Metodologia ERM2C: Para melhoria do processo de ensino-aprendizagem de lógica de programação

Ricardo Luiz B. L. Campos[1]

[1]Serviço Federal de Processamento de Dados (SERPRO)
SGAN 601 Módulo G (SUPDE/DEBSA/DE1BD) – 70.836-900 – Brasília – DF – Brazil

rluizcampos@yahoo.com.br

Resumo. *A lógica de programação é considerada uma das mais importantes disciplinas para os alunos que ingressam em um curso superior de Tecnologia da Informação e Comunicação (TIC), pois os conhecimentos adquiridos nela irão influenciar diretamente no desempenho das demais disciplinas correlatas durante o restante do curso. Por ser uma disciplina com maior índice de reprovação, o seu ensino é considerado um dos sete grandes desafios do ensino de computação para este século. Assim sendo, pesquisas sobre novas metodologias para a melhoria do processo ensino-aprendizagem, como a ERM2C (Entender, Revisar, Melhorar, Completar e Construir) a ser apresentada neste artigo, é imprescindível.*

Abstract. *Logic programming is considered one of the most important subjects for students pursuing a degree in Information Technology and Communication (ITC) because it directly influences how students will perform in the correlated subjects for the remainder of the program. Teaching logic programming is considered one of the seven greatest challenges of computer science education in this century, due to its high rate of failure among students. Therefore, studies concerning new methodologies that would improve of the teaching/learning process, such as the ERM2C (Understand, Review, Improve, Complete, and Build), which is introduced in this article, are indispensable.*

1. Introdução

Em 2000, com mais de 22 anos de experiência com programação e mais de 2 anos de experiência em docência em Instituição de Ensino Superior (IES), tive a 1ª experiência com docente da disciplina de Lógica de Programação em outra IES

Ao final deste primeiro semestre, preocupei-me com o resultado final: mais de 60% dos alunos foi reprovados por não conseguirem construir algoritmos. E conclui que, apesar de ter experiência em lógica de programação, não consegui ensina-la com a metodologia encontrada nas referências bibliográficas indicadas na ementa da disciplina. Por isso, conclui que precisava encontrar uma outra metodologia para utilizar no ensino da disciplina.

No semestre final do 2º semestre ministrando a disciplina e utilizando uma outra estratégia para o ensino, o percentual de mais de 50%, apesar de ser menor que do semestre anterior, ainda era alto, conclui não sabia ensinar lógica de programação. Mas questionei-me: será que o problema era somente comigo ?

Para responder esta questão, através de uma pesquisa informal sobre o resultado da disciplina de lógica de programação ou similar durante os 8 últimos semestres em 4 IES do Distrito Federal (Tabela 1) foi possível constatar que mais de 60% dos alunos eram reprovados a cada semestre. Com este resultado, foi possível concluir que o problema do ensino da disciplina não era um problema exclusivamente meu. Mas será que isto é, também, uma realidade no Brasil e no exterior ?

Tabela 1 – Reprovação em Lógica de Programação no DF

Instituição	Semestre							
	1	2	3	4	5	6	7	8
A	57,14%	46,15%	33,33%	46,15%	33,33%	46,15%	33,33%	18,18%
B	33,33%	57,14%	75,00%	33,33%	75,00%	33,33%	57,14%	75,00%
C	75,00%	70,58%	57,14%	46,15%	88,88%	46,15%	75,00%	75,00%
D	88,88%	57,14%	33,33%	75,00%	87,50%	75,00%	57,14%	33,33%

A resposta é simples: a lógica de programação é uma das disciplinas com maior índice de reprovação em IES (CAMPOS, 2009) (CAMPOS, 2009b) (RAABE, DE SANTIAGO e DAZZI, 2007) (MARCZAK e GIRAFFA, 2003) (DOS SANTOS e COSTA, 2006) (FACKEMBACH e DE ARAUJO) (PEREIRA JÚNIOR e outros, 2005) (RAPKIWCZ e outros, 2006). Por este motivo, segundo McGettrick (*apud* WESLEY, GONDIM e AMBRÓSIO, 2008), é considerado, o seu ensino, um dos sete grandes desafios do ensino de computação para este século. Mas qual a justificativa para isto ?

