VisuAlg - Ferramenta de Apoio ao Ensino de Programação

Cláudio Morgado de Souza[1]

[1]Universidade Severino Sombra, CECETEN, Bacharelado em Sistemas de Informação, claudio.morgado@uss.br

Resumo. *Historicamente, o ensino de programação tem sido considerado de difícil entendimento para os alunos por diversos motivos: falta de preparo dos estudantes, ausência de uma didática adequada e de ferramentas computacionais que ajudem os atores (professores e estudantes) a superarem os problemas que se apresentam no processo ensino-aprendizagem. Neste artigo discutem-se alguns aspectos das dificuldades no ensino de programação e apresenta-se o VisuAlg, programa que pretende auxiliar estudantes e professores a obter um maior entendimento e rendimento no estudo de disciplinas da área de programação.*

Palavras-Chave: Algoritmos, *Software* educacional, programação

1 Introdução

É prática comum nos cursos da área de Computação se começar o ensino de programação de computadores por uma disciplina na qual são discutidos conceitos básicos como tipos de dados, variáveis e constantes, ao mesmo tempo em que são apresentados comandos de entrada e saída de dados, operadores de atribuição, aritméticos, relacionais e lógicos (abordagem tradicional). Os componentes principais da programação estruturada (sequência, decisão e iteração) também são apresentados nesta disciplina, e alguns cursos introduzem conceitos mais avançados como subprogramas, vetores e recursividade.

Geralmente esta primeira disciplina não usa uma linguagem de programação real como Pascal, C ou Java, preferindo codificar os algoritmos estudados em uma forma de pseudocódigo que fica entre o formalismo das linguagens de programação e a liberdade da descrição em linguagem corrente. O Portugol, nome usual desta forma de pseudocódigo, permite que o aluno se concentre inicialmente na solução do problema proposto sem ter que dominar a sintaxe de uma linguagem de programação, mas ao mesmo tempo já apresenta conceitos como declaração de variáveis, linha de código, palavras-chave, comentários, dentre outros, que serão depois transpostos com mais facilidade para ambientes reais de desenvolvimento.

O restante deste artigo está organizado da seguinte forma: na seção 2 discutem-se as vantagens e desvantagens da abordagem tradicional em vista das especificidades do ensino de lógica de programação. Na seção 3 são citadas ferramentas, entre as quais está o **VisuAlg**, que se propõem a eliminar os problemas mostrados na seção 2, usando abordagens alternativas. Na seção 4 apresenta-se o *software* **VisuAlg** e suas funcionalidades. A seção 5 discute a aplicação do **VisuAlg** no laboratório de programação e as várias maneiras com que os alunos fazem uso do programa, e finalmente a seção 6 apresenta as considerações finais do trabalho.

2 Prós e Contras do Ensino Tradicional

Um ponto positivo da abordagem tradicional é a possibilidade de se usar apenas lápis e papel para se codificar um algoritmo, sem necessidade de computadores, interpretadores ou compiladores. Entretanto, para a maioria dos alunos esta estratégia se revela muito abstrata, pois não conseguem associar, por exemplo, a execução dos comandos `escreva("Digite um valor:")` e `leia(valor)` à exibição de um *prompt* na tela do computador e ao cursor esperando a entrada de dados, nem perceber como o fluxo de processamento ocorre na execução de comandos de decisão e iteração. A isto se soma o conhecido fato de que o aluno ainda enfrenta o obstáculo de entender o problema, criar por si uma solução ou compreender aquela apresentada por seus colegas ou pelo professor, isto é, a lógica de programação em si. Uma discussão sobre esta dificuldade e as razões para o despreparo dos alunos pode ser encontrada em Koliver, Dorneles e Casa (2004), e está fora dos propósitos deste artigo.

Pelo lado do docente, pode-se ver que a programação é uma disciplina com especificidades tais que exigem que seu ensino seja diferenciado, abandonando recursos estáticos tais como transparências, textos ditados, diagramas, etc. Este trabalho e o de Gomes, Henriques e Mendes (2008) propõem uma mudança pela qual a abordagem *bottom-up* (começar pela história das linguagens de programação, detalhes sintáticos de linguagens, conceitos de programação estruturada, etc.) do ensino de programação seria substituída por uma *top-down* (mostrar a finalidade e utilidade de se aprender programação, e se começar logo a produzir programas).

