O ENSINO DE PROGRAMAÇÃO DE COMPUTADORES EM UM AMBIENTE CRIATIVO E MOTIVADOR

TEACHING COMPUTER PROGRAMMING WITH A CREATIVE AND MOTIVATING ENVIRONMENT

Elena Mariele Bini[1]

André Koscianski[2]

[1] Docente da Faculdade Guairacá, Colégio Estadual Francisco Carneiro Martins, elena_bini@seed.pr.gov.br

[2] Docente do Programa de Pós-Graduação em Ensino de Ciência e Tecnologia da UTFPR, koscianski@utfpr.edu.br

Resumo

O processo de ensino e aprendizagem dos conceitos iniciais da programação de computadores é rico em desafios e dificuldades que são explorados de diversas formas pela literatura. Este artigo relata uma experiência de utilização do ambiente de programação Scratch para o ensino dos conceitos iniciais da programação de computadores para alunos de um curso Técnico em Informática. Além do ambiente Scratch, um roteiro de atividades que estimulam o desenvolvimento de habilidades de resolução de problemas também foi elaborado e testado. Os resultados dessa experiência foram favoráveis quanto ao aspecto motivacional proporcionado pela ferramenta e, sobretudo, confirmaram a importância de dar maior prioridade à habilidade de resolver problemas do que exercitar escrita de código ou pseudo-código.

Palavras-chave: programação de computadores, processo de ensino e aprendizagem, resolução de problemas, ambiente criativo e motivador

Abstract

Computer programming present various challenges and difficulties to both teachers and students. The subject is treated under diverse manners in the literature. This article reports an experiment with the use of the Scratch programming environment, to teach initial concepts computers programming, to students of a technology course. In addition to the Scratch environment, a series of activities that emphasize the development of problem solving skills has been devised and tested. The results of the study were positive with respect to the motivational factor of the tool and, more important, confirmed the importance

of according greater priority to the development of problem solution skills than exercising code and pseudo-code writing.

Keywords: programming of computers, teaching and learning process, problem solving, creative and motivating environment.

1 INTRODUÇÃO

Um dos pilares da atuação profissional em informática é a programação de computadores. Uma parte importante das grades curriculares de cursos de formação a nível médio ou superior dedica disciplinas e conteúdos a essa atividade.

O processo de aprendizado dos conceitos iniciais da programação de computadores é complexo e marcado pela presença de inúmeras dificuldades, que são detectadas em pesquisas no mundo todo. Alguns dos pontos mais destacados são a baixa capacidade em resolução de problemas aliada a equívocos na formulação de modelos mentais adequados, a falta de motivação para executar tarefas, a dificuldade para tratar abstração e ferramentas e linguagens não adaptadas pedagogicamente.

A capacidade de resolução de problemas é uma habilidade fundamental a ser estimulada nos estudantes, porém algumas vezes deixada de lado. Em certos casos é dada ênfase ao ensino de determinada linguagem de programação ou ao uso de pseudo-código, ficando em segundo plano um tratamento adequado para estimular os alunos a resolver problemas com iniciativa e criatividade. Nesses casos os alunos novatos em programação podem ser levados a acreditar que entender uma linguagem específica ou empregar pseudo-código os ajude a resolver problemas. Infelizmente, quando solicitados a desenvolver uma solução similar a outra anterior, não se mostram capazes de utilizar conhecimento prévio; na verdade, não desenvolveram a habilidade necessária e que não está contida em representações textuais ou pictográficas.

Este trabalho relata um teste empregando duas estratégias: o estímulo à resolução de problemas em um contexto criativo e motivador; e o uso do ambiente de programação Scratch para criar esse ambiente, em substituição à abordagem mais frequente, baseada em textos e problemas de caráter abstrato.