De acordo com Raabe, de Santiago e Dazzi (2007), a lógica de programação é uma das mais importantes disciplinas para os alunos que ingressam em um curso superior de Tecnologia da Informação e Comunicação (TIC), pois os conhecimentos adquiridos nela vão influenciar diretamente no desempenho das demais disciplinas correlatas durante o restante do curso. Assim sendo, é indispensável identificar quais os motivos do alto índice de reprovação, bem como quais as alternativas possíveis para reduzi-lo (CAMPOS, 2009) (CAMPOS, 2009b).

Como este objetivo, através de entrevistas individuais com 60 alunos de 2 turmas de umas das IES, verificou-se que a grande maioria (65%) tem dificuldade em construir a solução para resolver um problema proposto – o algoritmo –, mesmo que este problema seja classificado como de baixa complexidade (Figura 1).

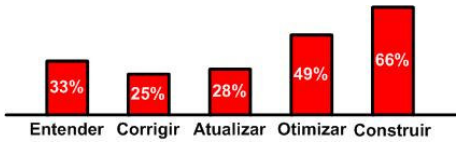

Figura 1 – Dificuldades referente à Lógica de Programação

A dificuldade para se construir a resolução do problema está relacionada com (CAMPOS, 2009) (CAMPOS, 2009b): (i) qual o nível de detalhamento que a solução deve ter; (ii) e a dificuldade dos discentes em associar os procedimentos estabelecidos na linguagem natural com os procedimentos na linguagem de programação.

O último item pode ser resolvido através da aplicação de uma regra que restrinja o conjunto de ações possíveis de um algoritmo para um universo igual de comandos padrões da maioria das linguagens de programação (PYOTT e SANDERS, 1991).

Já o primeiro, o detalhamento, está relacionado com a habilidade adquirida pela combinação do conhecimento proveniente dos estudos com a destreza resultante da prática que advém com o tempo (VILLAS e outros, 1987). Mas como adquirir experiência se há dificuldade na construção de uma solução ?

Para tentar responder esta pergunta, foram entrevistados os docentes das 2 IES visando identificar as metodologias utilizadas no ensino da disciplina. O resultado foi que a maioria adotava a forma apresentada pela maioria das referências bibliográficas indicadas para a disciplina (Figura 2): a construção da solução para resolução do problema proposto. Além desta estratégia, alguns professores ensinavam os alunos, também, a avaliar e/ou corrigir e/ou atualizar e/ou otimizar uma solução existente, para, depois, ensinar a criar.

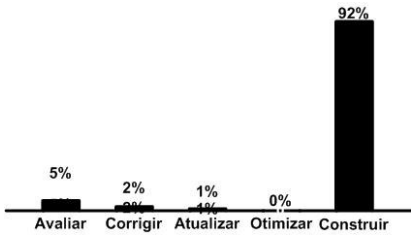

Figura 2 – Estratégias para o ensino de lógica de programação

A questão então é: se os alunos têm dificuldade em construir a solução para resolver um problema proposto e esta é a forma que a maioria dos docentes utiliza no ensino da disciplina, como será possível garantir o aprendizado e, também, melhorar o percentual de aprovação dos alunos ?

Esta resposta não é tão fácil. Mas, se considerado que a capacidade para a criação da solução de um problema é a habilidade a ser adquirida combinada a partir do conhecimento obtido a partir de estudos com a destreza proveniente da prática que advém do tempo, uma alternativa pode ser uma metodologia que contenha todas as práticas identificadas na pesquisa: (i) avaliação de uma solução existente; (ii) a manutenção de uma solução existente, seja para corrigir um problema ou para atualização; (iii) a otimização de uma solução existente; e (iv) a criação de uma nova solução.

A metodologia ERM2C (Entender, Revisar, Modificar, Complementar e Construir), que é objeto deste artigo, tem este objetivo. Ela agrupa as melhores práticas utilizadas pelos docentes com menor índice de reprovação.