3 Novas Abordagens e Experiências

Experiências têm sido feitas para se desenvolver ferramentas de apoio ao ensino de lógica de programação que eliminem, ou ao menos minimizem, os obstáculos citados. Umas usam abordagens ortodoxas, geralmente interpretadores de pseudocódigo como o Interpretador de Linguagem Algorítmica (ILA), desenvolvido sob a coordenação do Prof. D. Sc. Sérgio Crespo, da UNISINOS [Crespo 1990], e o Ambiente de Aprendizado de Programação (AMBAP), desenvolvido na UFAL sob a orientação da Profa. D. Sc. Eliana Silva de Almeida [Almeida 2001], e outras fazem uso de jogos computacionais [Rapkiewicz et al 2006], geração e interpretação de fluxogramas [Gondim e Ambrosio 2008], e animação de algoritmos [Mota, Pereira e Favero 2008], entre outras.

Como contribuição a este esforço foi proposto o desenvolvimento de uma ferramenta que se assemelhasse a um Ambiente Integrado de Desenvolvimento (*Integrated Development Environment* ou IDE) formal, como o *Delphi* ou *Visual Basic*, e que incorporasse recursos voltados para o ambiente acadêmico. O aplicativo seria dirigido especificamente para alunos iniciantes dos cursos de programação e seus professores. Os requisitos mínimos para tal ferramenta seriam:

- um editor de textos com as facilidades esperadas em uma aplicação com interface gráfica moderna, tais como salvar, carregar, copiar e colar, desfazer, etc.;
- uma "linguagem de programação" subjacente, com os comandos necessários para entrada e saída de dados, estruturas de decisão `if...then...else`, `case...of`, e as estruturas de repetição `for..next`, `while...do` e `repeat...until`, contemplando ainda os tipos de dados básicos (inteiro, real, literal e booleano), e variáveis simples e vetores;

- a possibilidade de se examinar a qualquer momento o conteúdo das variáveis do programa;
- simulação do *console* do computador, pelo qual o usuário vai interagir com o pseudocódigo sendo executado.

Os requisitos não-essenciais, mas desejados, seriam:

- suporte a subprogramas, naturalmente com variáveis locais e recursividade;
- suporte a constantes e tipos de dados estruturados (*records*);
- suporte a leitura e escrita em arquivos sequenciais e de acesso aleatório.

Estes últimos requisitos são considerados não-essenciais porque na maioria cursos quando o aluno está apto a utilizar tais recursos já ultrapassou o primeiro semestre e já está programando em linguagens como C e Java. Acredita-se que as funcionalidades citadas na primeira lista possibilitam a implementação de uma vasta gama de algoritmos dos mais variados tipos, permitindo a apresentação, prática e fixação dos fundamentos da lógica de programação.

VisuAlg: Interface e Funcionalidades

O **VisuAlg** é um aplicativo que fornece aos estudantes que se iniciam nas disciplinas de programação ferramentas para digitar, executar e depurar o pseudocódigo para resolver problemas propostos nas aulas e em exercícios, fornecendo também aos professores vários recursos didáticos para que expliquem como os programas funcionam, tais como execução passo a passo, visualização do conteúdo das variáveis, exame da pilha de ativação no caso de subprogramas, contador de execuções de cada linha do programa, etc.

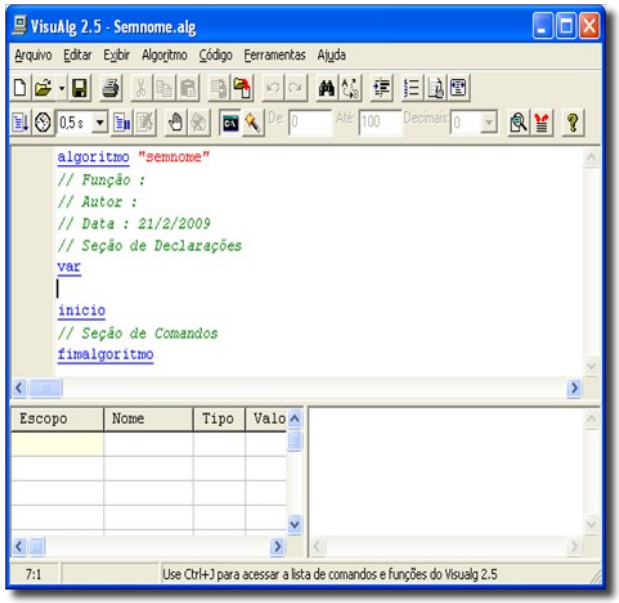

Figura 1. A tela principal do VisuAlg.

