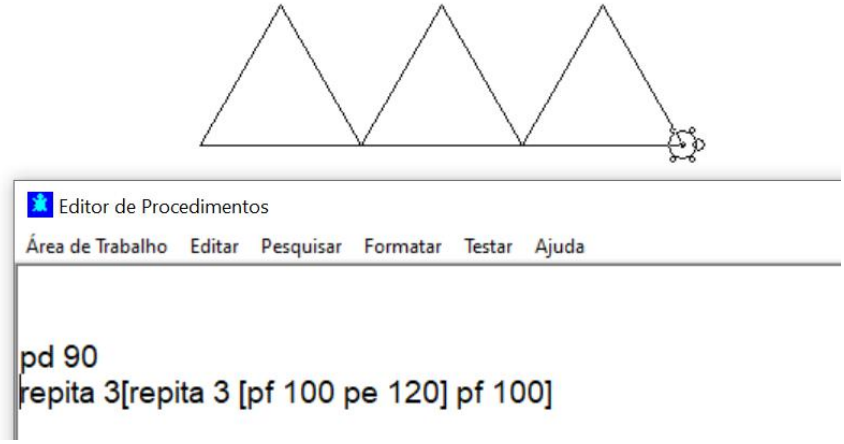
Figura 8 – Comandos utilizados para desenhar um triângulo utilizando estruturas de repetição.



Fonte: Dados da pesquisa.

Já o terceiro exercício proposto foi de desenhar três triângulos seguidamente. A única diferença com relação a desenhar um triângulo é que devemos andar para frente a mesma medida do lado do triângulo, pois a tartaruga finaliza o movimento do desenho no lado esquerdo do triângulo e virada para a direita (Figura 9), como já falado no parágrafo acima.

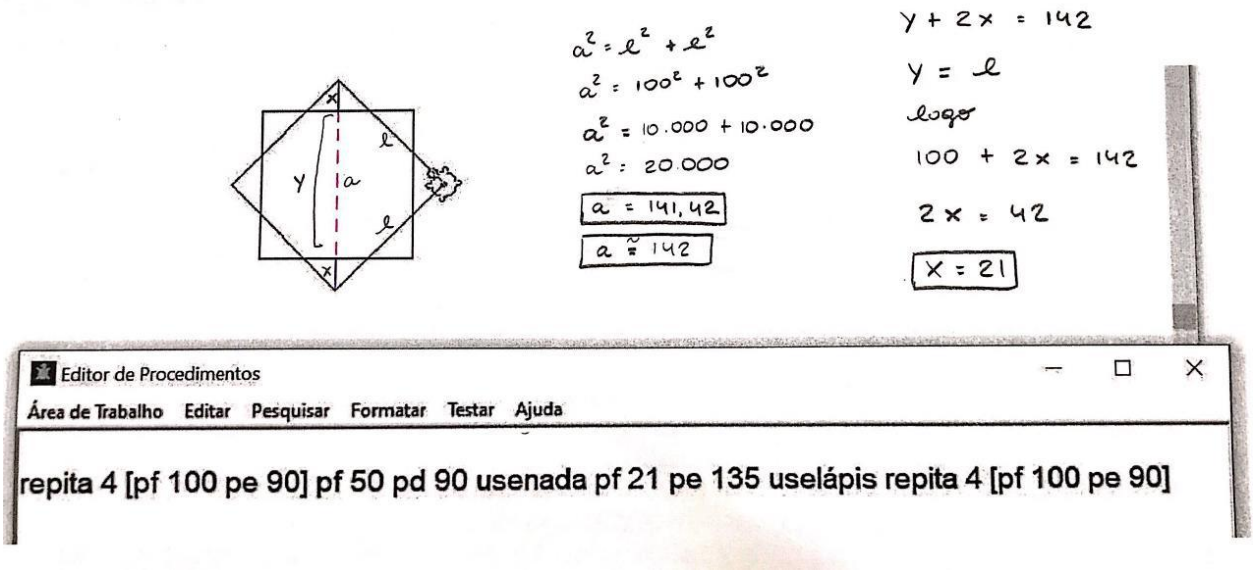
Figura 9 - Comandos para desenhar três triângulos com estruturas de repetição.



Fonte: Dados da pesquisa.