Durante 4 semestres letivos, a ERM2C foi testada com um grupo de 88 alunos voluntários das 4 IES pesquisadas. Em cada um dos semestres, 2 grupos de alunos foram trabalhados: um grupo com alunos que estavam cursando a disciplina pela primeira vez; o outro, por alunos que já tinham cursado a disciplina anteriormente e não obtiveram aprovação ao final do semestre.

O resultado final deste teste, embora ainda não conclusivo, foi satisfatório, pois o percentual de alunos reprovados foi reduzido de 37% para 18% dos alunos novatos e de 23% para 8% dos repetentes.

Além deste teste, a metodologia foi utilizada em um curso regular e particular de Lógica de Programação. Neste curso, havia 12 alunos entre novatos e repetentes que estavam matriculados em curso superior em TIC e apenas 25% deles foram reprovados no final do semestre, por desistência da disciplina.

Com o objetivo de apresentar a Metodologia ERM2C este artigo foi dividido em 4 partes: a introdução, a primeira; a 2ª, a apresentação da metodologia e um exemplo da sua aplicação; os resultados alcançados até o presente com a utilização da metodologia proposta serão apresentados na parte 3; e na 4ª, a conclusão deste artigo e a apresentação de trabalhos a serem desenvolvidos futuramente.

2. A Metodologia ERM2C

Para a melhoria do processo ensino-aprendizagem da disciplina de lógica de programação, os professores/pesquisadores trabalham com 3 vertentes (PEREIRA JÚNIOR e outros, 2005): Ferramentas; Estratégias; e Ferramentas e Estratégias. Destas vertentes, a ERM2C é uma nova estratégia para o ensino de lógica de programação.

Uma metodologia para o ensino desta disciplina, conforme Baeza-Yates, deve permitir, ao aluno, o entendimento das técnicas básicas de construção de algoritmos e a habilidade para analisar algoritmos, além do entendimento do problema de acordo com a sua complexidade.

Com todas essas capacidades, o programa, resultado de um algoritmo desenvolvimento de discente, será bom (CHANTLER, 1984): fazendo o que espera que ele faça (precisão) e continuando a fazer sempre (confiabilidade); manipule precisa e corretamente todos os dados (potência) sem quaisquer processamentos desnecessários (eficiência); e que qualquer alteração possa ser realizada facilmente (manutenibilidade).

Mas tendo pouca ou nenhuma experiência, torna-se difícil ou quase impossível que um aluno consiga fazer com que seu algoritmo tenha precisão, confiabilidade, potência, eficiência e manutenibilidade, ou seja, que tenha capacidade para um bom programa (CAMPOS, 2009) (CAMPOS, 2009b).

No entanto, caso o discente consiga desenvolver um algoritmo, como ele poderá comprovar que o resultado será um bom programa, ou seja, como poderá comprovar ou validar a precisão, confiabilidade, potência, eficiência e manutenibilidade ?

A resposta é simples, caso ele, antes de aprender a desenvolver a lógica para a solução de um problema, tenha aprendido a ler/entender qualquer algoritmo, seja de terceiros ou própria, com ou sem a definição do problema. Pois, a partir de bons exemplos pode-se construir produtos iguais ou melhores (CAMPOS, 2009b).

Baseado nesta premissa é que a metodologia ERM2C foi desenvolvida. A mesma foi dividida em 5 etapas, onde o discente adquirirá as capacidades para entender, revisar, melhorar e complementar um algoritmo pronto, para depois, construir. Em cada uma dessas capacidades, que foi denominada de etapa, há 3 níveis distintos: o básico; intermediário; e o avançado.

A ERM2C é uma metodologia na qual a característica individual do aluno é preservada. Assim sendo, a evolução do discente e não a da turma é que permitirá, dentro de um nível, as mudanças entre as 5 etapas e, também, as mudanças entre os 3

níveis. Ou seja, para a mudança do nível atual para o seu posterior, exceto o nível avançado que indica a conclusão da disciplina, o discente deverá ter concluído, com sucesso, todas as atividades da etapa Entender até a Construir do nível e, para a mudança da etapa atual para a sua próxima etapa, o aluno deverá ter concluído as atividades, com sucesso, da atual etapa.