A Figura 1 mostra a tela de abertura do **VisuAlg**, na qual se vê a área de digitação do pseudocódigo no centro, a memória do programa abaixo, à esquerda, e o *console* que exibe as saídas, embaixo à direita.

A memória é uma lista que contém as variáveis utilizadas no programa, e possui quatro colunas:

- escopo da variável, que pode ser global ou local, caso em que o nome da subrotina onde foi declarada é exibido;
- nome da variável; no caso de elementos de um vetor, o nome aparece seguido do índice ou índices do elemento, pois nesta lista cada um é considerado uma variável em separado;
- tipo da variável, que pode ser caractere, inteiro, real ou lógico;
- valor da variável.

A lista de variáveis possui também uma coloração diferente para tipos diferentes de variáveis no que tange a seu uso no pseudocódigo:

- variáveis declaradas na seção VAR (globais ou locais) são apresentadas em preto;
- variáveis que representam parâmetros passados por valor são apresentadas na cor verde;
- variáveis que representam parâmetros passados por referência são apresentadas em vermelho, e na segunda coluna está também o nome da variável externa à qual se refere, isto é, o argumento passado ao subprograma;
- variáveis que representam o retorno de uma função são representadas em azul e no lugar do nome está a expressão (RETORNO).

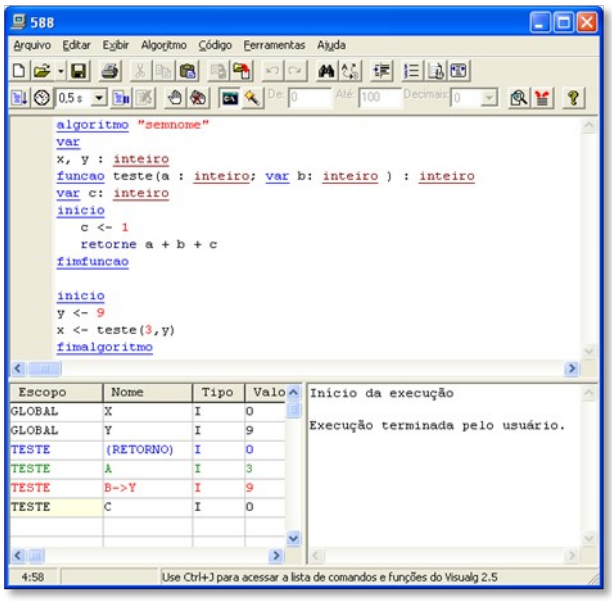

Figura 2. A simulação da memória de um programa no VisuAlg, abaixo, à esquerda.

Quando um programa está sendo executado no **VisuAlg** existem duas opções para a interação com o usuário: o modo chamado pelo programa de modo DOS (Figura 3), no qual uma janela simula a tela de um computador em modo texto, e o modo padrão (Figura 4), no qual as saídas são direcionadas ao *console* citado anteriormente, e as

entradas são obtidas através de uma janela de diálogo exibida quando da execução do comando `leia()`.

O modo DOS é indicado para quando o aluno está testando seu programa, por proporcionar um maior realismo; o modo padrão é útil para quando o professor está executando um programa passo a passo junto com os alunos para ilustrar algum conceito ou técnica de programação, já que a tela DOS não aparece sobre a tela principal do programa. A mudança entre um modo e outro pode ser feita através do menu ou da barra de ferramentas do **VisuAlg**.

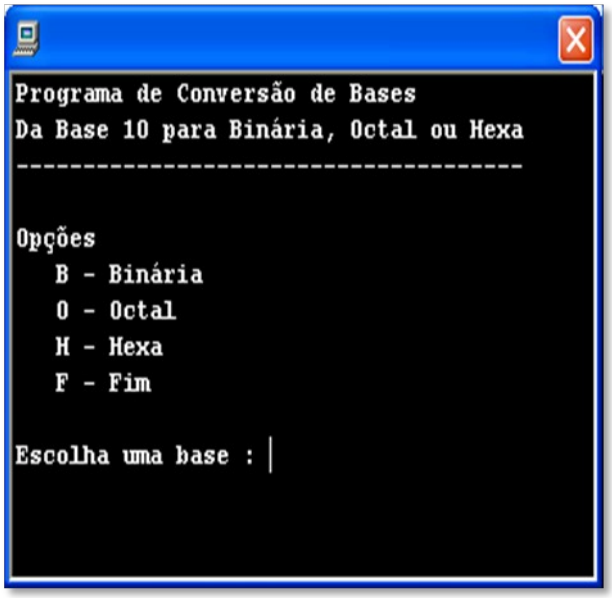

Figura 3. A execução de um programa no VisuAlg, no modo DOS.