O quarto exercício proposto foi desenhar dois quadrados sobrepostos, só que era necessário andar a posição de tamanho X, que equivale a metade da diferença entre hipotenusa e o lado do quadrado. Após as contas demonstradas na Figura 10, vimos que foi necessário percorrer este caminho sem riscar (utilizando o comando *usenada*) e depois voltar a riscar para desenhar o quadrado (utilizando o comando *uselápis*).

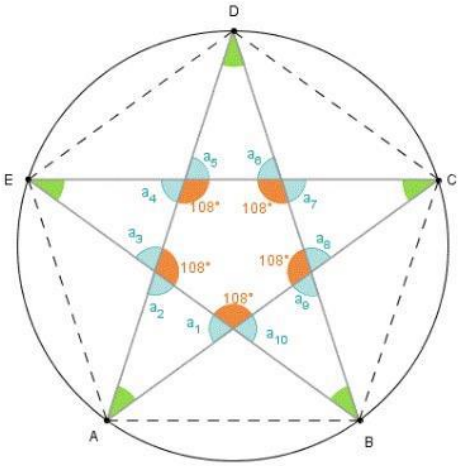
Figura 10 – Comando para desenhar dois quadrados sobrepostos utilizando estruturas de repetição e estratégias de *uselápis* e *usinada*.



Fonte: Dados da pesquisa.

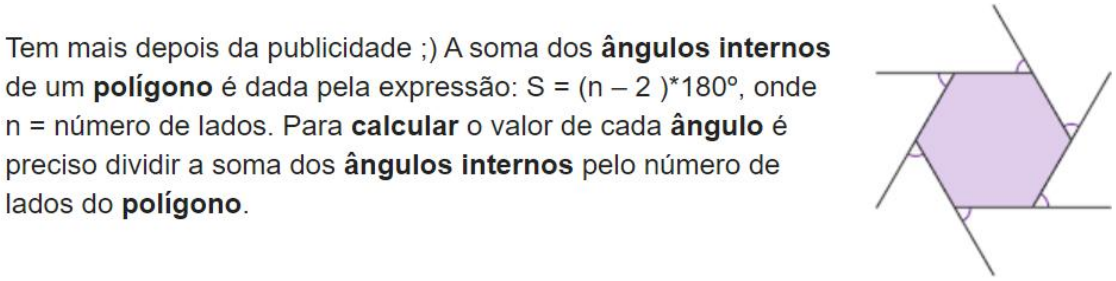
O quinto exercício proposto foi desenhar um pentagrama. Para isso, os alunos tinham que perceber que os lados inferiores de cada um dos cinco triângulos, juntos, formam um pentágono regular (Figura 11). Também teriam que descobrir a medida do ângulo interno deste pentágono, que é dada pela fórmula que obtém a soma dos ângulos internos do pentágono e depois divide-se o resultado pelo número de lados do mesmo (Figura 12). Com isso os estudantes teriam a informação que o ângulo interno do pentágono mediria 108 graus. Assim, deduz-se que os dois ângulos complementares aos ângulos internos no polígono medem 72 graus, restando então 36 graus para complementar o terceiro ângulo do triângulo, já que a soma dos ângulos internos de um triângulo sempre será 180 graus.

Figura 11 - Problema da estrela pentagonal.



Fonte: Clubes de Matemática da OBMEP (http://clubes.obmep.org.br/blog/probleminha-estrela-pentagonal/).

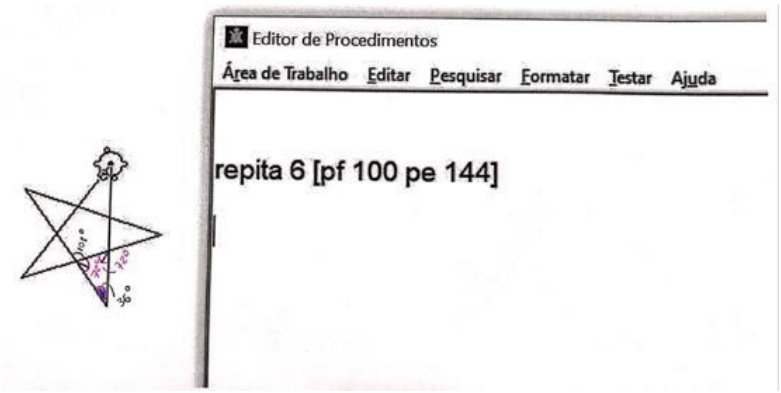
Figura 12- Fórmula para descoberta da soma dos ângulos internos de um polígono regular.