2.1 – Etapas da ERM2C

A metodologia ERM2C (Tabela 2) está dividida em 5 etapas: Entender; Revisar; Melhorar; Complementar; e Construir. Sendo que cada etapa possui 3 níveis: Básico; Intermediário; e Avançado.

Tabela 2 – Etapas da ERM2C

Etapa	Objetivo	Pré-requisito	Atividades
Entender	Capacitar o discente a interpretar um algoritmo, independente de quem tenha sido o autor do mesmo.	▪ Conceitos básicos de lógica de programação (ações, proposições, conjunção de proposições, disjunção de proposições, negação de proposição, expressões lógicas e operadores relacionais); ▪ Forma de representação da lógica para resolução do problema (no caso a ***LiA***, uma linguagem gráfica para representação de algoritmo desenvolvida para a metodologia); ▪ Ações que podem ser utilizadas em um algoritmo	A partir de um algoritmo pronto (desenvolvido e validado pelo professor), o aluno deverá responder um conjunto de questões sobre o processamento da solução.
Revisar	Capacitar o aluno a indicar, se existir, onde um determinado problema relatado está ocorrendo em um algoritmo.	▪ Ter capacidade de ler e entender um algoritmo.	A partir de um algoritmo pronto e com a descrição de um problema, o discente deverá responder se o problema existe ou não, e, existindo, indicar qual a sua localização.
Melhorar	Capacitar o aluno a propor as alterações em algoritmo visando correção de erros ou melhoria no seu desempenho.	▪ Ter capacidade de ler e entender um algoritmo e identificar problemas, se houver, no mesmo.	Atividades distintas a serem desenvolvidas: ▪ A partir de um algoritmo com problemas já identificados anteriormente (posição e descrição do mesmo), deve propor as modificações para que o mesmo venha fazer o que se espera (precisão); ▪ A partir de um algoritmo pronto irá identificar e substituir as ações que, embora contribuam para que o resultado final esperado seja alcançado, reduzem o desempenho do algoritmo (eficiência).

Etapa	Objetivo	Pré-requisito	Atividades
Complementar	Capacitar o discente a propor um conjunto de ações necessárias para que o algoritmo incompleto (que necessita de evolução e/ou adaptação por causa da alteração do seu objetivo inicial) faça com que o novo resultado final esperado seja alcançado.	▪ Ter habilidade para entender, revisar e melhorar um algoritmo pronto.	A partir de um algoritmo pronto em conjunto com as suas duas definições (a atual e a nova) irá propor um conjunto de ações necessárias para que o novo resultado esperado seja alcançado.
Construir	Capacitar o aluno a desenvolver uma solução lógica para o problema proposto.	▪ Ter capacidade de entender, revisar, melhorar e complementar um algoritmo pronto.	A partir da definição de um problema apresentar um algoritmo completo que produza o resultado final esperado.

2.2 – Um Exemplo da Aplicação da ERM2C

Para que a metodologia possa ser utilizada na disciplina de lógica de programação ou similar é necessário entender como a mesma pode ser conduzida.

Na etapa "Entender", um algoritmo pronto e validado é apresentando ao aluno (Figura 3) e algumas questões sobre o processamento é feita ao discente, como, por exemplo, "qual será o valor de X (elipse azul com a ação APR), se os valores informados para A e B (elipse azul com a ação RCB) forem 2 e 3 ou 5 e 4?". O aluno irá percorrer o algoritmo desde da ação inicial (elipse laranja com a ação INI) até a ação final (elipse laranja com a ação FIN) para entender que será apresentado o maior valor entre os dois valores recebidos (o objetivo do algoritmo). As respostas esperadas pelo discente serão, respectivamente, 3 e 5.

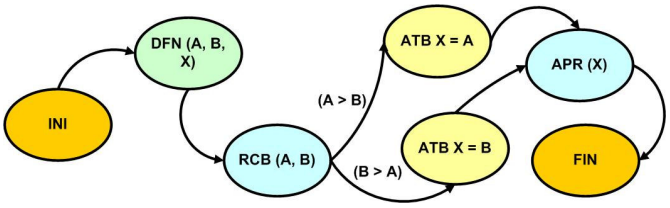

Figura 3 – Etapa: Entender

Como no algoritmo anterior (Figura 3) se os valores de A e B sejam iguais não haverá a apresentação do resultado, ou seja, há um erro de lógica. Ao questionar o discente sobre a existência de tal erro e que ele indique, se for verdade, onde o mesmo ocorre, a etapa "Revisar" está sendo trabalhada.

