Revista Brasileira de Informática na Educação – RBIE
Brazilian Journal of Computers in Education
(ISSN online: 2317-6121; print: 1414-5685)
http://br-ie.org/pub/index.php/rbie

Submission: 14/Mar/2018; 1st round notif.: 07/Jan/2019; New version: 19/Feb/2019; 2nd round notif.: 26Jun/2019
Camera ready: 30/Jun/2019; Edition review: 1/July/2019; Available online: 12/July/2019; Published: 31/July/2019

DuinoBlocks4Kids: Utilizando Tecnologia Livre e Materiais de Baixo Custo para o Exercício do Pensamento Computacional no Ensino Fundamental I por meio do Aprendizado de Programação Aliado à Robótica Educacional

Title: DuinoBlocks4Kids: Using Free Technology and Low-Cost Materials for the Exercise of Computational Thinking in Primary School via Programming and Educational Robotics

Rubens Lacerda Queiroz
Programa de Pós-Graduação em Informática (PPGI/UFRJ)
rubensqueiroz@outlook.com

Fábio Ferrentini Sampaio
Programa de Pós-Graduação em Informática (PPGI/UFRJ)
ffs@nce.ufrj.br

Mônica Pereira dos Santos
Programa de Pós-Graduação em Educação (PPGE/UFRJ)
monicapes@gmail.com

Resumo

Este artigo apresenta um estudo de caso sobre o desenvolvimento do Pensamento Computacional em crianças do Ensino Fundamental I, através do aprendizado de programação por meio da Robótica Educacional, fazendo-se uso exclusivamente de tecnologia livre e materiais recicláveis e de baixo custo. Busca-se também levantar hipóteses acerca da existência de uma relação direta entre certas características cognitivas de crianças com idade entre 8 e 10 anos (tais como a habilidade de sequenciar eventos ou ideias, a habilidade de realizar operações mentais a partir de experiências concretas, dentre outras) e a habilidade para realizar determinadas atividades relacionadas ao aprendizado de programação de computadores. Os resultados observados indicam (a partir do uso de um kit didático desenvolvido para a realização deste estudo) a possibilidade de desenvolvimento das seguintes habilidades do Pensamento Computacional: capacidade de abstração, compreensão de fluxos de controle, depuração e detecção sistemática de erros, pensamento iterativo, uso da lógica condicional e decomposição de problemas. No tocante às investigações relacionadas à maturidade cognitiva, foram encontrados indícios da existência de uma relação direta entre as características cognitivas analisadas e a realização de determinadas tarefas ligadas à programação de computadores, como o desenvolvimento de programas puramente sequenciais e a compreensão da ideia de processamento.

Palavras-Chave: *Pensamento Computacional; Ensino de programação; Robótica Educacional; Maturidade cognitiva*

Abstract

This paper presents a case study about the development of Computational Thinking in primary school children (3st to 4th grade) via the teaching of programming abilities with the use of educational robotics, free technology and recyclable, low cost materials. We aimed at raising some hypotheses on whether there is a straight relationship between some cognitive aspects of children aged 8-10 (such as the ability to put events and ideas in sequence, the ability to execute mental operations on the basis of concrete experience, among others) and the ability to execute activities that may be linked to the learning of computer programming. The observed results indicated (from the use of a didactic kit developed for the accomplishment of this study) the possibility to develop the following computational thinking skills: abstract thinking ability, understanding of flows of control, Debugging and systematic error detection, iterative thinking, use of conditional logic and problem decomposition. Regarding the investigations related to cognitive maturity, we found evidence of a correlation between the cognitive characteristics analyzed and the performance of certain tasks related to computer programming, such as the development of purely sequential programs and understanding of processing idea.

Cite as: Queiroz, R.L., Sampaio, F.F. & Santos, M.P. (2019). DuinoBlocks4Kids: Using Free Technology and Low-Cost Materials for the Exercise of Computational Thinking in Primary School via Programming and Educational Robotics (DuinoBlocks4Kids: Utilizando Tecnologia Livre e Materiais de Baixo Custo para o Exercício do Pensamento Computacional no Ensino Fundamental I por meio do Aprendizado de Programação Aliado à Robótica Educacional). Brazilian Journal of Computers in Education (Revista Brasileira de Informática na Educação - RBIE), 27(2), 167-196. DOI: 10.5753/RBIE.2019.27.02.167

Keywords: *Computational Thinking; Teaching computer programming; Educational Robotics; Cognitive maturity*

1 Introdução

O aprendizado de programação de computadores, já nos primeiros anos do Ensino Fundamental, tem sido visto como uma importante ferramenta para o desenvolvimento e exercício de algumas habilidades do Pensamento Computacional (Grover & Pea, 2013).

Segundo Wing (2006), pesquisadora que cunhou o termo *Computacional Thinking*, o Pensamento Computacional pode ser visto como uma forma de pensamento característica dos cientistas da computação, mas, universalmente aplicável, envolvendo um conjunto de atitudes e competências, tais como o uso da depuração, recursividade, generalização e decomposição na solução de problemas tanto técnico-científicos quanto da vida cotidiana. Trata-se de uma habilidade fundamental para qualquer pessoa e que deveria ser incorporada à capacidade analítica de toda criança.

No Brasil, pode-se verificar uma preocupação do Ministério da Educação neste sentido pela recente inclusão, na Base Nacional Comum Curricular (BNCC), do Pensamento Computacional como uma das competências a serem desenvolvidas pelos alunos já na Educação Básica, indicando-se o trabalho com algoritmos, tarefa inerente ao aprendizado de programação de computadores, como uma maneira de exercitá-lo (Ministério da Educação, 2017)

Outro aspecto que tem recebido crescente atenção em relação a esse tema, e que acaba por apontar a programação de computadores como sendo uma nova habilidade básica, tal qual a escrita, a leitura e a aritmética, é a importância do seu desenvolvimento como meio de possibilitar a todo cidadão ser produtor e não apenas consumidor de tecnologia digital, oportunizando assim, entre outras coisas, a mobilidade social em um mundo fortemente orientado por esse tipo de tecnologia (Whitehouse, 2016).

Diante deste cenário, diferentes autores, como Resnick et al (2009), Lye e Koh (2014) e Cooper, Grover, Guzdial e Simon (2014) vêm defendendo a necessidade de se incorporar o ensino de programação já nos primeiros anos do Ensino Básico (K12 nos EUA), e, para que iniciativas nesse sentido sejam adotadas, torna-se necessário o desenvolvimento de recursos idealizados especificamente para o ensino de programação de computadores a crianças. Como resposta a essa demanda, uma série de estratégias vêm sendo pensadas e implementadas, entre elas, a associação da Robótica Educacional com Ambientes de Programação Visual em Blocos.

Para que os benefícios do aprendizado de programação com robótica possam ser levados a um número maior de crianças, especialmente aquelas pertencentes a comunidades de baixa renda e matriculadas em escolas públicas, torna-se fundamental o desenvolvimento de estratégias de aprendizado economicamente acessíveis, no que o uso de Tecnologias Livres e materiais recicláveis e de baixo custo tornam-se grandes aliados.

Dentre as tecnologias atualmente existentes para o desenvolvimento de atividades na área de Robótica Educacional, a plataforma de prototipagem eletrônica de código aberto Arduino[1] vem se destacando devido ao seu baixo custo, qualidade, flexibilidade e facilidade de uso. No entanto, uma pesquisa acerca de ambientes de programação visual voltados ao ensino de programação com robótica, realizada pela equipe do laboratório de Inovações em Robótica Educacional da UFRJ (LIvRE[2]), apontou para a inexistência de ambientes de programação

[1] https://www.arduino.cc/
[2] http://ginape.nce.ufrj.br/LIVRE

voltados à programação de placas de prototipagem eletrônica Arduino, pensados e desenvolvidos especificamente para o ensino de programação a crianças.

Como resposta a essa demanda, surge a proposta do desenvolvimento do DuinoBlocks4Kids (DB4K), um kit didático baseado em Tecnologia Livre, voltado ao ensino de programação para crianças do Ensino Fundamental I, composto por um Ambiente de Programação Visual baseado em blocos, um conjunto de materiais de Robótica Educacional e uma série de atividades (Queiroz, Sampaio & Santos, 2017).

O objetivo geral do estudo aqui apresentado é, a partir do uso do kit DB4K, desenvolvido pelos autores desta pesquisa, investigar a possibilidade de desenvolvimento e exercício de habilidades do Pensamento Computacional em crianças do Ensino Fundamental I a partir do aprendizado de conceitos básicos de programação por meio de recursos baseados exclusivamente em Tecnologia Livre e materiais recicláveis e de baixo custo, associados a estratégias pedagógicas alicerçadas na Robótica Educacional, pensadas e desenvolvidas especificamente para esse público.

Em paralelo, busca-se levantar hipóteses acerca da existência de uma relação direta entre certos aspectos da maturidade cognitiva de crianças entre 8 e 10 anos, tais como a habilidade de sequenciar eventos ou ideias e a habilidade de realizar operações mentais a partir de experiências concretas, dentre outras, e as habilidades necessárias para a realização de determinadas tarefas relacionadas à programação de computadores.

A realização deste tipo de investigação, especialmente no que se refere à capacidade de abstração, torna-se bastante importante no sentido de auxiliar os educadores quanto à escolha das atividades de programação a serem realizadas com as crianças, de modo que essas atividades sejam oferecidas no momento apropriado e de maneira adequada.

2 Fundamentação teórica

2.1 Construcionismo e Robótica Educacional

A Teoria Construcionista de Seymour Papert une a Teoria Construtivista de Jean Piaget ao uso do computador na educação. Papert (1993) adicionou à teoria de Piaget (1974) - onde afirma que a manipulação de objetos é a chave para as crianças construírem seu próprio conhecimento - a ideia de que essa construção se dá de forma mais efetiva quando o aprendiz se engaja de maneira consciente na construção de algo tangível (Papert, 1993).

Papert percebeu no computador uma ferramenta capaz de ampliar as possibilidades de criação e, consequentemente, de aprendizado das crianças, ao permitir a elas o desenvolvimento de projetos com um grau de complexidade maior do que aqueles que elas seriam capazes de construir fazendo uso apenas do "mundo físico" (Papert, 1993). Dentro desse contexto, Papert criou o LOGO, um software que permite aos usuários, através de linhas de código, movimentar uma "tartaruga", um animal cibernético que pode ser tanto um objeto virtual (presente na tela do computador) quanto um objeto físico manipulável (Papert, 1993). Essa tartaruga deixa um "rastro" (uma linha desenhada) por onde "anda", permitindo ao usuário ter um feedback imediato dos comandos dados por ele ao computador. É neste processo, de reflexão do usuário sobre os resultados concretos dos comandos dados por ele ao computador a partir da observação dos elementos gráficos construídos por meio dos movimentos da tartaruga, que se estabelece a construção do conhecimento.

Por meio da utilização da sua tartaruga robô, o LOGO traz em si uma semente do uso da robótica na educação, ferramenta esta que vem se apresentando como um importante veículo de aplicação do "modo de pensar" Construcionista.

A Robótica Educacional permite à criança criar, manipular e controlar objetos concretos e, através destes, observar a materialização dos comandos dados por ela ao computador, processo este a partir do qual, tomando-se como base as teorias Construtivista (Piaget, 1974) e Construcionista (Papert, 1993), se estabelece, como já mencionado, a construção do seu conhecimento.

Além disso, por tratar-se de uma disciplina de caráter multidisciplinar, a Robótica permite que os alunos trabalhem uma grande diversidade de competências e habilidades à medida que engloba, em um único objeto de estudo, diferentes áreas do conhecimento, como matemática, eletrônica, mecânica, inteligência artificial, artes e programação (Campos, 2011), sendo a última o foco mais específico deste trabalho.

2.2 Robótica Educacional de Baixo Custo no Ensino Fundamental I

Estudos acerca dos kits utilizados no Brasil para o ensino de programação a crianças, por meio da Robótica Educacional, como os realizados por Costa Jr. e Guedes (2015) e por de França e do Amaral (2013), apontam o uso mais frequente de quatro kits: Lego Mindstorms[3], Modelix[4], Robomind[5] e Arduino. Os três primeiros são proprietários e possuem material pedagógico próprio. O Arduino é uma plataforma de baixo custo, baseada em Tecnologia Livre, não tendo sido encontrados, no entanto, materiais pedagógicos para esse kit no levantamento feito por Costa Jr. e Guedes (2015).

Essa inexistência (ou pouca oferta) de material didático apropriado para o uso de Tecnologia Livre (comumente gratuita ou de baixo custo) no aprendizado de programação com robótica no Ensino Fundamental, aliada ao custo relativamente elevado das soluções proprietárias citadas, pode ser percebido como um fator de contribuição para o agravamento da exclusão digital a qual comumente estão expostas as crianças residentes em comunidades de baixa renda, em geral matriculadas na rede pública de ensino.