1.2 VISÃO GERAL DO ENSINO DA PROGRAMAÇÃO DE COMPUTADORES

Um programa de computador consiste de um conjunto de instruções que a máquina deve executar para cumprir uma tarefa específica. Elaborar um programa consiste em conceber a solução dessa tarefa por meio de operações que a máquina possa realizar e, em seguida, materializar essa informação usando uma linguagem de programação. O raciocínio embutido na sequência de instruções – seja essa sequência representada em uma linguagem de programação ou qualquer outra representação – é conhecido como algoritmo.

Para exemplificar, o código apresentado no quadro 1 contem um trecho de programa para calcular a média das notas de 40 alunos:

Quadro 1: Exemplo de trecho de código na linguagem Pascal

```
i     := 0;
soma  := 0;

while (i < 40) do begin

      writeln ('Escreva uma nota: ');
      readln  (nota);

      soma := soma + nota;
      i    := i + 1;
end;

media := soma / 4;

writeln (media);
```

O exemplo foi escrito na linguagem de programação Pascal, desenvolvida por Niklaus Wirth precisamente como ferramenta de ensino. Percebe-se a presença de vários sinais de pontuação, que obedecem regras estritas de sintaxe. Esses sinais simplificavam a construção de compiladores (Pascal foi criado em 1970), mas dificultam a tarefa de programadores. O simples esquecimento desses caracteres, ou seu posicionamento em locais inadequados, são capazes de gerar um erro de programação. Para alunos novatos isso representa uma preocupação adicional, que pode desviar a atenção da questão fundamental de criar uma solução a um problema. Uma solução plausível seria substituir uma linguagem verdadeira por pseudo-código. Infelizmente, na grande maioria das vezes isto não é aplicado de maneira adequada, limitando-se à tradução em português de uma linguagem de programação, em lugar de suporte à abordagens de solução de problemas como refinamentos sucessivos (Wirth, 1971).

Outra ferramenta de apoio à concepção de programas é o fluxograma. Trata-se de uma representação gráfica, ilustrada na Figura 1.

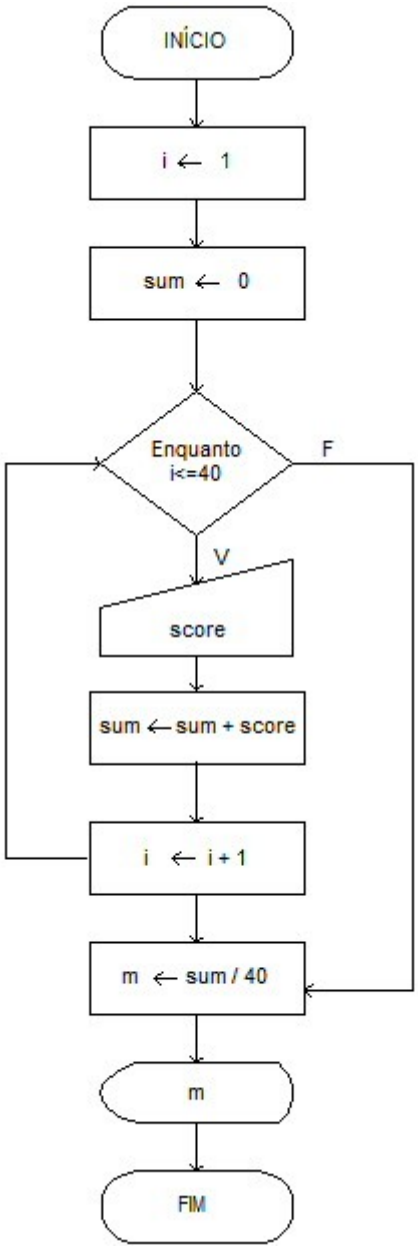

Figura 1: Exemplo de fluxograma

Se introduzir os conceitos iniciais da programação através de linguagens de programação complexas pode acarretar uma sobrecarga de dificuldade, aplicar fluxogramas ou pseudocódigo acaba por também contribuir para a permanência da abstração presente no processo de ensino-aprendizagem. Além disso, caso o aluno não tenha formulado um modelo mental adequado do funcionamento do computador, será inviável a aplicação de tais ferramentas pois ele não compreenderá como se dará a execução das instruções.