O discente deverá, então indicar (quadro vermelho na Figura 4) que o erro está em um dos desvios condicionais (A>B ou B>A). E que um dos 2 deve ser alterado. É isto que se espera do discente para estar apto na etapa "Revisar".

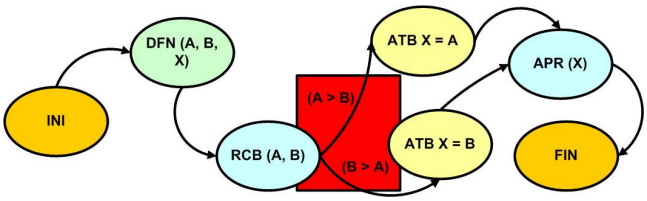

Figura 4 – Etapa: Revisar

Com o resultado da etapa "Revisar", o discente irá propor, como uma das atividades da etapa "Melhorar", a correção deste problema e tornar o algoritmo preciso. Neste exemplo, há várias alternativas, sendo uma delas a existência de duas expressões lógicas: uma, a (B>A), quando o valor de B for maior que o valor de A; e outra, a NÃO (B>A), quando o valor de B não for maior que o valor de A, ou seja, quando o A for igual ou maior que o B. Com esta indicação pelo aluno, o mesmo estará apto para iniciar a próxima atividade da etapa "Revisar".

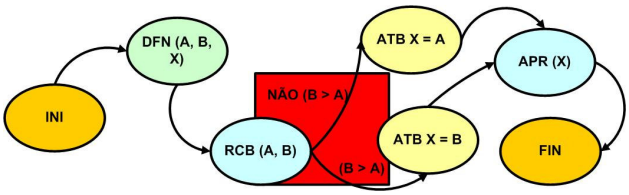

Figura 5 – Etapa: Melhorar (Erro)

A segunda atividade da etapa "Revisar", é identificar e substituir as ações que reduzem a eficiência do algoritmo. O algoritmo anterior (Figura 5) poderá ser melhorado conforme a proposta do quadro vermelho da Figura 6.

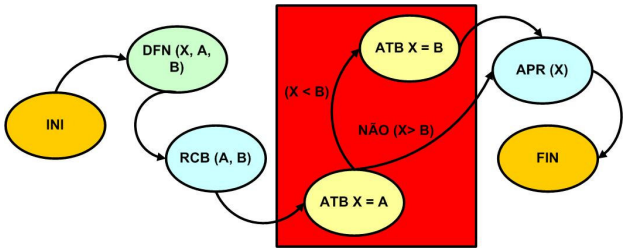

Figura 6 – Etapa: Melhorar (Desempenho)

Com a conclusão da etapa "Melhorar", o discente estará apto para a etapa "Complementar", onde deverá propor modificações no algoritmo apresentado para atender as novas necessidades. Quais as ações necessárias, por exemplo, para que o algoritmo anterior possa receber 3 valores e identificar o maior deles.

O discente deverá avaliar o algoritmo e identificar as ações que serão alteradas e as novas ações que serão inseridas (quadro vermelho da Figura 7), ou seja, a definição de mais uma variável (elipse verde com a ação DFN), a recepção dos valores (elipse azul com a ação RCB), as expressões lógicas e as ações de atribuições (elipse amarela com a ação ATB).