Atentos a essas questões, pesquisadores têm buscado realizar estudos acerca do desenvolvimento de materiais de robótica acessíveis (tanto quanto possível) a instituições de ensino com baixa disponibilidade de recursos financeiros. Podemos citar, dentre outros, os trabalhos de Sasahara e da Cruz (2007), Medeiros Filho e Gonçalves (2008), de Miranda, Sampaio e Borges (2011), Fabri Junior (2014), Chella (2016) e Ferreira, de Jesus, Rufo e Santos (2016).

No entanto, os ambientes de programação, bem como os materiais de robótica e as atividades didáticas propostas por estes trabalhos, não foram pensados e desenvolvidos visando seu uso especificamente por crianças do Ensino Fundamental I.

Muitos dos trabalhos não apresentam validação prática dos materiais de robótica sugeridos e dos ambientes de programação utilizados e também não apontam que habilidades específicas relacionadas à programação de computadores e ao Pensamento Computacional são possíveis de serem trabalhadas a partir do seu uso. Além disso, não trazem informações quanto à adequação

[3] https://www.lego.com/en-us/mindstorms
[4] http://modelix.cc/
[5] http://www.robomind.com.br/

ou não das soluções propostas a crianças do Ensino Fundamental I (Garneli, Giannakos & Chorianopoulos (2015).

Alguns dos estudos utilizam hardware próprio e/ou estruturas que precisam ser adquiridas no mercado ou exigem maquinário específico para sua construção, o que pode dificultar bastante o desenvolvimento dos robôs sugeridos por professores com pouco ou nenhum conhecimento em eletrônica e/ou impossibilidade de produção/aquisição das peças necessárias para sua construção.

O kit DuinoBlocks4Kids (desenvolvido como parte desta pesquisa), por outro lado, foi desenhado especificamente para o uso por crianças do Ensino Fundamental I, levando em consideração a maturidade cognitiva das mesmas (suas capacidades e "limitações" – ver seção 2.5), tanto no desenho da ferramenta de programação quanto dos materiais de robótica e das atividades didáticas.

No tocante aos materiais de robótica, diferentemente dos trabalhos acima citados, que apresentam o projeto de um robô específico a ser utilizado para o aprendizado de programação, o DB4K apresenta não um robô de baixo custo, mas sim, uma estratégia para desenvolvimento de materiais e atividades de baixo custo baseada no uso de materiais recicláveis e potes plásticos associados aos dispositivos (placas, sensores e atuadores) comumente presentes em kits básicos para aprendizado de programação com Arduino.

Nesta perspectiva, o kit DB4K disponibiliza um conjunto de robôs e atividades didáticas e aponta para habilidades específicas que podem ser trabalhadas a partir do uso desse aparato no aprendizado de programação, sendo que o professor pode, se desejar, com base nos exemplos disponibilizados, desenvolver seus próprios robôs e atividades, adequando-os aos interesses dos seus alunos e aos seus objetivos específicos.

2.3 Linguagens de Programação Visual para Crianças

Junto com os primeiros computadores pessoais (PCs), no fim dos anos 70, surgiu o interesse de se utilizar esses equipamentos nas escolas para o aprendizado de programação. No entanto, a dificuldade de compreensão, por parte das crianças, da sintaxe[6] das linguagens de programação existentes na época, como o BASIC[7], bem como a não conexão dos programas desenvolvidos com os interesses delas acabaram contribuindo para o insucesso de muitas dessas iniciativas (Resnick, et al., 2009).

Desde então, novas iniciativas surgiram no sentido de se tentar vencer estas dificuldades. No que diz respeito ao entendimento da sintaxe das linguagens de programação textuais, uma alternativa encontrada foi o uso de Linguagens de Programação Visuais (*Visual Programming Language* - VLP), ou seja, linguagens nas quais "a sintaxe (semanticamente significativa) inclui expressões visuais" (Burnett, 1999, p. 1).

Com base nesse paradigma, o MIT Media Lab[8], iniciou, em 2003, o desenvolvimento do Scratch[9], uma Linguagem de Programação Visual baseada em "blocos de encaixar" que tinha

[6] Sintaxe, em linguagens de programação, diz respeito ao conjunto de regras que define a forma de uma linguagem, estabelecendo como são compostas as suas estruturas básicas, ou seja, quais palavras e símbolos fazem parte desta linguagem e de que forma eles podem ser combinados e arranjados.
[7] BASIC, acrônimo para "*Beginner's All-purpose Symbolic Instruction Code*", é uma linguagem de programação de "fácil" aprendizagem, desenvolvida na década de 60 para fins didáticos, e que se tornou muito popular nos anos 70 e 80 com o advento dos computadores pessoais.
[8] https://llk.media.mit.edu/
[9] https://scratch.mit.edu/

por objetivo permitir que qualquer pessoa, de qualquer idade, pudesse programar. O sucesso do projeto foi tal que, em 2009, apenas dois anos depois do lançamento do site, usuários de todas as partes do mundo, na sua maioria crianças e jovens de 8 a 16 anos, já faziam *upload* de mais de 1500 projetos por dia (Resnick, et al., 2009).

Além do Scratch, outros projetos de grande vulto, voltados ao ensino de programação para crianças, também adotam o conceito da Programação Visual por meio de blocos de encaixar. Dentre eles podemos citar o Code.org[10], que possui suas próprias ferramentas de ensino de programação por blocos e o Programaê[11] e Code Club Brasil[12], que utilizam o Scratch em seus programas de aprendizado de programação.

No que tange à programação de placas Arduino (tecnologia adotada neste trabalho), como mencionado na seção 1, não foi encontrado um Ambiente de Programação Visual voltado à programação destas placas, pensado e desenvolvido especificamente para o ensino de programação a crianças do Ensino Fundamental I. Ou seja, um Ambiente de Programação Visual composto por blocos graficamente mais atrativos e dotados de uma semântica menos abstrata que a empregada nos ambientes de programação que trabalham com o mesmo hardware.

Como exemplos de Ambientes de Programação Visual para placas Arduino atualmente existentes, mas, que não foram projetados para serem usados por crianças, podemos citar o DuinoBlocks (Alves, Sampaio & Elia, 2013) que serviu de inspiração para o desenvolvimento do DB4K; O S4A[13] - Scratch for Arduino , um "fork"[14] do Scratch que permite a programação de placas Arduino; e o Ardublockly[15], ambiente sobre o qual foi desenvolvido o DuinoBlocks4Kids (ver seção 3.2).

Algumas características do ambiente de programação visual em blocos para Arduino DuinoBlocks4Kids (desenvolvido como parte deste trabalho) que o diferenciam dos ambientes de programação em blocos acima citados e que buscam tornar o seu uso mais acessível e aprazível a crianças do Ensino Fundamental I são: blocos de programação desenhados de modo a apresentarem uma semântica diretamente relacionada com os dispositivos sendo manipulados e com os efeitos por eles causados sobre esses dispositivos; supressão de detalhes relacionados ao hardware, como pinagens e valores de níveis de tensão; uso de linguagem icônica; conjunto "enxuto" de blocos; simplificação dos valores dos parâmetros utilizados nos blocos, como por exemplo: temperatura (alta ou baixa), luz (muita ou pouca), velocidade (rápida, média ou lenta).

2.4 Pensamento Computacional

"Pensar Computacionalmente" é reformular um problema aparentemente difícil de maneira a "transformá-lo" em um que saibamos resolver, é prevenir erros e estar pronto para corrigi-los, revisando cada etapa realizada na busca pela solução de um problema. É saber planejar na presença de incertezas e entender que é possível trabalhar de forma segura com problemas complexos sem precisar conhecer todos os seus detalhes (Wing, 2006).

[10] https://code.org/
[11] http://programae.org.br/
[12] http://codeclubbrasil.org/
[13] http://s4a.cat/
[14] *Forks* são softwares desenvolvidos a partir do código fonte de outros softwares, dando origem a um novo projeto da mesma "linha", mas, independente do projeto do qual se originou.
[15] https://ardublockly.embeddedlog.com/index.html

Para Hemmendinger (2010), o objetivo do desenvolvimento do Pensamento Computacional não é o de fazer com que todos passem a pensar como cientistas da computação, mas sim, habilitar as pessoas a aplicarem esta maneira específica de raciocinar na busca por novos questionamentos e na solução de diversos tipos de problemas nas mais variadas áreas do conhecimento.

Embora não haja ainda consenso acerca do conjunto exato de habilidades relacionadas ao Pensamento Computacional, de acordo com Grover e Pea (2013), a maior parte dos pesquisadores e educadores da área de Ciências da Computação têm aceitado, de maneira bastante ampla, esta forma de pensamento como sendo compreendida pelos seguintes elementos:

a) Abstração e Generalização de Padrões: A capacidade de abstração é "o processo mais importante e de nível mais elevado do Pensamento Computacional, [...] usado na definição de padrões, na generalização de instâncias específicas e parametrização [...]" (Wing, 2011, p. 20), sendo considerada a "pedra fundamental" desta forma de pensamento (Grover & Pea, 2013).

b) Processamento sistemático de informações: O processamento sistemático de informações se caracteriza pela busca do entendimento completo das informações disponíveis a cerca de um determinado problema com atenção meticulosa, pensamento profundo e raciocínio intenso. As informações "coletadas" por esse processo são então combinadas e usadas para a tomada de decisões (Chaiken & Ledgerwood, 2012).

c) Sistemas simbólicos e representações: Os sistemas simbólicos, ou sistemas de sinais, são conjuntos de sinais distinguíveis uns dos outros e cuja construção pode ser reproduzida. Exemplos de sistemas de sinais são os caracteres do alfabeto, o sistema numérico arábico e as linguagens de programação. Neste contexto, a representação (núcleo da significação) pode ser entendida como a "criação de significado" para os símbolos. (Maiers, W., Bayer, B., Esgalhado, B.D., Jorna, R. & Schraube, E., 1999, Elleström, 2014).

d) Noções algorítmicas de fluxo de controle: Fluxo de controle é o termo utilizado para decidir quais comandos do programa são executados em qual ordem" (Arnold, James, & David, 2009, p. 36).

e) Decomposição estruturada de problemas: Decomposição de problemas é a atividade de "desmembrar um problema em partes menores que possam ser mais facilmente resolvidas" (Barr & Stephenson, 2011, p. 52).

f) Pensamento paralelo, recursivo e iterativo: A essência da definição de processamento paralelo, ou *paralelismo*, seria a de se usar mais de um computador trabalhando em um mesmo problema que possa ser dividido em partes passíveis de serem resolvidas ao mesmo tempo (Sasikumar, Shikhare & Prakash, 2014). *Recursividade*, em computação, significa autorreferência. É um conceito poderoso utilizado no desenvolvimento de programas nos quais uma função (uma sequência de comandos que executa uma determinada tarefa) evoca a si mesma para solucionar um problema (Gupta, Agarwal & Varshney, 2007). E, por fim, *Iteração*, se refere à repetição de uma ação ou conjunto de ações. Este conceito é utilizado nos programas para que a execução de um trecho de código (sequência de instruções) seja repetida um certo número de vezes (Dimes & Rosa, 2016).

g) Lógica condicional: A lógica condicional é utilizada em programação para que um conjunto de instruções seja executado somente quando, enquanto ou até que determinadas condições sejam satisfeitas. A satisfação ou não dessas condições acarreta

na tomada de diferentes decisões que levam a lógica do programa a seguir diferentes fluxos (Bhattacharya, 2016).

h) **Restrições de eficiência e performance**: Os computadores possuem limitações de performance, tais como a velocidade do processador e da entrada e saída de dados entre o processador e memória (Filho, 2007) que acabam por impor restrições quanto à eficiência das soluções planejadas. Esses limites precisam ser levados em consideração quando do desenvolvimento de sistemas computacionais, especialmente aqueles dedicados à solução de problemas mais complexos.
i) **Depuração e detecção sistemática de erros**: "Depuração é a atividade [...] de encontrar e eliminar os erros (bugs) de um programa" (Aguilar, 2008, p. 672).

Como se pode observar, o Pensamento Computacional abarca muitos elementos que, para serem trabalhados, necessitam do exercício e desenvolvimento de uma gama ampla de habilidades. Para se abarcar todo esse conjunto de habilidades torna-se necessária a realização de um trabalho extenso e aprofundado de diferentes aspectos relacionados às Ciências da Computação, ou seja: Pensar Computacionalmente, ao contrário do que se pode imaginar em um primeiro momento, não se restringe à habilidade de programar computadores. No entanto, tem-se na programação (competência fundamental da Ciência da Computação) uma importante ferramenta de apoio ao desenvolvimento e exercício de algumas das habilidades cognitivas características do Pensamento Computacional (Grover & Pea, 2013), e é nesse âmbito que se insere a presente pesquisa.