2 Visão Geral da Literatura

As dificuldades apresentadas pelos alunos novatos em programação de computadores vêm sendo foco de pesquisas no Brasil e no mundo há anos. Inúmeras ferramentas e metodologias foram propostas tendo como objetivo amenizar tais dificuldades e tornar o processo de ensino e aprendizagem da programação de computadores menos complexo.

No caso do ensino no Brasil, dados mostram que durante algum tempo o foco dos estudos esteve na dificuldade de assimilação das abstrações presentes no processo de aprendizagem (Pereira Júnior e Rapkjewicz, 2004). Uma razão é a falta de comprovação prática com feedback imediato e fiel, recebido pela implementação direta no computador. Assim, houve o desenvolvimento de inúmeras ferramentas com o objetivo de amenizar tais abstrações. Todavia, tais ferramentas não foram suficientes.

O WEI, Workshop on Education in Computer Science, é um dos eventos mais importantes da área no Brasil. Ao analisar o foco das publicações do WEI 2005 ao WEI 2008, percebe-se um aumento de publicações relacionadas com metodologias de ensino, sejam elas aliadas a ferramentas computacionais ou não. O aumento de publicações sobre metodologia de ensino de programação não é uma tendência restrita ao Brasil e aparece em outros países (Ala-Mutka 2004, Cooper 2003, Gomes 2008, Havenga 2008, Jenkins 2002).

Geralmente, essas propostas focam o estímulo à resolução de problemas e a importância da construção de modelos mentais adequados sobre o computador e seu funcionamento. As questões básicas que cercam o problema podem ser retraçadas a pesquisas sobre modelos mentais, como em Ben-Ari (1998) e os trabalhos fundamentais de Wirth (1971) e Polya (1978) na resolução de problemas.

2.1 A FERRAMENTA SCRATCH

Com o objetivo de trazer uma contribuição ao assunto, buscou-se organizar uma abordagem de ensino. Alguns pontos-chaves foram identificados como prováveis fatores capazes de amenizar as dificuldades apresentadas pelos alunos novatos em programação e que deveriam nortear a proposta:

- enfatizar aspectos pedagógicos: optar por linguagens e ambientes de programação com menor carga sintática, como Python, em lugar de escolher linguagens de ampla aceitação na indústria como Java e C (McIver e Conway, 1996);
- estimular a capacidade de resolução de problemas (Falkner e Palmer 2009), especialmente os computacionais: os alunos devem ser motivados a compreender o problema e planejar a sua solução, além de retornar a solução elaborada e testada, procurando melhorá-la (Wirth 1971, Polya 1978);

- a resolução de problemas deve apresentar sentido aos alunos para assim tornar as aulas estimulantes: os problemas a serem propostos devem estar aliados aos interesses do educando. Ao agir por seu próprio interesse o aluno sente-se motivado a buscar a resolução dos problemas, inclusive passando mais tempo programando (Maloney et al. 2008);
- diminuição da abstração: utilização de softwares, especialmente de visualização, podem favorecer o entendimento das estruturas básicas da programação de computadores (Naps et al 2003; Moskal, D. Lurie e S. Cooper, 2000).

Buscou-se uma ferramenta computacional que estivesse em consonância com os pontos-chaves citados e baseada em critérios documentados (McIver e Conway, 1996; Parker et al, 2000). Outras características são apresentadas no quadro 2.