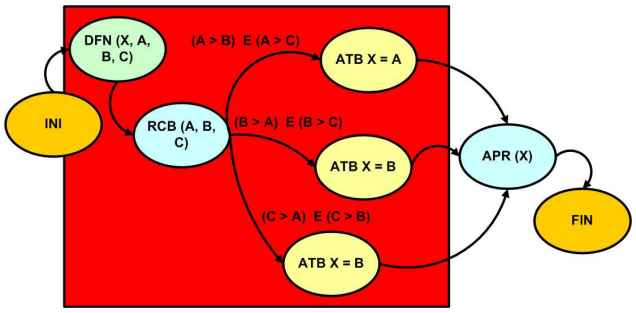

Figura 7 – Etapa: Complementar

Com a conclusão da etapa "Complementar", o aluno estará apto a desenvolver um algoritmo, ou seja, a iniciar a etapa "Construir". Nesta etapa, o discente com um problema (por exemplo, qual a lógica que permita identificar o maior e o menor de 3 números recebidos através do teclado) proporá uma lógica para resolve-lo. O importante é entender que o aluno já tem experiência com este problema ou similar, então a sua resolução será mais fácil.

3. Resultados Alcançados

A metodologia ERM2C, durante quarto semestres, foi aplicada como complementação de estudos para alunos matriculados efetivamente na disciplina de lógica de programação em 2 instituições de ensino superior, em horário alternativo às aulas, com duração de 48 horas/aula.

Os alunos que participaram da experiência eram voluntários e foram divididos em 2 grupos (cada qual com 11 alunos/semestre): o primeiro, com alunos que estavam cursando a disciplina pela primeira vez; e o segundo, com alunos que já tinha cursado a disciplina e não obtiveram aprovação ao final do semestre. Cada um dos grupos era trabalhado em horários distintos, não sendo permitido a troca de horário, mesmo que ocasionalmente.

O resultado final, embora não conclusivo, pode ser considerado satisfatório, tendo em vista que o percentual de reprovados reduziu de 37% para 18% para os alunos novatos e de 23% para 8% para os repetentes.

Após a conclusão a conclusão do quarto e último semestre, não houve continuidade da aplicação da metodologia por dificuldade de encontrar novos voluntários, bem como pelo afastamento do autor da atividade de docente.

Mas a metodologia foi utilizada em um curso regular e particular de Lógica em Programação no 2º semestre de 2009. Este curso foi ministrado em 48 horas, divididos em 3 meses, com 4 encontros mensais, cada um deles com 4 horas de duração.

O grupo de alunos era formado de 4 alunos novatos e de 8 repetentes e todos estavam matriculados em curso superior de TIC em IES. E deste grupo apenas 25% foram reprovados (neste percentual foram incluídos 2 alunos que desistiram da disciplina no IES e, conseqüentemente, do curso).

Considerando apenas os alunos não desistentes, o percentual de aprovação foi de 100% para os alunos novatos e de mais de 83% para os repetentes.

4. Conclusão e Trabalhos Futuros

A disciplina de lógica de programação é uma das com maior índice de reprovação nas IES. Por este motivo, o seu ensino é considerado um dos sete grandes desafios do ensino de computação do século.

O principal motivo deste problema é que os alunos sentem dificuldade em desenvolver a lógica para a resolução de um problema, mesmo que ele seja considerado de baixa complexidade. Principalmente, por não saberem por onde começar e para onde ir.

Para tentar reduzir ou eliminar este problema, a Metodologia ERM2C foi desenvolvida. Ela propõe que o ensino de lógica de programação desenvolva no discente as capacidades para: ler um algoritmo; identificar as ações erradas em algoritmo; propor modificações em um algoritmo; inserir novas ações em um algoritmo. E somente após estas capacidades serem adquiridas, é que o aluno irá desenvolver a capacidade para construir algoritmo.

Como é uma metodologia na qual a característica individual do aluno é preservada, durante o processo é a evolução do aluno que permitirá a mudança de um nível para o seu posterior, bem como de uma etapa para a sua posterior dentro de um nível.

Esta metodologia foi aplicada em 80 voluntários durante 4 semestres como complementação de estudos e, também, em um grupo de 12 alunos num curso regular e particular. O percentual de reprovação, em ambos os casos, foi inferior ao anteriormente identificados na pesquisa informal.

O resultado apesar de satisfatório, ainda é inconclusivo. Mas acredita-se que a ERM2C pode ser utilizada como uma metodologia na disciplina de lógica de programação ou similar, seja em cursos técnicos ou de nível superior.