2.5 O Período Operatório Concreto e o Aprendizado de Programação

Jean Piaget distingue quatro períodos gerais no que se refere ao desenvolvimento cognitivo, sendo eles: sensório motor (0 a 2 anos), pré-operatório (2 a 7/8 anos), operatório concreto (7/8 a 11/12 anos) e operatório formal (11/12 anos em diante) (Moreira, 1999).

No período operatório concreto, momento em que, em princípio, se encontravam os participantes do presente estudo, as crianças começam a realizar operações mentalmente e não apenas por meio de ações físicas, como acontece no período pré-operatório (Terra, 2010). Em outras palavras: "o sujeito se torna capaz de reconstruir no plano da representação o que já havia construído no plano da ação" (Souza, 2014, p. 144). Outro aspecto fundamental, em relação a esse período, é que, para realizar as operações, a criança recorre sempre a "objetos concretos" presentes ou já experimentados (daí a designação operatório concreto), sendo bastante limitada a realização de operações concretas em direção ao ausente (Furtado, Bock & Teixeira, 2001).

A capacidade de abstração, considerada como sendo a "pedra fundamental" do Pensamento Computacional (Grover & Pea, 2013), é, desse modo, ainda limitada neste período, tendo-se o predomínio do uso da abstração empírica (ou simples), que consiste na capacidade da criança de focar em uma determinada propriedade de um objeto e ignorar as demais (Kamii, 1992), bem como de construir raciocínios a partir da abstração de objetos pertencentes ao seu universo (Lister, 2011). Já o pensamento hipotético-dedutivo, característico do período operatório formal, para o qual seria necessário a construção de abstrações a partir de hipóteses (abstração reflexiva), e não com base em situações familiares, tende a não aparecer nesse período (Lister, 2011).

Embora limitado à realização de construções mentais a partir de referenciais concretos, o uso da abstração empírica se encaixa perfeitamente em um dos aspectos relacionados à capacidade de abstração apresentados por Kramer (2007) como sendo fundamentais para os cientistas da computação, qual seja: o processo de remover detalhes para simplificar e focar a

atenção, seja “retirando alguma coisa” ou desconsiderando uma ou mais propriedades de um objeto complexo para prestar atenção em outras.

Outra questão a ser considerada neste sentido é o fato de algumas estruturas de programação, como as de repetição (contada e condicional) e de decisão (simples e composta), possuírem aspectos que podem ser observados ou experienciados, tanto por meio de brincadeiras que não necessitem do uso do computador, quanto com a realização de atividades de programação cujos resultados possam ser visualizados e apresentem de forma bastante evidente uma relação direta com o “comando” que os produziu. Assim sendo, é bastante razoável acreditar que a realização dessas abstrações seja perfeitamente acessível a crianças do período operatório concreto.

Além das questões ligadas à capacidade de abstração, uma análise das características cognitivas de crianças no período operatório concreto nos possibilita aventar a possibilidade de existência de uma relação direta entre algumas dessas competências e certas habilidades relacionadas à programação de computadores, como por exemplo:

- A habilidade de sequenciar eventos ou ideias possibilitaria o desenvolvimento de programas puramente sequenciais, que, grosso modo, nada mais são do que um sequenciamento de comandos simples “posicionados” de forma a produzir um resultado esperado.
- Por construírem seu conhecimento a partir de uma ação, e não a partir de um conceito, é necessário, em princípio, que as crianças operacionais concretas vejam “fisicamente” o resultado da execução de um comando para então poder conceber mentalmente a relação entre o comando utilizado no programa e o resultado obtido. Mas, uma vez isso feito, a habilidade delas de realizar operações mentais (a partir de experiências concretas) e não apenas por meio de ações físicas, e de pensar simultaneamente no todo e nas suas partes, possibilitaria a essas crianças construírem mentalmente a sequência de eventos resultantes de um conjunto de comandos utilizados no programa. Ou seja, essa habilidade permitiria às crianças operacionais concretas a construção de um trecho completo de código (composto por um determinado número de comandos) e a “visualização mental” do resultado da execução desse trecho de código sem a necessidade de observarem “fisicamente”, a todo instante, o resultado da execução individual de cada um dos comandos.
- A capacidade de estabelecer corretamente relações de causa e efeito, meio e fim, possibilitaria trabalhar-se com crianças no período operatório concreto a ideia de processamento, qual seja, de que os comandos presentes em um programa são o “meio” pelos quais se obtém o resultado observado quando da execução desse programa, ou seja, o “fim”.
- Ser capaz de trabalhar simultaneamente com mais de um ponto de vista sobre uma mesma ideia, possibilitaria às crianças operacionais concretas perceberem e compreenderem que um mesmo problema pode ter mais de uma solução, e que uma mesma solução pode ser aplicada a mais de um problema, ou seja, que um mesmo resultado pode ser obtido a partir de diferentes programas e que um mesmo programa (ou “trecho de código”) pode ser utilizado para solucionar mais de um problema.
- A reversibilidade possibilitaria a essas crianças serem capazes de, a partir da constatação de que algo no resultado obtido não se deu como o esperado, modificar ou retirar do programa apenas os comandos que elas acreditam ser a causa do problema, construindo o programa correto a partir desse ponto, ou seja, sem desfazer-se de todo o código desenvolvido anteriormente para reconstruir, a partir do

começo, passo a passo, um novo programa. Além disto, permitiria a elas fazerem o "reaproveitamento de código", quando os objetivos de um programa sendo construído diferirem apenas parcialmente de um programa previamente desenvolvido.

3 Materiais e métodos

O método de pesquisa adotado neste trabalho foi a realização de um estudo de caso único, exploratório com observação participante (Yin, 2001). O caso estudado foi uma oficina de aprendizado de programação com robótica com crianças do Ensino Fundamental I, pertencentes a uma comunidade de baixa renda da cidade do Rio de Janeiro e matriculadas em escolas públicas, na qual foi adotado o kit didático DuinoBlocks4Kids.

O estudo é exploratório pois procura gerar mais informações sobre o fenômeno investigado, buscando levantar hipóteses que possam orientar estudos posteriores (Meirinhos & Osório, 2010) e a observação realizada é participante pois o pesquisador foi ao mesmo tempo investigador e participante (Yin, 2001), tendo assumido o papel de professor na oficina realizada.

Como aporte teórico utilizado para nortear as observações e sustentar as inferências feitas a partir delas, foram utilizados o conceito de Pensamento Computacional delineado por Jeannette Wing (2006) e a teoria de desenvolvimento cognitivo de Jean Piaget (Moreira, 1999).

3.1 Coleta de dados e avaliação

A coleta de dados foi realizada por meio do registro das ações dos usuários e observações de campo. O registro das dinâmicas dos alunos no ambiente de trabalho foi feito com ouso de uma filmadora, e um software de captura de tela foi utilizado para gravar as ações realizadas pelas crianças em seus computadores. Para complementar os registros em vídeo fez-se uso de um "diário de bordo" no qual eram realizadas anotações acerca das observações feitas no decorrer de cada encontro.

Esses dados foram coletados durante uma oficina de aprendizado de programação com robótica realizada com sete crianças (cinco meninos e duas meninas) residentes em uma comunidade de baixa renda da cidade do Rio de Janeiro, sem experiência prévia em programação de computadores e matriculadas em escolas públicas, sendo quatro delas do 4º ano e três do 3º ano. A Oficina contou com 14 encontros, de 90 minutos cada, e utilizou como material didático o Kit DuinoBlocks4Kids, desenvolvido como parte desta pesquisa (ver seção 3.2).

Os dados coletados durante a oficina foram então utilizados para a avaliação do aprendizado de estruturas básicas de programação e sua relação com aspectos do Pensamento Computacional (ver seção 2.4), bem como para a investigação das hipóteses levantadas sobre uma possível relação direta existente entre a maturidade cognitiva de crianças no estágio operatório concreto e a realização de determinadas atividades relacionadas ao aprendizado de programação (ver seção 2.5). Os resultados destas análises encontram-se disponíveis na seção 4.

A avaliação do aprendizado das estruturas básicas de programação seguiu o modelo utilizado pelo grupo DevTech da TUFFS University[16] (Bers, Flannery, Kazakoff & Crouser, 2010). Esse modelo faz uso de uma escala Likert com os níveis apresentados no Quadro 1.

Essa escala é aplicada a itens referentes à aquisição ou exercício das habilidades pretendidas em cada aula, como no exemplo presente no Quadro 2.

Quadro 1: Escala Likert utilizada para a avaliação do aprendizado.

5	4	3	2	1	0
Aquisição ou realização completa do objetivo, tarefa ou conteúdo	Aquisição ou realização quase completa do objetivo, tarefa ou conteúdo	Aquisição ou realização parcial do objetivo, tarefa ou conteúdo	Aquisição ou realização bastante incompleta do objetivo, tarefa ou conteúdo	Não aquisição ou realização do objetivo, tarefa ou conteúdo	Nem sequer tentou

Quadro 2: Exemplo de itens de referência para avaliação de habilidades específicas [17].

1	Entende o funcionamento do sensor de distância	5	4	3	2	1	0
2	Monta corretamente o circuito proposto	5	4	3	2	1	0
3	Compreende a função do bloco *enquanto <condição> faça*	5	4	3	2	1	0
4	Utiliza adequadamente o bloco *enquanto <condição> faça*, inserindo o bloco com o sensor correto como fator condicionante	5	4	3	2	1	0
5	Programa corretamente o motor DC e o LED em conjunto com o sensor de distância	5	4	3	2	1	0
Obs.							

Em associação à avaliação das habilidades adquiridas ou exercitadas, é feita também uma avaliação acerca da habilidade de depuração e correção dos programas desenvolvidos, conforme o modelo presente no Quadro 3.

Quadro 3: Itens de referência para avaliação da habilidade de depuração e correção dos programas desenvolvidos.

1	Percebe que alguma coisa não está funcionando como esperado	5	4	3	2	1	0	NA
2	Conserva a meta original	5	4	3	2	1	0	NA
3	Tem uma hipótese para a causa do problema	5	4	3	2	1	0	NA
4	Tenta solucionar o problema	5	4	3	2	1	0	NA
Obs.								

NA = Não se aplica.

A avaliação do desenvolvimento ou exercício das habilidades do Pensamento Computacional, bem como de outros aspectos relacionados à maturidade cognitiva das crianças, foi realizada a partir das competências que se entendia como sendo necessárias para a construção adequada dos programas propostos.

No contexto desta pesquisa, entendeu-se que seria possível trabalhar, por meio do aprendizado das estruturas básicas de programação abordadas na oficina realizada para a coleta de dados, as seguintes habilidades (ver seção 2.4):

[16] http://ase.tufts.edu/devtech

[17] Os itens de avaliação específicos para cada uma das aulas encontram-se disponíveis em: http://ginape.nce.ufrj.br/LIVRE/paginas/dissertacoes/d_2017_rubens_queiroz.pdf

a) **Capacidade de abstração**: Como um dos exemplos de abstração ligados à Ciência da Computação, Wing (2011) cita o algoritmo, definido por ela como sendo "a abstração de um processo que pega entradas, executa uma sequência de passos e produz saídas para satisfazer a um objetivo desejado" (Wing, 2011, p. 20). Tendo em vista essa definição, no escopo desse trabalho, a capacidade de abstração foi avaliada com base na habilidade das crianças de observarem a execução ou descrição de um processo do mundo real e abstrair esse processo na forma de um programa de computador.

b) **Compreensão de Fluxos de Controle**: Nos programas construídos com o Duinoblocks4Kids (ver seção 3.2.2), cada bloco representa um comando, sendo esses comandos executados pelo computador, um a um, em um loop infinito, do primeiro bloco (posicionado no topo da "pilha" de comandos), até o último bloco (posicionado na base da pilha). Alguns blocos, como o *Acender LED* e o *Girar motor DC*, representam "comandos simples". Já os blocos de controle: *repetir*, *enquanto*, *se* e *se/senão*, representam estruturas que podem ser vistas, sintaticamente, como um único comando composto por um conjunto de comandos que podem ou não ser realizados uma ou *n* vezes. Tanto a construção de programas puramente sequenciais, ou seja, que utilizam apenas "comandos simples", quanto a criação de programas mais elaborados, que façam uso de estruturas de repetição contada ou condicional, por exemplo, só se fazem possíveis se as crianças forem capazes de assimilar algumas noções básicas de fluxo de controle.

c) **Pensamento Iterativo**: O pensamento iterativo pode ser trabalhado no DB4K por meio de programas que façam uso do bloco *repita*, bem como por meio do uso do bloco *enquanto*.

d) **Uso da Lógica Condicional**: Para a construção de programas que façam uso da estrutura de repetição condicional *enquanto*, trabalhada durante a oficina, é necessário que as crianças consigam desenvolver um entendimento, ainda que rudimentar, acerca do uso da lógica condicional.

e) **Decomposição de problemas**: Mesmo programas básicos, que contenham, por exemplo, uma sequência de poucos comandos simples e dois laços de repetição, cada um destes contendo também uma sequência de poucos comandos, podem ser divididos em, digamos, 3 partes menores (ver exemplo na Figura 1), de modo que cada uma dessas partes possa ser desenvolvida e testada separadamente, diminuindo assim a complexidade do que precisa ser analisado, por exemplo, na busca por erros.