Quadro 2: Alguns critérios para seleção de linguagens para uso pedagógico na pesquisa

Critério	Descrição
Forma de distribuição	software livre
Paradigma de programação suportado	permitir programação estruturada
Suporte on-line de comunidade ativa	possuir comunidades ativas de desenvolvimento e pesquisa

Após pesquisas o ambiente de programação Scratch foi selecionado e um teste foi realizado. Tal ambiente de programação satisfaz as necessidades elencadas como fundamentais para a metodologia. O ambiente de programação Scratch foi lançado em 2007 como uma ferramenta voltada ao ensino de programação à novatos. A ferramenta não é a primeira desenvolvida com esse objetivo, mas apoia-se em pressupostos pedagógicos muito sólidos. A ferramenta foi desenvolvida no renomado Massachussets Institute of Technology e aplica idéias do mundialmente conhecido software LOGO, também proposto por Papert no MIT (Maloney, 2008). A programação dispensa a digitação de código e se baseia em arrastar e soltar blocos de comandos.

A figura 2 apresenta a tela principal do ambiente de programação. Scratch oferece uma interface agradável com blocos de comandos organizados em categorias (lado esquerdo) e a visualização das ações (lado direito). Possibilita aos alunos integrar, de forma simples recursos como sons, imagens e vídeos. Tais características são capazes de levar ao desenvolvimento de programas que ressoam com os interesses dos alunos, geralmente no formato de jogos e animações gráficas.

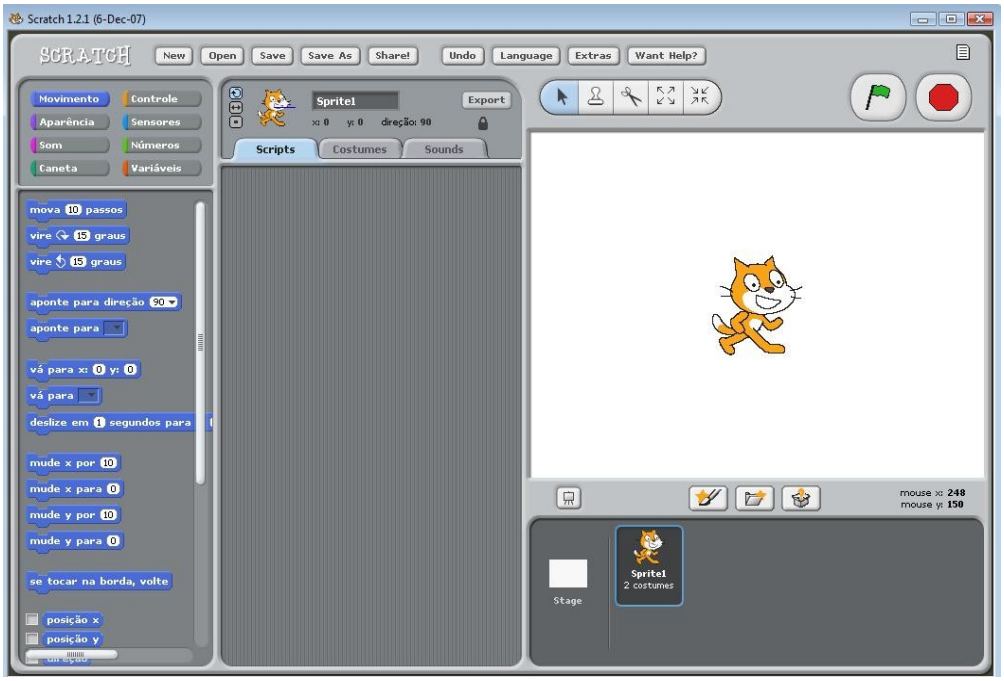

Figura 2: Tela principal do ambiente de programação Scratch

O teste da ferramenta Scratch aconteceu com 15 alunos do curso Técnico em Informática, modalidade integrado. A turma estava matriculada na segunda série do curso e teve o primeiro contato com a programação de computadores nessa disciplina. A figura 3 mostra uma aluna participante utilizando o ambiente.