Fonte: mundoeducacao.bol.uol.com.br.

Assim, é preciso a última "sacada" para desenhar o pentágono. Não precisamos desenhar triângulo por triângulo. Conhecendo o ângulo interno dos triângulos formados, precisamos girar o valor complementar destes ângulos para a esquerda (144 graus) e andar pra frente 100 unidades por 6 vezes. Como podemos observar, a maior dificuldade está na interpretação do problema matemático, que é complexo e abstrato (Figura 13).

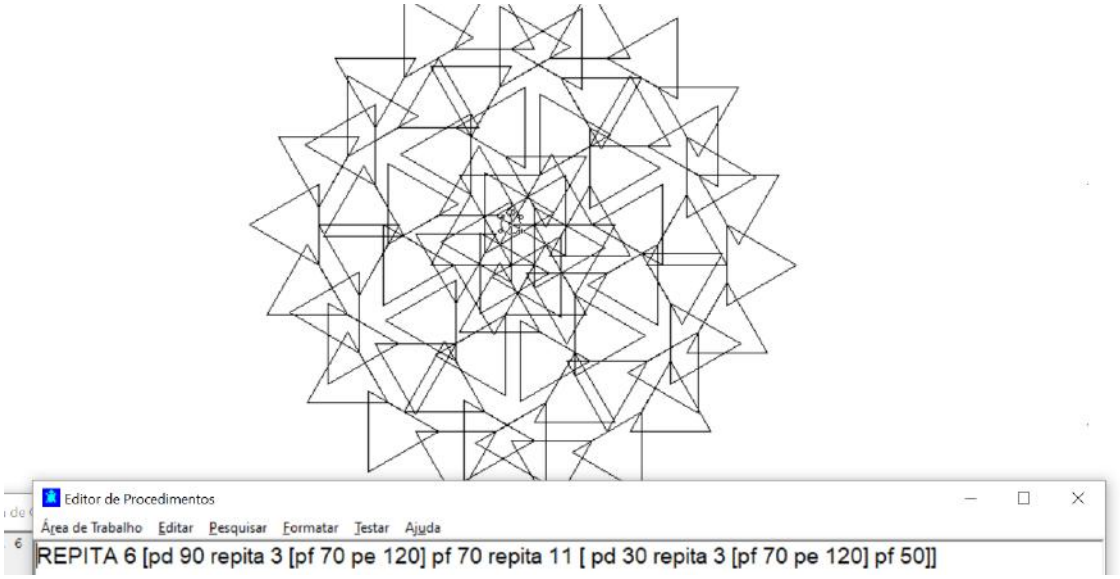
Figura 13- comandos para desenhar o pentágono.



Fonte: Dados da pesquisa.