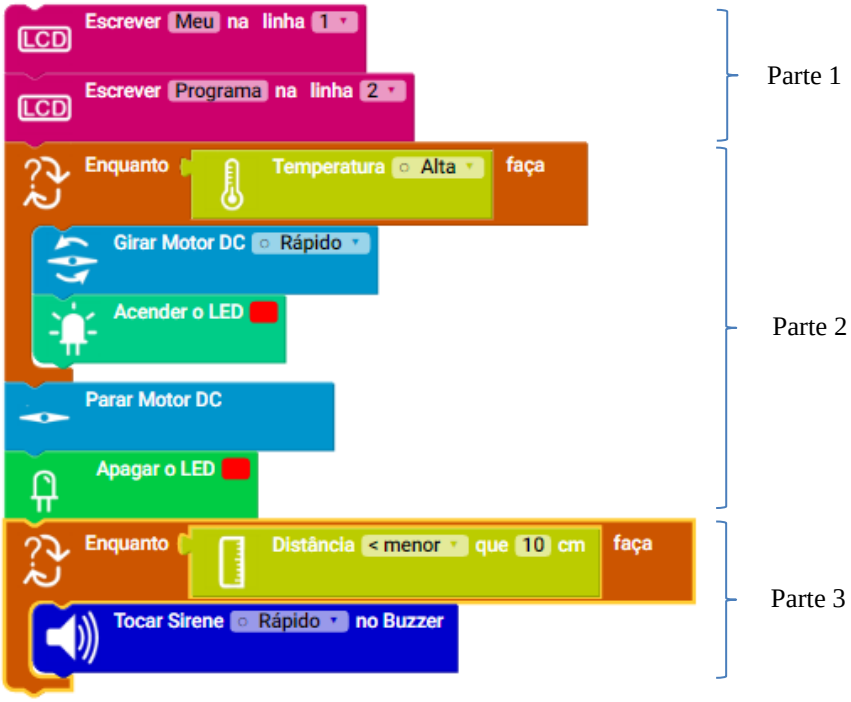

Figura 1: Exemplo de decomposição de problemas.

A *parte 1* do programa exemplo (Figura 1) é responsável pela escrita do texto "Meu Programa" no display LCD presente no "robô". A *parte 2* é responsável pelo comportamento a ser assumido pelo robô em resposta aos valores de temperatura lidos através do seu sensor de temperatura e, por fim, a *parte 3* é responsável pela definição das ações a serem realizadas pelo robô em resposta à aproximação de algum objeto. Cada uma dessas partes do programa define um comportamento específico do robô, podendo ser desenvolvida e testada separadamente. A combinação desses três comportamentos mais simples em um único programa estabelece o comportamento final, mais complexo, esperado para o robô.

Esse gênero de exercício poderia auxiliar as crianças a perceberem a possibilidade do uso desse tipo de estratégia na solução de problemas de outra natureza.

f) **Depuração**: O Ambiente de Programação em Blocos DB4K não possibilita a ocorrência de erros de compilação. Desse modo, a depuração fica focada na detecção e correção de possíveis erros na lógica do programa construído. O trabalho conjunto do Ambiente de Programação em Blocos com os materiais de robótica utilizados na oficina (ver seção 3.2.3) possibilita um exercício inicial acerca da depuração sistemática de erros a partir da realização da sequência de atividades presente na Figura 2. Este exercício traz para as crianças, dentre outros benefícios, a possibilidade de uma maior autonomia em relação ao seu aprendizado, bem como o entendimento de que uma falha pode apontar o caminho para um resultado positivo (Barr & Stephenson, 2011).

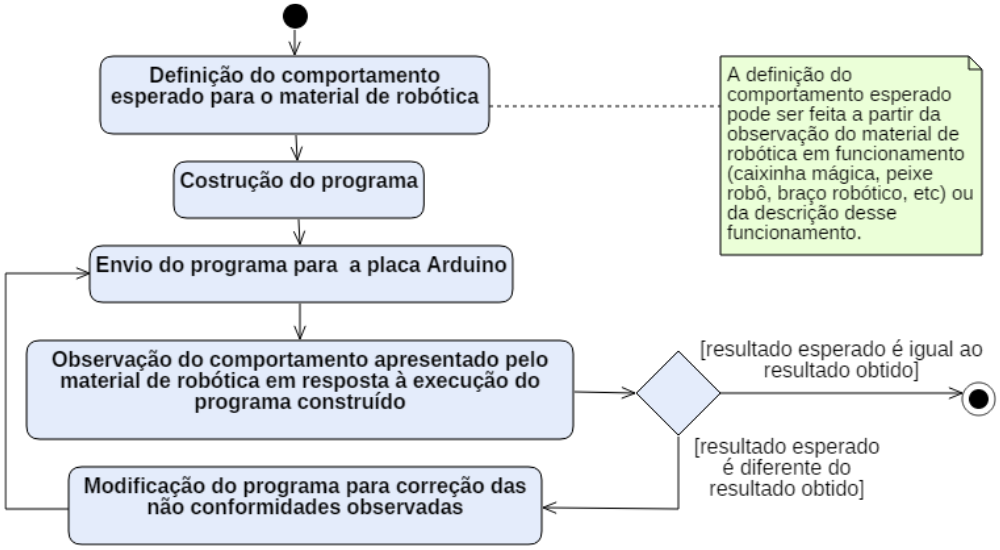

Figura 2: Diagrama de Atividades da tarefa de depuração de erros.

Por fim, as estruturas de programação que se buscou verificar serem viáveis de se trabalhar com as crianças por meio do Kit desenvolvido e através das quais, supunha-se, poderiam ser exercitadas as habilidades acima descritas são a **programação sequencial**, a **estrutura de repetição contada** (repita <n> vezes), a **estrutura de repetição condicional** (Enquanto <condição> faça) e a **estrutura de decisão** (Se <condição> então).

3.2 O Kit DuinoBlocks4Kids (DB4K)

Como já mencionado, o Duinoblocks4Kids (desenvolvido pelos autores desta pesquisa) é um Kit didático para o aprendizado de programação via Robótica Educacional, baseado no uso de Tecnologia Livre e materiais recicláveis e de baixo custo, composto por um Ambiente de Programação Visual baseado em blocos, um conjunto de materiais de Robótica Educacional e uma série de atividades.

3.2.1 O Ambiente de Programação Visual em Blocos

O ambiente de programação visual em blocos DuinoBlocks4Kids[18] (Figura 3) foi desenvolvido a partir do Ardublockly[19], um ambiente de programação visual para placas Arduino que faz uso de uma biblioteca criada pela Google Developer[20], especificamente para o desenvolvimento desse tipo de ambiente, chamada Blockly[21].

[18] O ambiente de Programação Visual DB4K, juntamente com todos os códigos-fonte, encontra-se disponível para download no seguinte endereço: http://ginape.nce.ufrj.br/LIVRE/paginas/db4k/db4k.html
[19] https://ardublockly.embeddedlog.com/
[20] https://developers.google.com/
[21] https://developers.google.com/blockly/

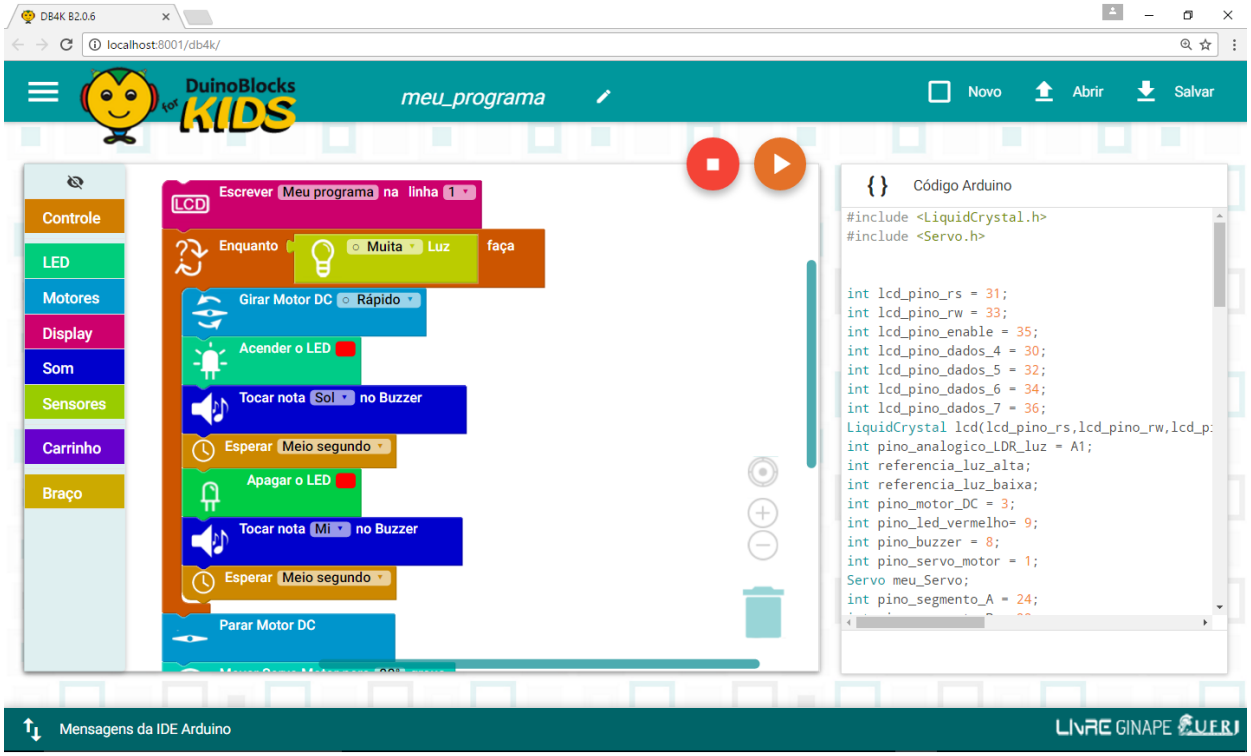

Figura 3: Visão geral da Interface do DB4K.

O Ambiente DB4K permite, por meio de seus blocos, a programação de um conjunto de LEDs, 1 LED RGB, 1 display de 7 segmentos, 1 display LCD, 1 servo motor, 1 motor DC, 1 buzzer, 1 sensor de distância, 1 sensor de luz e 1 sensor de temperatura, além de possuir blocos específicos para o controle de um carro-robô e um braço robótico (Ver Seção 3.2.3). Além dos blocos diretamente relacionados com os dispositivos a serem manipulados, foram também desenhados blocos para as estruturas de controle utilizadas em programação, como repetição e decisão.

Para que o usuário possa, caso deseje, ter acesso à linguagem textual Wiring[22] associada a cada bloco do DuinoBlocks4Kids, decidiu-se por disponibilizar, na interface do DB4K, além da área de programação com blocos, uma área para a exibição do programa textual correspondente ao programa criado por meio de seus blocos.

O diagrama de casos de uso presente na Figura 4 mostra as principais funcionalidades do DB4K, tendo sido destacadas, com a cor azul, as mais comumente utilizadas durante as atividades de aprendizado de programação.

[22] Linguagem nativa para a programação das placas de prototipagem eletrônica Arduino.