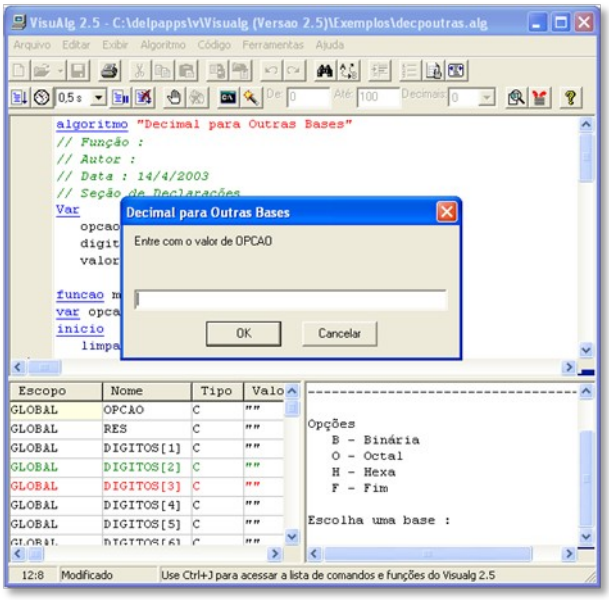

Figura 4. A execução de um programa no VisuAlg, no modo padrão.

A linguagem utilizada pelo interpretador é uma das muitas variações do Portugol, amplamente utilizado nos cursos de computação no Brasil. A sintaxe é parecida com a do Pascal, mas sem o ponto-e-vírgula para separar comandos. Optou-se por uma abordagem parecida com a do BASIC, com um comando por linha obrigatoriamente, para facilitar e digitação e entendimento do pseudocódigo, e também porque não é considerada boa prática de programação a colocação de mais de um comando por linha, o que diminui a legibilidade do código. Comentários podem ser colocados no pseudocódigo com o uso de //. As palavras reservadas e comandos do Portugol utilizado no **VisuAlg** podem ser encontrados em [Souza 2003]. A lista das funções pré-definidas pode ser encontrada em [Souza 2007]. A Tabela 1 apresenta um exemplo de pseudocódigo ilustrando várias características da linguagem de programação do **VisuAlg**.

Tabela 1. Exemplo de pseudocódigo usando a sintaxe do VisuAlg.

```
algoritmo "Bubble Sort"
// Força a execução do programa na janela DOS
dos
var
   a, b : inteiro
   x : vetor[1..10] de real

procedimento troca( var a, b : real )
var t : real
inicio
    t <- a
    a <- b
    b <- t
fimprocedimento

inicio
// Leitura dos dados
// Gerar numeros aleatorios entre 0 e 100, com 3 casas
decimais
aleatorio 0,100,3
para a de 1 ate 10 faca
   leia(x[a])
fimpara
// Ordenação
para a de 1 ate 10 faca
   para b de 1 ate 9 faca
      se x[b] > x[b+1] entao
         troca(x[b],x[b+1])
      fimse
   fimpara
fimpara
// Impressão dos dados ordenados
para a de 1 ate 10 faca
   escreval(a:3," - ", x[a] : 10 : 3)
fimpara
```

```
fimalgoritmo
```

A execução do programa pode ser feita de três maneiras: o modo padrão, em que as instruções do pseudocódigo são interpretadas e executadas imediatamente, o modo passo a passo, comum a vários ambientes de desenvolvimento, em que o usuário comanda a execução do programa linha por linha, para efeitos de depuração ou, no caso do **VisuAlg**, para que o estudante possa examinar as variáveis à medida que seus valores mudam, perceber o fluxo de processamento em estruturas de decisão e repetição, etc., e o modo de animação, parecido com o modo padrão, com a diferença que antes da execução de uma linha de código há uma pausa que pode variar de 0.2 segundos a 5 segundos, o que permite ao estudante acompanhar o fluxo de processamento. A mudança entre estes modos de execução pode ser feita através do menu principal ou da barra de ferramentas, e também com comandos incluídos no código: por exemplo, o comando `timer on` liga a animação, e o `timer off` a desliga. Durante a animação, qualquer atribuição de valor a uma variável faz com que ela se torne visível e destacada na "memória" do programa, auxiliando na explicação do funcionamento de laços, contadores, acumuladores, etc.