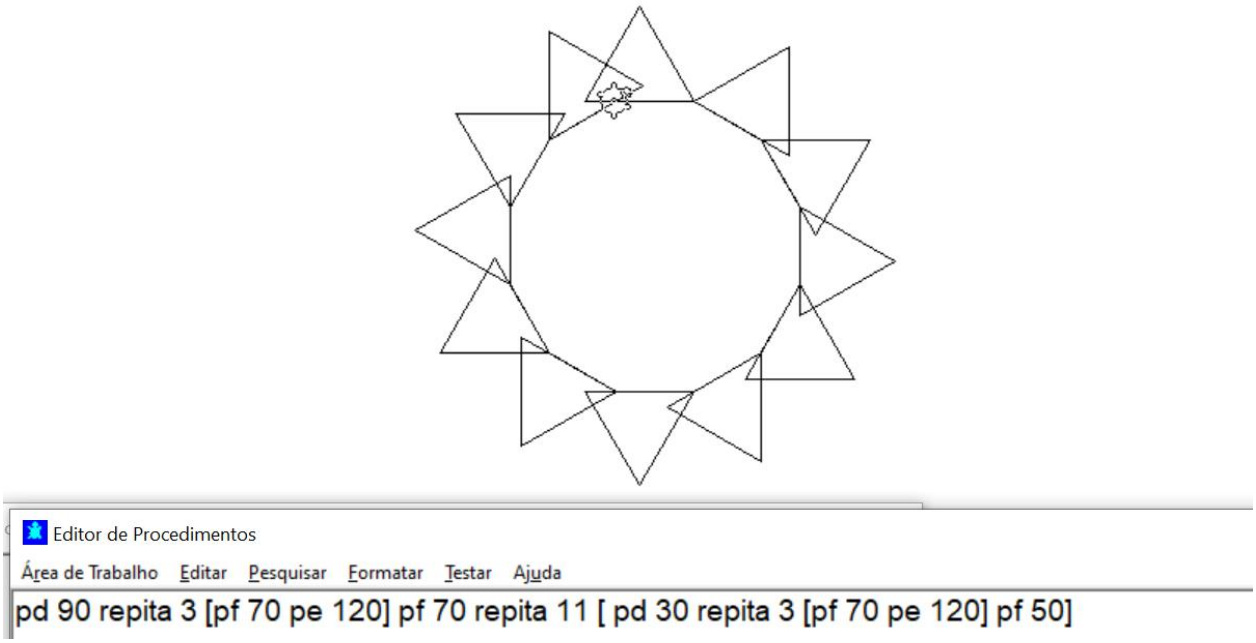
Enfim, o sexto e último exercício foi desenhar uma mandala através de 6 repetições de uma sequência de 12 triângulos. A mandala representada na Figura 14 é formada por 6 conjuntos de 12 triângulos interligados através de ângulos de 30 graus (Figura 15).

Figura 14- comandos para desenhar a mandala completa.



Fonte: Dados da pesquisa.

Figura 15 - Comandos para desenhar os conjuntos de 12 triângulos formadores da mandala completa.



Fonte: Dados da pesquisa.

3.3 Desenvolvimento das soluções pelos estudantes

Em geral, a maior dificuldade dos estudantes foi em descobrir como construir as figuras, ou seja, a dificuldade não estava em usar os comandos da ferramenta LOGO isoladamente, mas em saber para onde movimentar a tartaruga quanto aos ângulos que seriam utilizados e a distância que ela iria percorrer. Esta dificuldade corrobora com Falkembach et al (2003) ao

citar que uma das maiores dificuldades dos estudantes é a interpretação do problema a ser resolvido. Os alunos tinham dificuldade em decompor as figuras e de identificar várias figuras compondo uma figura mais complexa. Logo, daí a dificuldade em descobrir os ângulos e distâncias, pois, ao olhar a figura como um todo sem utilizar a abstração para dividir em partes e retirar destas partes as informações que lhes interessam, não é possível caminhar no processo de construção da figura completa. Esta última dificuldade corrobora com Rodrigues (2002), quando o autor cita as dificuldades dos estudantes em assimilar abstrações envolvidas no processo de ensino-aprendizagem de fundamentos de programação. A dificuldade em assimilar as abstrações não está presente somente na área de programação, mas também na área matemática.

Com relação à assimilação dos comandos de forma isolada, o comando de repetição isoladamente foi facilmente absorvido pois todos já conheciam, ou seja, foi apenas uma transposição de uma linguagem para outra. Com relação ao uso dos comandos *uselápis* e *usenada*, houve algumas dificuldades com relação à posição do comando no código, pois erros de posicionamento levam a equívocos no desenho das figuras ou até mesmo falta de desenho, pois a tartaruga se movimenta sem deixar rastros e muitas vezes deixa a impressão de que não está acontecendo nada.

Também houve dificuldades dos alunos em saber que fórmulas poderiam aplicar para descobrir determinadas medidas. Isto pode ter acontecido pois quando o problema é dado, as perguntas a serem respondidas são várias, mas não são dadas. É papel do aluno fazer as perguntas. Por exemplo, se eu quero saber a medida de determinado lado, eu tenho que saber determinado ângulo. E sucessivamente, para saber a medida do ângulo ele tem que usar os postulados de complementaridade, e daí por diante. Uma trilha tem que ser traçada pelo aluno, um caminho a percorrer a partir de algumas informações que são dadas de forma implícita na figura. Ele tem que descobrir por onde partir e que caminho seguir para deduzir as medidas.