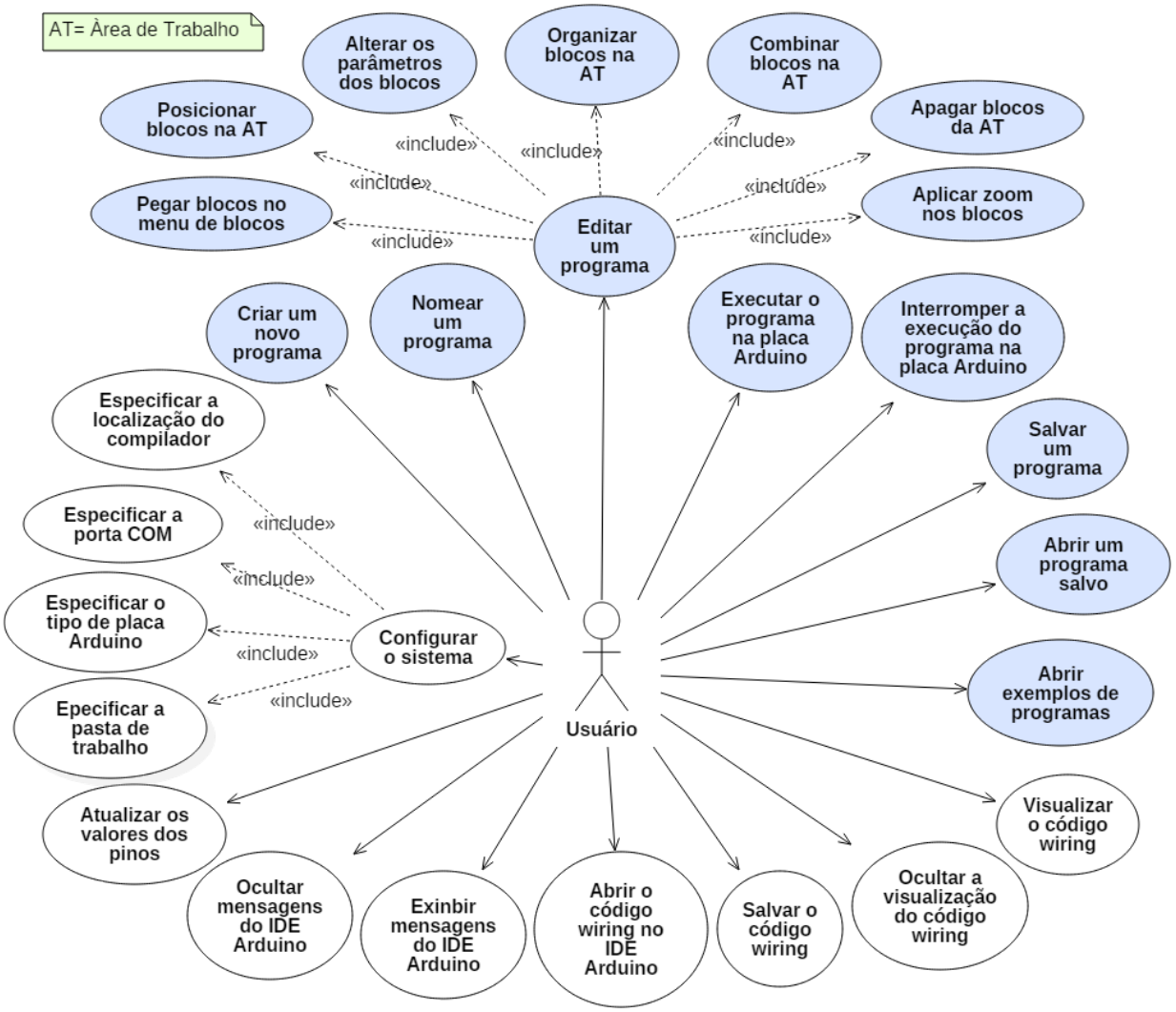

Figura 4: Diagrama de casos de uso com as principais funcionalidades do DB4K.

Dentre os casos de uso destacados em azul, encontram-se as funcionalidades utilizadas na principal sequência de ações efetuada pelas crianças na interface do ambiente de programação quando da realização dos exercícios de programação dos materiais de robótica (ver seção 3.2.3). Esta sequência pode ser observada no diagrama presente na Figura 5.

3.2.2 As Atividades Didáticas

Por meio de trabalhos práticos, aulas expositivas dialogadas, brincadeiras e narrativas, as atividades propostas visam, em associação com o ambiente de programação em blocos e com os materiais de robótica, trabalhar as estruturas básicas de programação apresentadas na seção 3.1, quais sejam: sequenciamento de comandos, repetições contadas, repetições condicionais e estruturas de decisão.

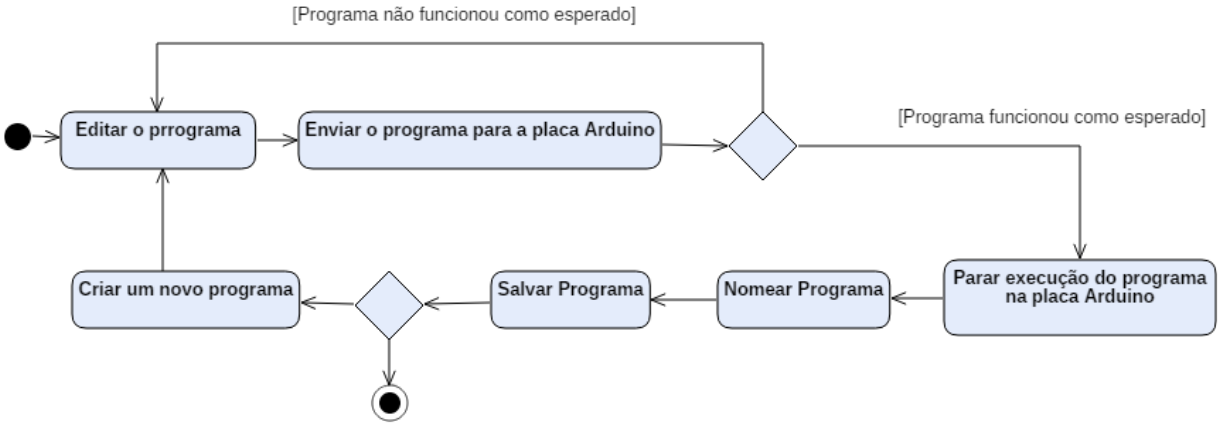

Figura 5: Diagrama da principal sequência de ações realizadas pelas crianças na interface do ambiente de programação DB4K.

O desenho das atividades foi inspirado nos trabalhos de (Bers, Flannery, Kazakoff & Crouser, 2010) e (Sullivan & Bers, 2016) do Grupo de Pesquisa DevTec[23] da Tufs University[24].

As atividades práticas referem-se à montagem e programação de pequenos circuitos e, também, à programação de um conjunto de materiais de robótica. Alguns desses materiais serão apresentados na seção 3.2.3. As brincadeiras, inspiradas na técnica de Computação Desplugada (Bell, Alexander, Isaac & Grimley, 2009), visam apresentar, de maneira lúdica e divertida, as ideias de sequenciamento e repetições de ações, bem como de tomada de decisão (utilizadas nas estruturas condicionais), conceitos esses que são posteriormente trabalhados no ambiente de programação em blocos em conjunto com os materiais de robótica. As narrativas são utilizadas com os robôs de garrafa pet (ver seção 3.2.3) para se contextualizar o uso dos dispositivos de robótica e estruturas de programação sendo trabalhados[25].

No âmbito da educação, o uso de narrativas cria situações de aprendizado e desenvolve formas criativas de se resolver problemas (Butcher, 2006), configurando-se como uma ferramenta poderosa e versátil de ativação do aprendizado e engajamento dos alunos (Szurmak & Thuna, 2013). Dentre os benefícios relacionados ao uso de narrativas no ensino, possuem especial relevância:

- Narrativas tornam algo abstrato mais concreto/direto;
- Narrativas contextualizam as informações à medida em que criam um arcabouço no qual os alunos podem acomodar os novos conhecimentos (e assim ampliar sua retenção e entendimento);
- Narrativas possibilitam aos estudantes terem experiências emocionais mais imediatas com as quais eles podem se relacionar (e, consequentemente, lembrar)
  (Szurmak & Thuna, 2013, p. 550).

No kit DB4K foram adotadas narrativas fantásticas nas quais robôs (feitos de garrafas pet e outros materiais recicláveis e de baixo custo) são as personagens principais.

[23] http://ase.tufts.edu/devtech/
[24] https://www.tufts.edu/
[25] O detalhamento dos planos de aula, com a especificação e descrição de todas as atividades realizadas, bem como dos conteúdos abordados, materiais adotados, desenvolvimento metodológico e itens de avaliação de aprendizado específicos para cada aula, encontra-se disponível na dissertação de Mestrado de Queiroz (2017) e no site do LIvRE (http://ginape.nce.ufrj.br/LIVRE)

3.2.3 Os Materiais de Robótica

Para uso com o ambiente de programação visual, compondo também o Kit DB4K, foi desenvolvido um conjunto de materiais de robótica. Um deles é uma caixinha plástica batizada de caixinha mágica (Figura 6) que possui todos os dispositivos programáveis via DB4K já devidamente conectados à uma placa Arduino, o que possibilita às crianças o desenvolvimento de programas mais elaborados para o controle de circuitos cuja construção seria muito complexa para ser realizada por elas em aula.

Figura 6: A "caixinha mágica" contendo os seguintes componentes: 1 display LCD, um display de 7 segmentos, 1 buzzer, 1 servo motor, 1 motor DC, 4 LEDs, 1 LED RGB, 1 sensor de distância, 1 sensor de luz e 1 sensor de temperatura.

Além da caixinha, foram desenvolvidos três robôs de garrafa pet utilizados nas atividades com narrativas, mencionadas na seção anterior, como um protótipo de peixe-robô, um morcego-robô e um cachorro-robô (Figura 7).

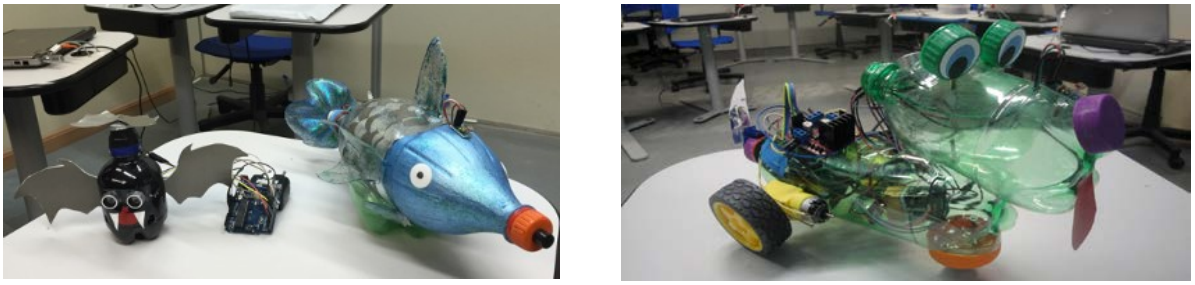

Figura 7: Morcego-robô, Peixe-Robô e Cachorro-Robô.

Com a caixinha mágica as crianças conhecem e a aprendem a programar os dispositivos de robótica. E a contextualização do uso destes dispositivos, como mencionado anteriormente, é feita por meio dos robôs de garrafa pet.

Esses robôs são totalmente produzidos com materiais recicláveis e de baixo custo (como garrafas pet, papéis coloridos, cola, fita adesiva, entre outros) e fazem uso de dispositivos comumente presentes em kits básicos para aulas de Robótica com Arduino, que podem ser reaproveitados de um robô para o outro. No que concerne à construção desses materiais, não é necessário que os professores possuam qualquer conhecimento ou habilidade especial além daqueles já necessários para a realização de aulas introdutórias de Robótica Educacional com Arduino[26].

[26] Tutoriais em vídeo para a construção dos materiais de robótica encontram-se disponíveis em: https://www.youtube.com/qrzBots. Os professores podem desenvolver novos materiais e narrativas inspirados nos exemplos sugeridos nos planos de aula, de maneira a adequar as atividades aos interesses dos alunos.

Como já mencionado, o ambiente possibilita ainda a programação de um carrinho-robô e um braço robótico (Figura 8) de baixo custo e facilmente encontrados no mercado.

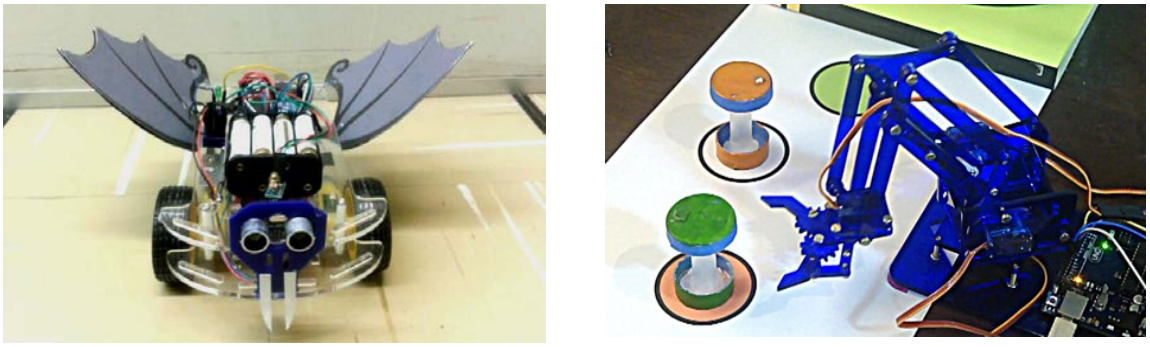

Figura 8: Carro-Robô e Braço Robótico Programáveis via DB4K.

4 Resultados

Para a apresentação dos resultados referentes ao aprendizado das estruturas de programação e exercício de habilidades do Pensamento Computacional, optou-se por utilizar as estruturas de programação como elemento norteador, apresentando-se então, para cada uma delas, os resultados observados em relação ao exercício do Pensamento Computacional e das habilidades cognitivas características do Período Operatório Concreto apresentadas na seção 2.5. O Quadro 4 apresenta uma síntese desses resultados[27].