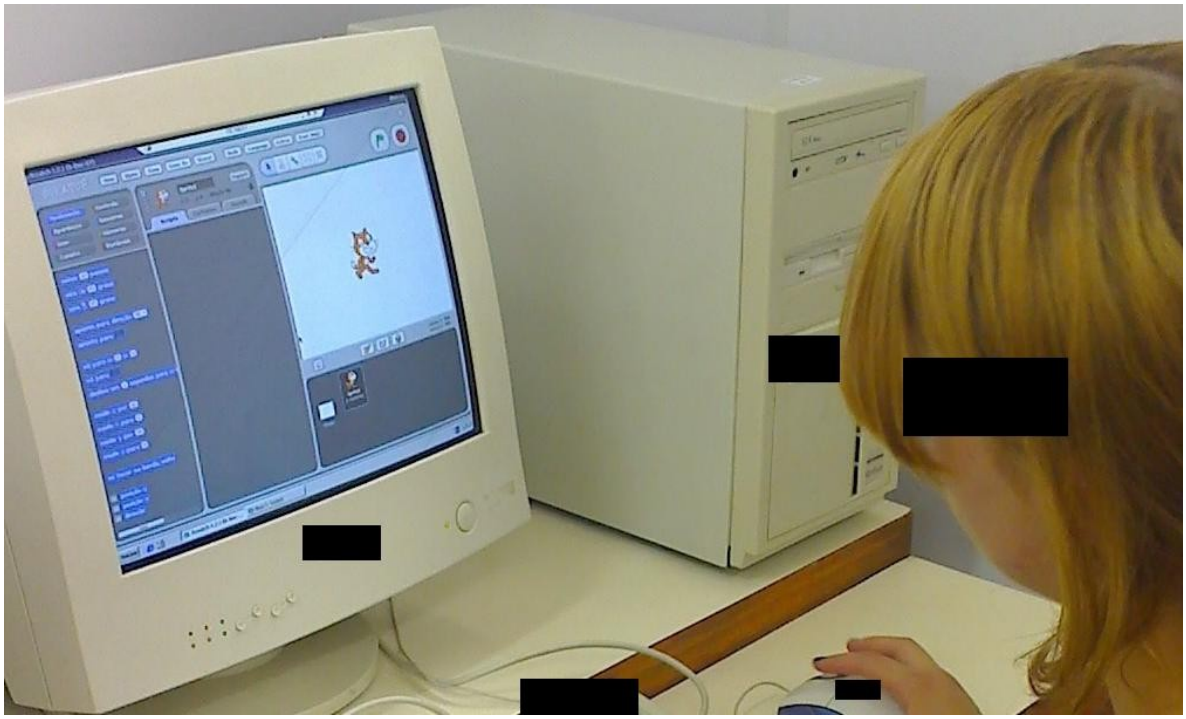

Figura 3: Aluna participando do teste realizado

Os encontros aconteceram em laboratório de informática, em contra-turno, totalizando 15 horas. O objetivo foi verificar o comportamento dos alunos diante da ferramenta, bem como testar alguns exercícios e desafios elaborados para estimular a resolução de problemas.

RESULTADOS E DISCUSSÕES

No momento da realização do teste os alunos estavam no terceiro bimestre letivo. Em função disso já conheciam conceitos básicos, como variáveis, estruturas de seleção e de repetição, comandos de entrada, saída e atribuição. Em um primeiro momento os alunos foram organizados em duplas livres para explorar a ferramenta e produzir programas. Em cada encontro o material da aula era entregue e eles deveriam estudar exemplos e realizar atividades propostas, com acompanhamento e ajuda da professora. Essa organização foi adotada seguindo a estratégia proposta por Papert (2008), onde o aluno exercer comando de seu próprio aprendizado. Percebeu-se que os alunos, mesmo acostumados a escrever algoritmos em pseudo-linguagem, viam-se sem saber como começar a programar um problema já visto anteriormente. Isso acontecia por se verem em um contexto diferente daquele vivido anteriormente em sala de aula e, sobretudo, por não adotarem uma estratégia adequada. Como já ressaltado, não é na escrita de código que se concebe uma solução e, justamente, não existe escrita de código usando Scratch. Uma possibilidade para resolver o problema de concepção é empregar as heurísticas de Polya para solução de problemas. A partir do terceiro encontro, o trabalho de Polya foi apresentado aos alunos e a aplicação de sua heurística incentivada.