Outras ferramentas didáticas do **VisuAlg** associadas à execução do código são os comandos `pausa` e `debug`. O primeiro funciona como um *breakpoint* disparado por código. Quando a execução atinge o comando `pausa`, independente do modo em que estiver irá para o modo passo a passo. Neste ponto o estudante poderá realizar a análise necessária em seu código, e continuar a execução comando por comando ou voltar para o modo em que estava. O comando `debug` tem como argumento uma expressão lógica, e interrompe a execução do programa caso o resultado desta expressão seja verdadeiro. Funciona como um *breakpoint* condicional.

O **VisuAlg** dá suporte a subrotinas (procedimentos e funções) e permite o exame da pilha de ativação destas subrotinas quando a execução está em modo passo a passo. Esta ferramenta permite ao professor ilustrar o mecanismo de chamadas de subrotinas e recursividade, por exemplo, conforme se pode ver na Figura 5.

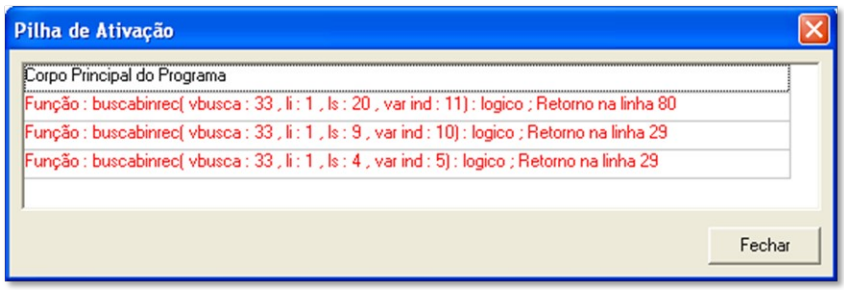

Figura 5. A pilha de ativação de subrotinas no VisuAlg**.**

Muitas vezes, para se examinar a eficiência de uma solução para um problema, como nos algoritmos de ordenação, é necessário se saber quantas vezes uma determinada linha de código foi executada. O **VisuAlg** possui uma ferramenta chamada Perfil de Execução, que permite que esta estatística seja analisada durante a execução de um programa no modo passo a passo, ou logo após o término da execução. Nela, as linhas do programa são exibidas com seu número e conteúdo, e número de vezes que foram executadas, conforme se pode ver na Figura 6.

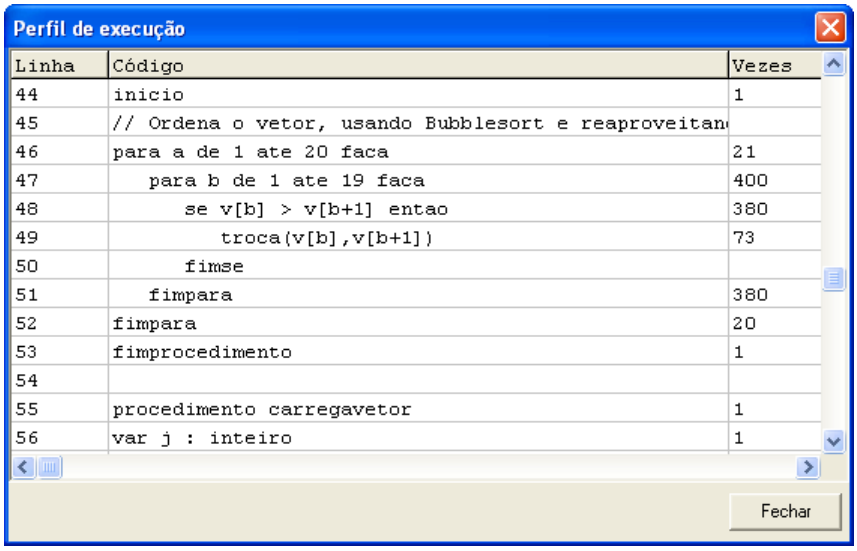

Figura 6. A ferramenta Perfil de Execução.

O VisuAlg no Laboratório de Programação

Pelo fato de ser totalmente escrito em português e de empregar um modelo intuitivo e bem conhecido da grande maioria dos alunos – o editor de textos ao estilo do bloco de notas – o **VisuAlg** (assim como o ILA e o AMBAP) pode ser empregado logo na primeira aula de programação, sem causar grande impacto. A experiência mostra que ele facilita o entendimento de como um programa de computador funciona, possibilita um *feedback* imediato sobre a correção e exatidão do pseudocódigo digitado, e o que se considera mais importante, instiga o aluno a experimentar e ver o resultado de suas alterações imediatamente. Naturalmente este mesmo resultado pode ser conseguido com um compilador C ou Pascal, mas acredita-se que com maior esforço por parte dos alunos, por causa de fatores como a língua do IDE e a rigidez das linguagens de programação, entre outros.