Quadro 4: Síntese dos resultados relacionados ao aprendizado de programação.

		ES	ERCont	ERCond
PC	Capacidade de abstração	✓	✓	✓
	Depuração e detecção sistemática de erros	✓	✓	✓
	Noções algorítmicas de fluxo de controle	✓	✓	✓
	Pensamento Iterativo		✓	✓
	Uso da lógica condicional			✓
	Decomposição de problemas			✓
OC	Realização de operações mentais	✓	✓	✓
	Sequenciamento de eventos	✓	✓	✓
	Percepção de relações de causa e efeito	✓	✓	✓
	Reversibilidade		✓	✓
	Distinção do todo e das suas partes		✓	✓
	Trabalhar com mais de um ponto de vista sobre uma mesma ideia		✓	✓

PC = Pensamento Computacional; **OC** = Habilidades Cognitivas Características do Período Operatório Concreto; **ES** = Estrutura Sequencial; **ERCont** = Estrutura de Repetição Contada; **ERCond** = Estrutura de Repetição Condicional; ✓= Habilidade Exercitada/Demonstrada.

4.1 Estrutura de Programação Sequencial

A *estrutura de programação sequencial* foi trabalhada por meio de brincadeiras baseadas na técnica de computação desplugada (Bell, Alexander, Isaac & Grimley, 2009) e da construção de programas no ambiente DB4K (ver seção 3.2.1) para o controle de LEDs e motores na caixinha

[27] Os dados coletados possibilitaram uma avaliação bastante minuciosa de todos os aspectos investigados. O detalhamento desses dados, bem como as avaliações e inferências realizadas a partir deles encontram-se disponíveis na dissertação de Mestrado de Queiroz (2017).

mágica (ver seção 3.2.3). Inicialmente foram construídos programas de um único comando e, posteriormente, programas compostos por uma sequência simples de ações sequenciais, como por exemplo:

- Fazer o motor DC girar rápido por 10 segundos com o LED vermelho aceso e depois parar o motor, apagar o LED e parar o programa (Figura 9.a).
- Acender o LED azul, esperar 1 segundo, depois apagar o LED azul e esperar 1 segundo (Figura 9.b).

Para a construção de programas puramente sequenciais, ainda que fazendo uso de poucos comandos, é preciso que se consiga construir mentalmente o resultado esperado e depois, por meio de um programa, organizar o conjunto de ações que, acredita-se, levará a esse resultado, para só então "solicitar" ao computador que realize a tarefa desejada. Esta pode ser uma atividade não muito trivial para algumas crianças entre oito e nove anos, uma vez que, por estarem no início do período operatório concreto, a habilidade de realizar operações mentais (ainda que apoiadas em experiências concretas) e de conseguir distinguir o todo e suas partes (importantes para a realização deste tipo de tarefa) podem não estar suficientemente desenvolvidas. Especialmente se for levado em consideração que a maturidade cognitiva é dependente “das características biológicas do indivíduo e de fatores educacionais, sociais” (Furtado, Bock & Teixeira, 2001, p. 102).

a

b

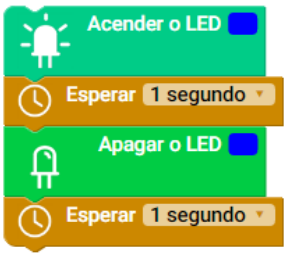

Figura 9:Exemplos de programas utilizando apenas a estrutura sequencial[28].

Ao final dos três encontros utilizados para se trabalhar a *estrutura de programação sequencial*, pôde-se observar, por meio dos programas construídos pelas crianças em resposta aos desafios propostos, que, das sete crianças presentes inicialmente na oficina, três do quarto ano e uma do terceiro ano, embora tendo apresentado alguma dificuldade inicial, acabaram por demonstrar um entendimento claro da relação existente entre a sequência de comandos utilizados no programa e o comportamento observado nos dispositivos presentes na caixinha mágica. Esse fato pôde ser constatado não apenas por meio dos programas construídos por essas crianças em resposta imediata ao enunciado das tarefas, mas, também, pelo reconhecimento e correção de erros de programação realizados quando o resultado observado na caixinha mágica

[28] Cabe mencionar aqui que a sequência de comandos presente em um programa do DB4K é executada automaticamente em um loop infinito (correspondendo aos comandos posicionados dentro da função *void loop* dos programas em Wiring).

não correspondia ao comportamento por elas esperado. As demais crianças finalizaram o terceiro encontro ainda apresentando dificuldade em relação a esses entendimentos.

No que tange ao desenvolvimento e prática de habilidades relacionadas ao Pensamento Computacional, essa construção mental da sequência das ações necessárias para a obtenção de um determinado resultado e sua posterior materialização na forma de um programa de computador, bem como a correção dos programas a partir da comparação do código construído com os resultados observados na caixinha mágica, reflete o uso da capacidade de abstração, o entendimento de noções algorítmicas de fluxo de controle e a aptidão para depurar e detectar erros.

No tocante às habilidades cognitivas características do período operatório concreto e sua relação com o aprendizado de programação de computadores, esses resultados também indicam a habilidade de sequenciar eventos, de realizar operações mentalmente e de compreender relações de causa e efeito.

4.2 Estrutura de Repetição Contada

Foram necessários cinco encontros para que se pudesse trabalhar de forma satisfatória o uso da *estrutura de repetição contada* e do bloco *esperar* (correspondente ao comando *delay* da linguagem Wiring[29]).

Juntamente com o aprendizado da *estrutura de repetição contada*, foi trabalhada nesses encontros, de maneira detalhada, a sequência de comandos necessários para fazer um LED piscar (acender o LED, esperar algum tempo, apagar o LED, esperar algum tempo), uma vez que as crianças estavam apresentando dificuldade no tocante a essa tarefa, o que pode ser resultado de uma dificuldade do trato com conceitos mais abstratos, como a noção de tempo.

Os exercícios realizados com o objetivo de ajudar a vencer essa dificuldade contribuíram para evidenciar o potencial do aprendizado de programação com robótica no desenvolvimento da capacidade da depuração e correção de erros. Um exemplo disso pôde ser constatado quando um aluno, depois de montar o programa presente na Figura 10.a, ao enviá-lo para a caixinha mágica observou que o LED amarelo permanecia aceso, e então exclamou: “Ih, o meu LED amarelo ficou aceso”. Ao ser então questionado sobre o motivo do ocorrido, afirmou: “Porque eu não botei para ele apagar”. Diante da constatação do equívoco, ele corrigiu o programa (Figura 10.b) e enviou novamente para a caixinha, observando agora o seu correto funcionamento. Essa percepção da relação direta existente entre os blocos inseridos no programa e as ações realizadas pela caixinha mágica apontam para um entendimento desse aluno no tocante à percepção de relações de causa e efeito.

Vencidas as dificuldades em relação à sequência de comandos necessária para fazer um LED piscar, ao menos no que diz respeito às quatro crianças que estavam apresentando um melhor entendimento dos conteúdos sendo abordados na oficina, partiu-se para o trabalho com a *estrutura de repetição contada*.

[29] O computador executa as operações a ele solicitadas em uma velocidade muito alta, muitas vezes na casa dos nanosegundos. Desse modo, quando se deseja que o resultado da execução de um determinado comando, como acender LED, Apagar LED, escrever no LCD, etc, seja observável para o olho humano, torna-se necessário, em certas situações, que se estipule um “atraso” no início da execução do próximo comando (ou seja, um tempo de espera). Isso faz-se necessário, por exemplo, para que se possa ver a luz do LED acesa antes que ela apague ou para se ver um texto escrito no LCD antes que ele seja limpo.

Figura 10: Identificação de erros no programa por meio do processo de depuração.

O entendimento do uso dessa estrutura pôde ser observado, por exemplo, em situações como a do desenvolvimento dos programas presentes na Figura 11.

Figura 11: Uso do bloco *repetir*.

Uma vez construído pelos alunos o programa que fazia o LED azul piscar 2 vezes (com o uso do bloco *repetir*), e o LED amarelo piscar 1 vez (Figura 11.a), foi solicitado que fizessem o LED amarelo também piscar 2 vezes. Diante dessa solicitação, um dos alunos, imediatamente, pegou um novo bloco *repetir*, encaixou-o embaixo do bloco *repetir* já presente no programa, e movimentou, para dentro do novo bloco, os blocos utilizados para fazer o LED amarelo piscar (Figura 11.b)

Os exercícios com a *estrutura de repetição contada* também possibilitaram constatar que algumas crianças já estavam, de fato, conseguindo projetar mentalmente um programa de computador a partir da observação de eventos do mundo real, ou seja, construir mentalmente um algoritmo: uma sequência de ações que representasse um evento observado.

Essa habilidade pôde ser verificada, por exemplo, quando, em um exercício no qual as crianças deveriam construir um programa após observarem a caixinha mágica piscando o LED amarelo 3 vezes, o LED vermelho 5 vezes e, por fim, o Motor DC girando por 5 segundos e depois parando, um dos alunos perguntou: "Pode usar o bloco *repetir*?" Ou seja, o questionamento deste aluno indica que ele traduziu as ações observadas em uma sequência de comandos e ainda visualizou duas formas de organizar esses comandos: por meio de uma estrutura puramente sequencial ou com o uso de uma *estrutura de repetição contada*, o que indica, além do uso da capacidade de abstração, o pensamento iterativo e a habilidade deste aluno de trabalhar simultaneamente com mais de um ponto de vista sobre uma mesma ideia, uma habilidade característica do período operatório concreto.

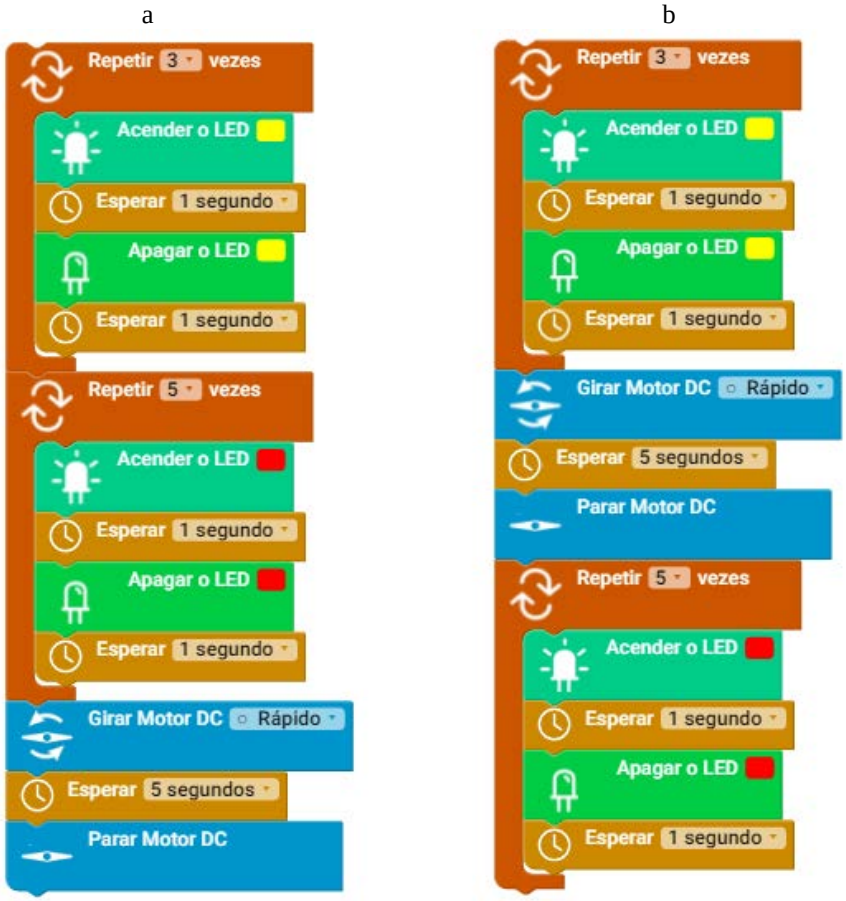

Figura 12: Reaproveitamento de código e noções de fluxo de controle.

Outras habilidades observadas durante as aulas nas quais trabalhou-se a *estrutura de repetição contada* foram a capacidade de reaproveitar código e o entendimento de fluxos de controle, explicitadas, entre outras ocasiões, quando, ao observar a caixinha apresentando as mesmas ações do exercício anteriormente descrito, porém, com uma mudança na ordem dos eventos, com o motor DC girando entre as sequências de piscadas dos LEDS amarelo e vermelho, e não após estas, um dos alunos, habilmente, reorganizou os blocos do programa já construído (Figura 12.a), reposicionando os comandos responsáveis por fazer o motor girar por

cinco segundos entre as duas estruturas de repetição responsáveis por fazer os LEDs piscarem (Figura 12.b), ao invés de descartar o programa anteriormente desenvolvido e construir, do princípio, um programa inteiramente novo. No que se refere às habilidades cognitivas características do período operatório concreto e sua relação com o aprendizado de programação de computadores, este resultado indica também o uso da reversibilidade e da capacidade de distinguir o todo e suas partes.