Durante a realização do teste foram coletados dados por observação direta e também com entrevistas.

A observação direta permitiu concluir que a utilização do ambiente de programação Scratch tornou o aprendizado mais dinâmico, divertido e motivador. Os alunos demonstraram muito interesse e vontade de continuar a utilizar a ferramenta, mesmo após o encerramento da pesquisa. Eles estavam visivelmente mais motivados para trabalhar em laboratório, desenvolvendo animações computacionais e pequenos jogos, do que em sala de aula escrevendo pseudocódigo para resolver problemas abstratos.

Alguns alunos demonstraram interesse em avançar na complexidade das atividades sugeridas, criando animações mais arrojadas que aquelas previamente solicitadas. Ao mesmo tempo houveram manifestações de inquietação com uma possível avaliação sobre o Scratch. Uma pergunta realizada duas vezes foi:

“Professora, caso eu faça diferente, vou perder nota?”

Somente após serem lembrados que aquele momento era destinado ao exercício da criatividade e da resolução de problemas é que os alunos sentiram-se realmente livres.

Percebeu-se, entretanto, que apenas a utilização da ferramenta não foi suficiente para modificar a atitude em relação às dificuldades apresentadas em sala de aula. Alguns alunos, ao se depararem com a dificuldade de planejamento para a solução do problema, desistiram sem ao menos tentar. Ou, tendo solucionado problema semelhante, não mostraram iniciativa de buscar embasamento na experiência já vivida para a solução de novos problemas. Esses fatos reforçam a idéia de que o cerne das dificuldades apresentadas pelos alunos novatos em programação está na baixa capacidade para resolução de problemas e que concentrar-se na representação de algoritmos ou sua tradução em diferentes meios (como pseudo-código e linguagem) não é suficiente para tratar a questão.

Isso pode ter ocorrido em parte pela não elaboração adequada de modelos mentais e pela aplicação de estratégias de ensino que não consideram aspectos cognitivos do aprendizado.

As entrevistas realizadas ao término do teste indicam que o ensino dos conceitos iniciais da programação de computadores usando uma linguagem interpretada favorece a aprendizagem graças ao feedback fiel e imediato. Isso também exige dos estudantes maior planejamento e/ou correções durante a implementação do programa, além de contribuir para a diminuição das abstrações presentes nesse processo. Alguns alunos mostraram não entender o que era realmente o problema a ser solucionado, em consequência fracassando no planejamento. Nesse caso há uma questão anterior a ser tratada, que é resolver a leitura, interpretação e compreensão do problema e seu contexto. De forma geral os alunos demonstraram concordância e interesse no uso do feedback imediato proporcionado por uma linguagem interpretada, em lugar do uso de pseudo-código.

Confirmou-se que o ambiente de programação Scratch torna a programação fácil por reduzir fortemente a exigência de sintaxe. Sua utilização também é capaz de tornar a programação divertida, já que os alunos podem inserir em seus programas elementos como sons e imagens por eles mesmos produzidos.

CONCLUSÃO

Existem diversos trabalhos que buscam melhorar o processo de ensino e aprendizagem dos conceitos iniciais da programação de computadores. Uma mudança de foco importante consiste em dois pontos: dar prioridade a ferramentas e linguagens mais apropriadas pedagogicamente; e aliar a isso uma metodologia que motive os alunos a praticar a resolução de problemas, um ato intelectual no qual a escrita de textos não passa de elemento acessório.

O primeiro teste de uma nova abordagem utilizando a ferramenta Scratch e exemplos estratégicos trouxe resultados bastante favoráveis. O teste destacou que a utilização de linguagens interpretadas foi útil, mas não é o único elemento de solução. Muitos alunos não conseguem solucionar de forma adequada um problema computacional, isso por não saberem interpretar o enunciado do problema e deles extrair as informações necessárias.