Houve também muitas dificuldades em relação à sequência dos comandos que formariam o algoritmo, ou seja, o caminho completo para desenhar cada figura do início ao final. Este fato corrobora com Campos (2009), quando cita a dificuldade dos alunos na aplicação das estruturas de controle dos algoritmos.

Enfim, concluímos que desenhar figuras geométricas através dos algoritmos é mais complexo do que resolver um problema matemático puro. Muitas vezes os alunos conseguiam deduzir as medidas da figura, mas tinham dificuldade no algoritmo para desenhá-las através da tartaruga. Este fato corrobora com Regebe (2007) ao constatar que um item de conhecimento matemático puro se tornava muito mais difícil quando a resolução era exigida através de um algoritmo, tornando-se assim um item híbrido.

Enfim, nosso resultado também corroborou com Fensham (1994), pois ficou claro que não houve transferência direta dos conhecimentos de geometria plana para o desenvolvimento de figuras geométricas através de algoritmos, mas é preciso a construção de um novo conhecimento a partir dos dois domínios.

4. Considerações Finais

Neste artigo apresentamos a aplicação de uma oficina com o objetivo de propor problemas abstratos em lógica de programação envolvendo conteúdos e problemas abstratos de geometria plana, assim como os resultados obtidos nesta aplicação. A necessidade surgiu diante da dificuldade de os alunos de lógica de programação trabalharem com conceitos abstratos e diante da necessidade de estratégias que desenvolvam a capacidade de abstração e resolução de problemas matemáticos dos alunos.

Observamos que os alunos apresentam dificuldades na interpretação do problema proposto, na decomposição do problema em partes, na visualização de várias figuras dentro de uma figura mais complexa e na capacidade de abstração para identificar na figura as medidas que lhes interessam na resolução do problema dado.

Percebemos também que ao passar da fase de reconhecimento da figura e suas medidas, os alunos também apresentam dificuldades para implementar os algoritmos que constroem a figura do início ao fim com a sequência de comandos que formam o algoritmo.

Concluímos que desenhar figuras matemáticas através dos algoritmos é mais complexo do que resolver um problema matemático puro. Muitas vezes os alunos conseguiam deduzir as medidas da figura, mas tinham dificuldades na construção dos algoritmos para desenhá-las através de comandos que deixavam rastro com o movimento da tartaruga. Logo, itens que envolvem matemática e programação tornam-se mais complexos que itens que envolvem ou um conteúdo ou outro isoladamente, resultado que corrobora com Regebe (2007). Ou seja, o conhecimento não é transferido facilmente de uma área para outra, mas é necessária a construção de um novo conhecimento, corroborando com Fensham (1994).

Também observamos que a maior dificuldade dos alunos é criar estratégias matemáticas em situações que requerem pensamento abstrato em maior nível de complexidade. O fato de o problema não conter explicitamente as perguntas que devem ser feitas pelo aluno para chegar ao resultado dificulta o entendimento e a resolução do problema. Os alunos requerem um alto nível de suporte no desenvolvimento desta trilha, e não se tornam independentes nesta parte da atividade. Notamos também uma dificuldade em divisão de problemas, ou seja, para que os alunos pudessem criar a trilha da solução, seria necessário dividir o problema todo em partes menores e ordená-las.

Enfim, todas essas conclusões corroboram com os resultados da pesquisa de Regebe (2017), que propõe um maior investimento em estratégias que promovam a abstração, raciocínio lógico/matemático, interpretação e divisão de problemas matemáticos no currículo dos alunos para suprir esta lacuna que encontramos hoje.

Como trabalho futuro, propomos um estudo longitudinal a ser feito juntamente com a disciplina de matemática, ao longo do conteúdo de geometria plana, para investigar os ganhos no desenvolvimento de problemas matemáticos de geometria plana quando os alunos são motivados a resolver problemas similares envolvendo os mesmos conteúdos através da ferramenta LOGO.

Referências

Amantes, A. (2009). *Contextualização no ensino de Física: efeitos sobre a evolução do entendimento dos estudantes*. Tese de doutorado, Instituto de Física, Universidade Federal de Minas Gerais.