4.3 Estrutura de Repetição Condicional

Foram necessários cinco encontros para que se pudesse trabalhar de forma satisfatória o uso da *estrutura de repetição condicional* (em conjunto com as estruturas anteriormente trabalhadas). Cabe mencionar que uma das crianças que vinha apresentando um aproveitamento bastante satisfatório dos conteúdos desistiu da oficina durante o aprendizado da *estrutura de repetição contada*, tendo sido esta a única desistência da oficina.

O entendimento inicial do funcionamento da *estrutura de repetição condicional*, assim como no caso das estruturas trabalhadas anteriormente, foi realizado por meio de atividades baseadas na técnica de computação desplugada. Vencida esta etapa, foi proposto um primeiro exercício de programação com o ambiente em blocos que consistia em as crianças observarem um vídeo que mostrava a caixinha mágica mudando de comportamento quando uma lanterna iluminava o seu sensor de luz (o que sugeria o uso da *estrutura de repetição condicional*) e, então, desenvolver o programa que fizesse a caixinha funcionar conforme o observado.

Como resultado pôde-se perceber que as crianças apresentaram bastante dificuldade em determinar a lógica do funcionamento observado. Essa dificuldade aparentemente adivinha do fato de que, dependendo do tempo em que a lanterna permanecia iluminando o sensor de luz, a caixinha apresentava diferentes números de execuções das ações internas ao laço de repetição, o que diferia do resultado "constante" observado quando do uso da *estrutura de repetição contada*. Assim, para chegar ao programa esperado (Figura 13), ou a alguma variante dele, todas as crianças precisaram de uma permanente orientação realizada por meio de questionamentos lançados pelo professor.

Para facilitar o processo, foi feita uma sugestão inicial, de que construíssem o programa em duas partes. Primeiro a parte responsável pelo funcionamento observado quando o sensor de luz não estava sendo iluminado e, depois, a parte correspondente às ações realizadas pela caixinha enquanto a lanterna estava iluminando o sensor. A dinâmica utilizada nesta atividade, ou seja, da divisão do problema em partes menores, mostra que é possível realizar-se, mesmo no desenvolvimento de programas muito simples, um trabalho inicial acerca da "decomposição de problemas", uma habilidade também característica do Pensamento Computacional.

Após a realização de uma série de atividades nas quais as crianças tinham como objetivo construir programas a partir da observação do funcionamento dos materiais de robótica que faziam uso de sensores (estando incluídos nestes materiais, a caixinha mágica, o peixe-robô de garrafa pet, o cachorro-robô de garrafa pet e o morcego-robô de garrafa pet (ver seção 3.2.3)), as 3 crianças com melhor aproveitamento venceram as dificuldades iniciais apresentadas em relação ao uso da lógica condicional e passaram a demonstrar a capacidade de construir programas que fizessem uso de mais de um laço de repetição condicional.

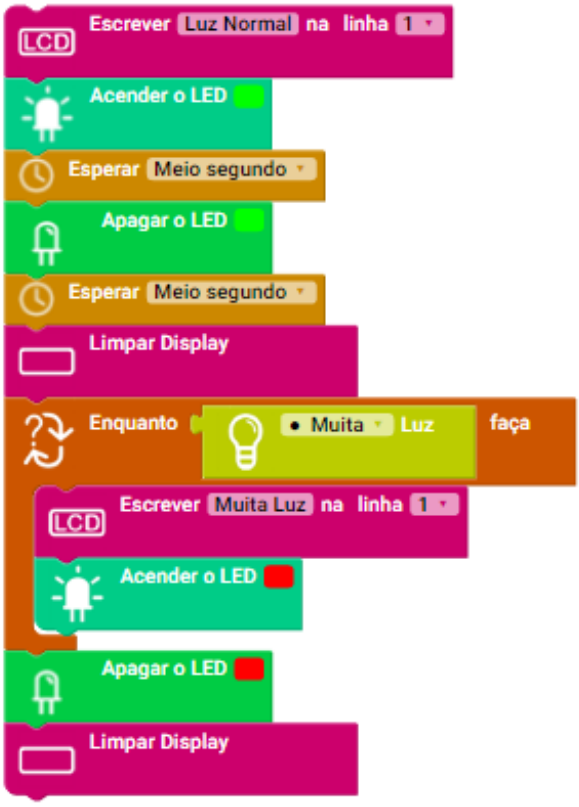

Figura 13: Programa a ser construído na primeira atividade de programação com a *estrutura de repetição condicional*.

No DB4K, o fator condicionante do bloco *enquanto* é sempre um sensor, ou seja, a leitura de um fator ambiental, como a quantidade de luz, o valor da temperatura ou a proximidade de um objeto. Assim sendo, o entendimento do uso da lógica condicional ficou evidenciado no momento em que algumas crianças passaram a construir programas com mais de um laço de repetição condicional a partir da observação do comportamento de materiais de robótica que faziam uso de mais de um sensor.

Pode-se citar como exemplo da demonstração dessa habilidade, o exercício no qual, ao observar o carrinho robótico (ver seção 3.2.3) desviando de obstáculos (por meio do uso de um sensor de distância) e movimentando-se para trás quando um foco de luz incidia sobre ele (por meio do uso de um sensor de luz), um dos alunos construiu o programa correspondente utilizando, por iniciativa própria e de forma adequada, os dois blocos *enquanto* necessários e seus respectivos fatores condicionantes, além de posicionar os demais comandos nos locais apropriados. Seu único erro foi esquecer de selecionar o operador de comparação correto para o sensor de distância (ou seja, não ter trocado o operador padrão “>” por “<” (Figura 14).

As três crianças que estavam apresentando maior dificuldade em absorver os conteúdos trabalhados finalizavam as tarefas propostas nestes encontros apenas com o auxílio do professor ou a partir da observação dos programas dos colegas.

No último encontro (décima quarta aula), como atividade de encerramento da oficina, foi solicitado às crianças que fizessem o desenho de um robô que utilizasse alguns dos atuadores (motores, LEDs, displays, entre outros) e sensores (sensor de luz, sensor de distância, entre outros) trabalhados em aula, e explicassem o seu funcionamento.

Figura 14: Programa construído a partir da observação do funcionamento do carrinho robótico.

Os três alunos que vinham apresentando maior dificuldade no uso das estruturas de repetição contada e condicional, acabaram por desenvolver projetos de robôs que faziam uso apenas de atuadores. Por outro lado, os três alunos com melhor aproveitamento desenharam robôs que utilizavam sensores, ou seja, que possuíam um funcionamento baseado no uso da lógica condicional e explicaram, com sucesso, o porquê da presença de cada um dos dispositivos utilizados nos seus projetos e de que modo eles determinavam ou refletiam o funcionamento de seus robôs. Como exemplo podemos citar os projetos presentes na Figura 15.

a b

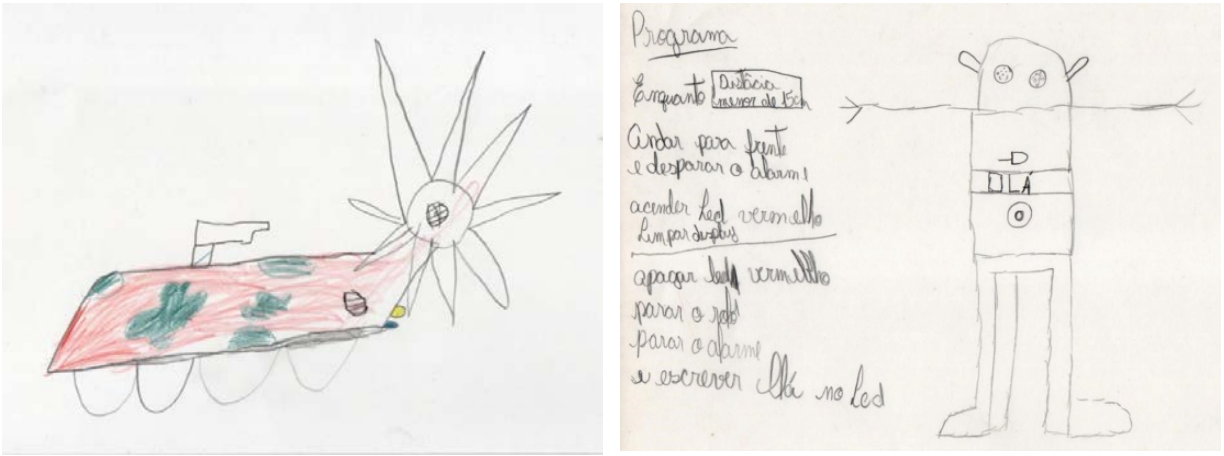

Figura 15: Exemplos de projetos de robôs desenvolvidos pelos alunos no último dia de aula.

A Figura 15.a apresenta um “robô de combate” com quatro rodas, uma serra circular na parte dianteira (movida por um motor DC), um sensor de distância acoplado à serra, um canhão na parte superior e um sensor de luz na lateral. De acordo com a explicação dada pelo aluno projetista, a serra iria girar sempre que um outro robô se aproximasse, o que seria percebido pelo sensor de distância. Já o canhão seria ativado toda vez que uma luz forte (o farol de um robô adversário, por exemplo) incidisse sobre o robô.

No projeto da Figura 15.b, o aluno, além de projetar o robô, que fazia uso de um sensor de distância para disparar seu deslocamento e soar um alarme, construiu, por iniciativa própria, ao lado do desenho, o algoritmo que descrevia o funcionamento do seu robô, e depois montou, também por iniciativa própria, o programa correspondente no ambiente de programação visual em blocos.

A *estrutura de decisão*, inicialmente prevista para ser abordada na oficina, não foi trabalhada, uma vez que o ritmo de aprendizado das crianças exigiu um tempo maior do que o previsto no desenvolvimento das atividades. No entanto, a habilidade do Pensamento Computacional específica a ser exercitada através desta estrutura seria o uso da lógica condicional, já trabalhada por meio do aprendizado da *estrutura de repetição condicional*.

5 Conclusões

Os resultados apresentados indicam a viabilidade de se trabalhar, com crianças na faixa etária e pertencentes a contextos socioeconômicos semelhantes aos dos sujeitos participantes deste estudo, a partir do aprendizado de estruturas básicas de programação por meio da Robótica Educacional, fazendo uso exclusivamente de tecnologia livre e materiais recicláveis e de baixo custo, as cinco habilidades do Pensamento Computacional que se havia suposto serem possíveis de se trabalhar, sendo elas: a capacidade de abstração (mais especificamente a abstração empírica), compreensão de fluxos de controle, depuração e detecção sistemática de erros, pensamento iterativo, uso da lógica condicional e decomposição estruturada de problemas.

A demonstração dessas habilidades, em conjunto com outras competências manifestadas pelas crianças durante a realização das atividades propostas na oficina, como a capacidade de reaproveitar código, a percepção da ideia de processamento, a construção mental das ações necessárias para a realização de uma determinada tarefa, o entendimento de que os mesmos comandos organizados de diferentes maneiras podem levar a diferentes resultados e de que um mesmo problema pode apresentar diferentes soluções, ajudam a reforçar as hipóteses levantadas na seção 2.5, acerca da possível existência de uma relação direta entre determinadas características cognitivas de crianças operatórias concretas e a realização de algumas atividades relacionadas à programação de computadores.

Das sete crianças participantes do estudo, três finalizaram a oficina demonstrando ser capazes de desenvolver atividades que demandassem as habilidades acima mencionadas, três apresentaram dificuldades nesse sentido, e uma desistiu da oficina, tendo, no entanto, demonstrado competência para desenvolver todas as atividades propostas nas aulas em que esteve presente. A maior dificuldade encontrada pelas crianças refere-se a questões relacionadas à capacidade de abstração, como a utilização da noção de tempo e de laços de repetição, e a construção mental (ainda que realizada a partir de referenciais concretos) da sequência de ações necessárias para a obtenção de um determinado resultado e sua posterior materialização na forma de um programa de computador.

A hipótese aqui levantada é a de que essa dificuldade seja decorrente de questões relativas à maturidade cognitiva, uma vez que todos os sujeitos observados se encontravam dentro da faixa etária correspondente ao estágio operatório concreto, no qual a capacidade de abstração ainda está em desenvolvimento. Além disso, uma vez que a maturidade cognitiva é dependente de fatores biológicos, educacionais e sociais, é possível que, em relação a certos aspectos, algumas das crianças observadas apresentassem uma maturidade pré-operatória, período no qual ainda não se consegue, de modo geral, realizar operações mentalmente (ver seção 2.5).