REFERÊNCIAS

ALA-MUTKA, Kirsti. Problems in Learning and Teaching Programming – a literature study for developing visualizations in the Codewitz-Minerva project. Comunicação Pessoal.Tampere University of Technology, 2004.

BEN-ARI, Mordechai.: Constructivism in computer science education. In: SIGSCE - Technical Symposium on Computer Science Education, Atlanta, GA, USA, pp. 257-261, 1998.

COOPER, Stephen; DANN, Wanda; PAUSCH, Randy. Teaching objects first in introductory computer science, In: Proceedings of the 34th Technical Symposium on Computer Science Education, Reno, Nevada, USA, 2003.

ENBODY, Richard J.; PUNCH, William F.; McCULLEN, Mark. Python CS1 as preparation for C++ CS2. In SIGCSE, Chattanooga, Tennessee, USA, 2009.

FALKNER, Katrina, PALMER, Edward. Developing authentic problem solving skills in introductory computing classes. In SIGCSE'09, Chattanooga,Tennessee, USA, 2009.

GOMES, Anabela; HENRIQUES, Joana; MENDES, António. Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. In Educação, Formação & Tecnoloigias; vol.1(1), pp. 93-103, 2003.

HAVENGA, Hester Maria. An investigation of students knowledge, skills and strategies during problem solving in object-oriented programming. Tese (Doctor of philosophy in mathematics, science and technology education) - University of South Africa. 2008.

JENKINS, Tony. On the difficulty of learning to program. In: Proceedings of 3rd Annual LTSN_ICS Conference (Loughborough University, United Kingdom, August 27-29). The Higher Education Academy, p.53-58, 2002.

MALONEY. John.; PEPPLER, Kylie; KAFAI, Yasmin.; RESNICK, Mitchel; RUSK, Natalie. Programming by Choice: Urban Youth Learning Programming with Scratch. In ACM SIGCSE Bulletin archive, Vol 40, 367-371, ISSN:0097-8418, 2008.

McIVER, Linda; CONWAY, Damian. Seven deadly sins of introductory programming language design, Monash University, Australia 1996.

MOSKAL, Barb; LURIE, Deborah; COOPER, Stephen. Evaluating the effectiveness of a new instructional approach. In SIGCSE 2000, Norfolk, Virgina, USA, 2000.

NAPS, Thomas; ROBLING, Guido; ALMSTRUM, Vicki; DANN, Wanda; FLEISCHER, Rudolf; HUNDHAUSEN, Chris; KORHONEN, Ari; MALMI, Lauri; McNALLY, Myles; RODGER, Susan; VELASQUEZ-ITURBIDE, J. Angel. Exploring the role of visualization and engagement in computer science education. In: ACM SIGCSE Bulletin, v.35. n.2, 2003.

PAPERT, Seymour. A máquina das crianças: repensando a escola na era da informática. Edição revisada. Porto Alegre: Artmed, 2008.

PARKER, Kevin. R.; CHAO, Joseph. T.; OTTAWAY, Thomas. A.; CHANG, Jane. A formal language selection process for introductory programming courses. Journal of Information Technology Education, vol. 5, 2000.

PEREIRA JÚNIOR, José Carlos Rocha.; RAPKIEWICZ, Clevi Elena. O Processo de Ensino e Aprendizagem de Algoritmos e Programação: Uma Visão Crítica da Literatura. In: III Workshop de Educação em Computação e Informática do Estado de Minas Gerais, WEIMIG'04, Belo Horizonte-MG, 2004.

POLYA, George. How to solve it: a new aspect of mathematical method. Princeton University Press, 2004.

WIRTH, Niklaus. Program Development by Stepwise Refinement, Communications of the ACM, vol. 14, pp. 221-227, 1971.