Bigolin, N. M., Silveira, S. R., Bertolini, C., de Almeida, I. C., Geller, M., Parreira, F. J., & Macedo, R. T. (2020). Metodologias Ativas de Aprendizagem: um relato de experiência nas disciplinas de programação e estrutura de dados. *Research, Society and Development*, 9(1), e74911648-e74911648.

Campos, R. L. B. L. (2009, September). Lógica de programação: Há como melhorar o aprendizado fugindo dos padrões estabelecidos nos livros didáticos e adotados pela maioria dos docentes. In *XVII Congresso Iberoamericano de Educación Superior em Computacion* (CLEI-2009-CIESC) (pp. 22-25).

Falkembach, G. A. M., Amoretti, M. S. M., Tarouco, L. R., & Viero, F. (2003). Aprendizagem de algoritmos: uso da estratégia ascendente de resolução de problemas. 8º *Taller Internacional de Software Educativo*. Santiago, Chile.

Fischer, K. W., & Bidell, T. R. (1998). Dynamic development of psychological structures in action and thought.

Gomes, A., Carmo, L., Bigotte, E., & Mendes, A. (2006, September). Mathematics and programming problem solving. In *3rd e-learning conference–computer science education* (pp. 1-5).

Gomes, A., Henriques, J., & Mendes, A. (2008). Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. *Educação, Formação & Tecnologias*-ISSN 1646-933X, 1(1), 93-103.

Gomes, A., & Mendes, A. (2014, October). A teacher's view about introductory programming teaching and learning: Difficulties, strategies and motivations. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings* (pp. 1-8). IEEE.

Grollmus, N. S., & Tarrés, J. P. (2015). Relatos metodológicos: difractando experiencias narrativas de investigación. In *Forum Qualitative Sozialforschung/Forum: Qualitative Social Research* (16(2), 24).

Martins, S. W., & Correia, L. D. A. (2003). O Logo como ferramenta auxiliar no desenvolvimento do raciocínio lógico–um estudo de caso. In *Internacional Conference on Engineering and Computer Education–ICECE.*

Pereira Júnior, J. C. R., & Rapkiewicz, C. E. O Processo de Ensino-Aprendizagem de Fundamentos de Programação: Uma visão crítica da pesquisa no Brasil. In *I Workshop de Educação em Computação*.

Pimentel (2019). Linguagem LOGO. https://sites.google.com/site/infoeducunirio/perspectiva-construtivista/linguagem-logo

Probleminha: Estrela Pentagonal. Clubes de Matemática da OBMEP: Disseminando o estudo da matemática. http://clubes.obmep.org.br/blog/probleminha-estrela-pentagonal/

Ramos, B. A., & Moraes, E. C. (2020). Robótica Educacional como metodologia motivadora no ensino de lógica de programação na Educação Profissional e Tecnológica. *Research, Society and Development*, 9(12), e18591210938-e18591210938.

Regebe, F. (2017). *A evolução das habilidades cognitivas de raciocínio lógico em tecnologia da informação*. Tese de doutorado, Instituto de Física, Universidade Federal da Bahia

Regebe, F., Amantes, A. (2019) Performance analysis of students regarding the complexity and content dimensions of items in learning of programming logic. In *European Science Education Research Association – ESERA*

Rodrigues, M. C. (2002) Como Ensinar Programação?

Silva, M. N. P. Soma dos ângulos internos de um polígono regular. https://mundoeducacao.bol.uol.com.br/matematica/soma-dos-angulos-internos-um-poligono-regular.htm#:~:targetText=Tem%20mais%20depois%20da%20publicidade%20%3B)&targetText=A%20soma%20dos%20%C3%A2ngulos%20internos,n%C3%BAmero%20de%20lados%20do%20pol%C3%ADgono..

Schwartz, M. S., & Fischer, K. W. (2004). Building general knowledge and skill: Cognition and microdevelopment in science learning. *Cognitive developmental change: Theories, models, and measurement*, 157-185.

Shitsuka, D. M., Pereira, A. S., Shitsuka, R., & Boghi, C. (2019). Aprendizagem ativa de programação em turmas de engenharia: uma pesquisa-ação. *Research, Society and Development*, 8(3), 01-19.