É um fato digno de nota, no entanto, que mesmo nas turmas que utilizam o **VisuAlg**, alguns alunos progridem normalmente sem usá-lo, e outros fazem questão de desenvolver seus algoritmos no caderno primeiro e só então os passam para o programa, como uma etapa de teste e validação. Estes fatos indicam que há maneiras diferentes de se aprender programação, e que possivelmente ferramentas de apoio de diversos tipos são necessárias, cada uma se adequando a um estilo individual de aprendizagem.

Conclusão

As dificuldades enfrentadas no início do aprendizado de programação e a própria característica da disciplina indicam que o ensino convencional deve ser deixado de lado e que o uso de ferramentas de auxílio ao aprendizado deve ser estudado e difundido para que o rendimento dos alunos seja melhorado. Neste contexto, desenvolveu-se o **VisuAlg** como uma opção mais próxima daquilo que um programador encontra em seu trabalho, ao mesmo tempo provendo ferramentas específicas para o uso no ambiente de aprendizado.

O uso deste *software* nos estágios iniciais do ensino de programação tem-se mostrado bastante produtivo por permitir que desde o início os estudantes tenham

contato com um ambiente de desenvolvimento próximo ao que encontrarão em sua vida profissional, embora mais simples e usando uma linguagem de programação sem tantos recursos. Ambientes como o **VisuAlg** promovem a experimentação e permitem desde cedo o entendimento do funcionamento de um programa de computador. Koliver, Dorneles e Casa (2004) relatam que há indicações de que o uso do **VisuAlg** (ou ferramenta similar) pode melhorar o desempenho dos estudantes, além de ser um fator motivador.

Acredita-se que estudos mais aprofundados devam ser realizados, em um universo mais amplo de estudantes, no qual o uso ou não do **VisuAlg** seja a única ou uma das únicas variáveis, já que é sabido que o conhecimento e didática do professor e o conteúdo apresentado também influenciam sobremaneira o resultado do processo.

Referências Bibliográficas

Koliver, C., Dorneles, R. V., Casa, M. E. 2004. Das (Muitas) Dúvidas e (Poucas) Certezas do Ensino de Algoritmos. In: XII Workshop de Educação em Computação (WEI 2004), 2004, Salvador. Anais do XXIV Congresso da Sociedade Brasileira de Computação. p. 949-960.

Gomes, A., Henriques, J., Mendes, A. J. 2008. Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. In: Educação, Formação & Tecnologias; v.1. p. 93-103.

Crespo, S. 1990. ILA – Interpretador de Linguagem Algorítmica. Endereço: http://www.exatec.unisinos.br/professores/gerador.php?professor=crespo&id_menu=434&id_submenu=189 Acesso em 21/02/2009.

Almeida, E.S. 2001. AMBAP - Ambiente de Aprendizado de Programação. endereço: http://www.ufal.br/tci/ambap/ Acesso em 21/02/2009.

Rapkiewicz, C. E., Falkemback, G., Seixas, L., dos Santos, N. S., Cunha, V. V., Klemann, M. 2006. Estratégias Pedagógicas no Ensino de Algoritmos e Programação Associadas ao Uso de Jogos Educacionais. RENOTE. Revista Novas Tecnologias na Educação, v. 4, n. 2, p.1-11, 2006.

Gondim, H. W. A. S., Ambrosio, A. P. L. 2008. Esboço de Fluxogramas no Ensino de Algoritmos. In: WEI - Workshop sobre Educação em Computação, 2008, Belém do Pará - PA. Anais do WEI 2008. p. 109-117.

Mota, M. P., Pereira, L. W. K., Favero, E. L. 2008. JavaTool: Uma Ferramenta para o Ensino de Programação. In: Congresso da Sociedade Brasileira de Computação, 2008, Belém. XXVIII Congresso da Sociedade Brasileira de Computação. p. 127-136.

Souza, C.M., 2003. Referência da Linguagem de Programação do VisuAlg. endereço: http://www.apoioinformatica.inf.br/visualg/refer.htm. Acesso em 21/02/2009.

_____, C.M., 2007. As Funções do VisuAlg Versão 2.0. endereço: http://www.apoioinformatica.inf.br/visualg/funcoes.htm. Acesso em 21/02/2009.