Como trabalho futuros, uma nova turma de um curso regular de lógica de programação será montada no 1º semestre de 2010 e, para que a metodologia possa ser utilizada por outros profissionais, a publicação do livro "Lógica de programação: Estudo dirigido utilizando a metodologia ERM2C".

Referências

BAEZA-YATES, R. Teaching algoritms. Depto. de Ciências da La Computacion. Universidade de Chile, Santiago, Chile.

CAMPOS, R. L. B. L. Lógica de programação: Há como melhorar o aprendizado fugindo dos padrões estabelecidos nos livros didáticos e adotados pela maioria dos docentes ? XVII Congresso Iberoamericano de Educacion Superior em Computacion (CLEI-2009-CIESC), 22-25/Set/2009. Brazil, RS, Pelotas.

CAMPOS, R. L. B. L. ERMC^2 : Uma proposta de metodologia para melhoria do ensino-aprendizado de lógica de programação. XI Congresso Chileno de Educacíon Superior em Computacion (CCESC), 2009. Santiago, Chile, Jornadas Chilenas de Computacion 2009, Santiago, Chile, 2009b, Vol. Único.

CHANTLER. A. Técnicas e práticas de programação. Rio de Janeiro, RJ, Brasil: Campus, 1984.

DOS SANTOS, R. P.; COSTA, H. A. X. Análise de Metodologias e Ambientes de Ensino para Algoritmos, Estruturas de Dados e Programação aos iniciantes em Computação e Informática. Infocoomp, Journal of Computer Sciense,, Volume 5, Number 1, March 2006. Universidade Federal de Lavras – UFLA, Lavras, MG, Brasil.

FALCKEMBACH, G. A. M.; DE ARAUJO, F. V. Aprendizagem de algoritmos: dificuldades na resolução de problemas. Universidade Luterana do Brasil – ULBRA, Santa Maria, RS, Brasil. Faculdade Dom Alberto, Santa Cruz do Sul, RS, Brasil.

JERIZ, R. Sugestão para avaliação dos alunos no início do semestre. Universidade Católica do Norte do Chile. Conversar informal durante as Jornadas Chilenas de Computacional de 2009.

MARCZAK, S. dos S.; GIRAFFA, L. M. M. Ambientes Inteligentes para Suporte ao Ensino de Programação. Techical Reports Series, Number 28, August, 2003, Faculdade de Informática – PUCRS.

PEREIRA JÚNIOR, J e outros. Ensino de algoritmos e programação: uma experiência no nível médio. XXV Congresso da Sociedade Brasileira de Computação, Unisinos, São Leopoldo, RS, Brasil, 22 a 29 de Julho, 2005.

PYOTT, S.; SANDERS, I. ALEX: An aid to teaching algortims. SIGCSE, Vol. 23, Nº 3, Bulletin, Set/1991, pag. 36.35.

RAPKIEWICZ, C. E. e outros. Novas Tecnologias na Educação. CINTED-UFRGS, V. 4 Nº 2, Dezembro, 2006.

RAABE, A. L. A. Raabe; DE SANTIAGO, R.; DAZZI, R. L. S.. Adquirindo experiência na construção de ferramentas de apoio a aprendizagem de algoritmos. Workshop de Ambientes de Apoio a Aprendizagem de Algoritmos e Programação. Simpósio Brasileiro de Informática na Educação, 2007. Anais dos XVIII Simpósio Brasileiro de Informática na Educação, 2007.

VILLAS, M. V. e outros. Programação: conceitos, técnicas e linguagens. Rio de Janeiro, RJ, Brazil: Campus, 1987.

WESLEY, H.; GONDIM, A. P.; AMBROSIO, A. P. Esboços de fluxogramas no ensino de algoritmos. Workshop sobre Educação em Computação – WEI. Anais do XXVIII Congresso da Sociedade Brasileira de Computação. Belém do Pará, PA, Brasil, 12 a 18 de julho de 2008.

XAVIER, G. M. M e outros, C. Estudo de fatores que influenciam a aprendizagem introdutória de programação. Disponível em http://www.uefs.br/erbase2004/documentos/weibase/Weibase2004_Artigo002.pdf).