5.1 Principais Contribuições

A abstração é o processo mais importante do Pensamento Computacional (Wing, 2011), e a programação de computadores uma importante ferramenta de apoio ao desenvolvimento e exercício de uma série de habilidades desta forma de pensamento, bem como uma competência fundamental da Ciência da Computação (Grover & Pea, 2013). De acordo com Dijkstra (1972,

p. 864) “a exploração efetiva dos seus poderes de abstração deve ser considerada uma das atividades mais vitais de um programador competente”.

Há hoje um crescente movimento no sentido de se introduzir o ensino de programação já nos primeiros anos do Ensino Fundamental e, até mesmo, a partir da Educação Infantil. Torna-se assim imprescindível a realização de pesquisas acerca da relação existente entre a capacidade de abstração de crianças nessa faixa etária e as habilidades necessárias ao aprendizado de conteúdos relacionados à programação de computadores, de modo que esses conteúdos sejam abordados no momento mais propício e do modo mais adequado. Uma importante contribuição deste trabalho é chamar a atenção para este fato e contribuir com a investigação de hipóteses acerca desse tema.

Outra contribuição significativa é o kit DB4K, cujo conjunto de materiais pedagógicos demonstrou-se adequado ao desenvolvimento e exercício de algumas habilidades cognitivas do Pensamento Computacional por meio da programação de computadores. E, por ser baseado em Tecnologia Livre associada a materiais recicláveis e de baixo custo, o kit apresenta-se economicamente mais acessível do que as soluções proprietárias atualmente disponíveis no mercado, podendo contribuir assim para uma maior democratização do acesso ao aprendizado de programação de computadores nos primeiros anos do Ensino Fundamental.

5.2 Trabalhos Futuros

Como possíveis trabalhos futuros, derivados desta pesquisa, destacam-se: a realização de um estudo mais amplo acerca das teorias de desenvolvimento cognitivo, bem como das habilidades do Pensamento Computacional, de maneira a se produzir investigações mais detalhadas a respeito das hipóteses levantadas por essa pesquisa; o desenvolvimento de novas oficinas para a realização de investigações mais abrangentes acerca dos aspectos explorados por este trabalho e a realização de estudos do uso do kit DB4K envolvendo crianças com dificuldades cognitivas.

6 Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001, com o financiamento parcial da RNP (Rede Nacional de Ensino e Pesquisa) através do Edital para os Grupos de Trabalho (GTs) da referida organização para o período 2014-2015, e da FAPERJ (Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro)

Referências

Aguilar, L. J. (2008). *Fundamentos de Programação: Algoritmos, estruturas de dados e objetos.* (3 ed.). Porto Alegre: AMGH.

Alves, R. M., Sampaio, F. F. & Elia, M.F. (2014) DuinoBlocks: Desenho e Implementação de um Ambiente de Programação Visual para Robótica Educacional. *Revista Brasileira de Informática na Educação,* 22(3), 126-140. doi: http://dx.doi.org/10.5753/rbie.2014.22.03.126 [GS Search]

Arnold, K., James, G., & David, H. (2009). *A linguagem de programação Java* (4 ed.). Porto Alegre: Bookman.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? *Acm Inroads, 2*(1), 48-54. doi: 10.1145/1929887.1929905

Bell, T., Alexander, J., Isaac, F., & Grimley, M. (2009). Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology*, 13(1), 20-29. [GS Search]

Bers, M. U., Flannery, L., Kazakoff, E., & Crouser , R. J. (2010). *A Curriculum Unit on Programming and Robotics*. DevTech Research Group. Tufts University, Medford. [GS Search]

Bhattacharya, J. (2016). Rudiments of Computer Science (3 ed., Vol. 1). Calcutá: Academic Publishers.

Burnett, M. M. (1999). Visual programming. Em *Wiley Encyclopedia of Electrical and Electronics Engineering*. New York: John Wiley & Sons, Inc. doi: 10.1002/047134608X.W1707

Butcher, S. E. (2006). Narrative as a Teaching Strategy. *Journal of Correctional Education*,57, 195–208. Disponível em https://www.jstor.org/stable/23282752

Campos, F. R. (2011). *Currículo, Tecnologias e Robótica na Educação Básica (*Tese de Doutorado). PUC. São Paulo. [GS Search]

Chaiken, S., & Ledgerwood, A. (2012). *A theory of heuristic and systematic information processing*. In P. A. M. Van Lange, A. W. Kruglanski, & E. T. Higgins (Eds.), Handbook of theories of social psychology , 246-266. Thousand Oaks, CA, : Sage Publications Ltd.

Chella, M. T. (2016). Ori: Plataforma para Robótica Educacional de Baixo Custo. *Anais da VI Mostra Nacional de Robótica (MNR 2016)*, 778-781. [GS Search]

Cooper, S., Grover, S. Guzdial, M., & Simon, B. (nov de 2014). A Future for Computing Education Research. *Communications of the ACM*, 57 (11), 34-36. doi: 10.1145/2668899 [GS Search]

Costa Jr., A. d., & Guedes, E. B. (2015). Uma Análise Comparativa de Kits para a Robótica Educacional. *Anais do XXXV Congresso da Sociedade Brasileira de Computação*. Recife. Disponível em http://www.lbd.dcc.ufmg.br/colecoes/wei/2015/012.pdf

de França, R. S., & do Amaral, H. J. (2013). Ensino de Computação na Educação Básica no Brasil:Um Mapeamento Sistemático. *Anais do XXXIII Congresso da Sociedade Brasileira de Computação*, 426-431. [Gs Search]

de Miranda, L. C., Sampaio, F. F., & Borges, J. d. (Fevereiro de 2011). RoboFácil: Especificação e Implementação de um Kit de Robótica para a Realidade Educacional Brasileira. *Revista Brasileira de Informática na Educação*, 18(3), 46-58. doi: http://dx.doi.org/10.5753/rbie.2010.18.03.46. [GS Search]

Dijkstra, E. W. (out de 1972). The humble programmer. *Communications of the ACM, 15*(10), 859-866. doi: 10.1145/355604.361591

Dimes, T., & Rosa, R.S. (2016). *Programação em C# Para Iniciantes*. Babelcube Inc.

Elleström, L. (2014). Two Types of Media Transformation. In L. Elleström (Org.), *Media Transformation: The Transfer of Media Characteristics Among Media* (p. 11–35). London: Palgrave Macmillan. doi: 10.1057/9781137474254-2

Fabri Junior, L. A. (2014). *O uso de Arduino na criação de Kit para oficinas de robótica de baixo custo para escolas públicas*. Dissertação (Mestrado em Tecnologia e Inovação). Limeira. [GS Search]

Ferreira, L. A., de Jesus, Â. M., Rufo, M. B., & Santos, F. C. (2016). Se-Robô: Aplicativo para Robótica Educacional de Baixo Custo. *Anais do XXVII Simpósio Brasileiro de Informática na Educação-SBIE*, 1285-1289. doi: http://dx.doi.org/10.5753/cbie.sbie.2016.1285 [GS Search]

Filho, C.F. (2007). *História da computação: O Caminho do Pensamento e da Tecnologia.* Porto Alegre: EDIPUCRS.

Furtado, O., Bock, A. M., & Teixeira, M. d. (2001). *Psicologias: uma introdução ao estudo de psicologia* (13 ed.). São Paulo: Saraiva.

Garneli, V., Giannakos, M. N., & Chorianopoulos, K. (2015, Março). Computing education in K-12 schools: A review of the literature. In *Global Engineering Education Conference* (EDUCON), 2015 IEEE, 543-551. IEEE. doi: 10.1109/EDUCON.2015.7096023

Grover, S., & Pea, R. (Jan de 2013). Computational Thinking in K-12: A Review of the State of the Field. *Educational Researcher, 42*(1), 38–43. doi: 10.3102/0013189X12463051 [GS Search]

Gupta, P., Agarwal, V., & Varshney, M. (2007). *Data Structure Using 'C'* (1 ed.) Boston: Firewall Media.

Hemmendinger, D. (Jun de 2010). A plea for modesty. *Acm Inroads, 12*, 4-7. doi: 10.1145/1805724.1805725

Kamii, C. (1992). *A criança e o número: implicações educacionais da teoria de Piaget para a atuação junto a escolares de 4 a 6 anos* (36 ed.). Campinas: PAPIRUS.

Kramer, J. (abr de 2007). Is abstraction the key to computing? *Communications of the ACM, 50*(4), 36–42. doi: 10.1145/1232743.1232745

Lister, R. (2011). Concrete and other neo-Piagetian forms of reasoning in the novice programmer. *Proceedings of the Thirteenth Australasian Computing Education Conference* (pp. 9-18). Perth: Australian Computer Society. [GS Search]

Lye, S.Y., & Koh, J.H.L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12. *Computers in Human Behavior.* v.41,51-61. doi: 10.1016/j.chb.2014.09.012 [GS Search]

Maiers, W., Bayer, B., Esgalhado, B.D., Jorna, R., Schraube, E. (Eds.). (1999). *Challenges to Theoretical Psychology*. North York: Captus Press.

Medeiros Filho, D. A., & Gonçalves, P. C. (2008). Robótica educacional de baixo custo: Uma realidade para as escolas brasileiras. *Anais do XXVIII Congresso da SBC-XIV Workshop de Informatica na Escola*, 264-273. [GS Search]

Meirinhos, M., & Osório, A. (2010). O estudo de caso como estratégia de investigação em educação. *EduSer - Revista de Educação, 2*(2), 49-65. [GS Search]

Ministério da Educação. (2017). *Base Nacional Comum Curricular*. Disponível em http://basenacionalcomum.mec.gov.br/

Moreira, M. A. (1999). *Teorias de aprendizagem.* São Paulo: EPU.

Papert, S. (1993). *Mindstorms: Children, Computers and Powerful Ideas* (2 ed.). New York: Basic Books.

Piaget, J. (1974). *To understand is to invent.* New York: Grossman Publishers.

Queiroz, R. L. (2017) *DuinoBlocks4Kids: utilizando Tecnologia Livre e materiais de baixo custo para o exercício do Pensamentos Computacional no Ensino Fundamental I por meio do aprendizado de programação aliado à Robótica Educacional* (Dissertação de Mestrado). Universidade Federal do Rio de Janeiro. Rio de Janeiro. Disponível em http://ginape.nce.ufrj.br/LIVRE/paginas/dissertacoes/d_2017_rubens_queiroz.pdf

Queiroz, R.L., Sampaio, F.F., & Santos, M.P. (2017). *DuinoBlocks4Kids: Utilizando Tecnologia Livre e Materiais de Baixo Custo para o Exercício do Pensamento Computacional no Ensino Fundamental I por meio do Aprendizado de Programação Aliado à Robótica Educacional*. Anais dos Workshops do Congresso Brasileiro de Informática na Educação (CBIE-2017), Recife, 2017. doi: http://dx.doi.org/10.5753/cbie.wcbie.2017.25

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Evelyn, E., Brennan, K., . . . Silverman, B. (2009). Scratch: programming for all. *Communications of the ACM, 52*(11), 60-67. doi: 10.1145/1592761.1592779

Sasahara , L. R., & da Cruz, S. M. (2007). Hajime–Uma nova abordagem em robótica educacional. Anais do Workshop de Informática na Escola, (459-461). [GS Search]

Sasikumar, M., Shikhare, D. & Prakash, R.P. (2014). *Introduction to Parallel Processing*. (2 ed.) Deli: PHI Learning.

Souza, N. M. (2014). Reflexões sobre a teoria piagetiana: o estágio operatório concreto. *Cadernos de Educação: Ensino e Sociedade, 1*(1), 134-150. [GS Search]

Sullivan, A., & Bers, M. U. (2016). Robotics in the early childhood classroom: learning outcomes from an 8-week robotics curriculum in pre-kindergarten through second grade. *International Journal of Technology and Design Education, 26*(1), 3–20. [GS Search]

Szurmak, J.; Thuna, M. (2013). *Tell me a story: The use of narrative as a tool for instruction*. Proceedings of ACRL, Indianapolis, IN, 546–552. [GS Search]

Terra, M. R. (2010). *O desenvolvimento humano na teoria de Piaget*. Disponível em https://www.unicamp.br/iel/site/alunos/publicacoes/textos/d00005.htm

Whitehouse. (2016). Office of the Press Secretary. *President Obama Announces Computer Science For All Initiative*. Disponível em https://obamawhitehouse.archives.gov/the-press-office/2016/01/30/fact-sheet-president-obama-announces-computer-science-all-initiative-0

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35. doi: 10.1145/1118178.1118215

Wing, J. M. (2011). Computational Thinking-What and Why? *The Link Magazine*, 20-23. Disponível em http://www.cs.cmu.edu/sites/default/files/11-399_The_Link_Newsletter3.pdf

Yin, R. K. (2001). *Estudo de Caso: Planejamento e Métodos* (2 ed.). Porto Alegre: Bookman.