Professor, muito obrigada pelo material enviado. Ele foi muito útil e respondeu boa parte das minhas dúvidas. Além do que já foi comentado, eu gostaria de saber:

1. Seu método resolve vários problemas que os alunos enfrentavam no aprendizado da programação. Como você identificou estes problemas?

**Resposta abaixo (muito longa)**

2. Quanto tempo você levou para desenvolver o Portugol? E o Visualg 3.0? Eles continuam sendo desenvolvidos e aperfeiçoados?

**Resposta abaixo (3 perguntas, 3 respostas longas)**

3. Você comenta que o Visualg 3.0 facilita o entendimento de conceitos obscurecidos pelas linguagens de programação. Poderia dar um exemplo disso na prática?

**Resposta abaixo**

4. Na sua opinião, qual é o principal diferencial do método de ensino que você desenvolveu?

R.***O fato de ser em português é o principal, mas existem vários outros recursos adotados, como ver o conteúdo das variáveis de memória durante a sua execução seria outro, ou é fácil e simples de se usar, não precisa instalar no micro “ele roda em um pendrive”, tem ajuda online em português, e estou sempre atualizando ele.***

5. Nos últimos anos, muitas escolas no Brasil e no mundo acrescentaram aulas de programação em seus currículos, e alguns nomes importantes da área têm defendido a importância do aprendizado da programação mesmo para pessoas que não irão trabalhar na área. O que você pensa sobre o assunto? Nos dias atuais, qual é a importância de aprender programação e quem deveria aprendê-la?

**R. Primeira parte da pergunta:**

**Na verdade isso já está acontecendo a muitos anos em cursos de engenharia (civil, mecânica, elétrica, etc) e não só para cursos de informática.**

**Segunda parte da pergunta:**

**Bom eu digo o seguinte muitos falam e criar aplicativos para celulares, o que facilita muito a vida dos usuários em geral, assim alguém tem que criar os aplicativos, e alguém tem que ensiná-los a criar.**

6. Qual seria a sua dica para alguém que deseja aprender programação, mas pensa não ser capaz por achar que a programação é algo muito difícil?

**R. Inicialmente procurar uma boa escola técnica tipo (exemplo CEDUP, IF), depois baixar o Visualg 3.0 do site oficial :** [**http://visualg3.com.br/**](http://visualg3.com.br/)

**ou no Source Forge:** [**https://sourceforge.net/projects/visualg30/**](https://sourceforge.net/projects/visualg30/)

**Depois instalar (copiar-la) para o seu computador e começar a fazer os testes que o professor passar na escola. Aí já é um começo.**

**Mas, na internet (google) há muitas apostilas prontas que ensinam a montar e no meu site também. Qualquer coisa me contatem pelo meu e-mail:** [professor.antonio.nicolodi@gmail.com](mailto:professor.antonio.nicolodi@gmail.com)

= = = = = = = = = = = = = = = = = = =

Ok! é uma história longa, mas vamos lá....

Antes de responder vou explicar alguns conceitos:

Até meados de 1985 quando alguém ia fazer um curso de informática em Blumenau e região (vale), era pra fazer um curso de programação COBOL, dado pela Cetil que depois virou CETELBRAS.

A empresa chamada **Cetil Informática** foi a empresa precursora da informática em **Blumenau/SC e nossa região do vale do Itajaí** e como tal desenvolveu todo uma logística: com pequenas empresas de desenvolvimentos de sistemas, treinamentos, manutenções (infra), redes, internet, etc.

Os clientes que existiam, sintião que faltavam mão-de-obra qualificada (programadores, analistas de sistemas, técnicos de manutenção, operadores, digitadores, professores, instrutores, etc) e esses profissionais eram formados pela FURB Faculdade de BLUMENAU/SC.

E todos os cursos avulsos (não técnicos, nem de graduação) eram voltados para as necessidades do momento (da época 1980-1990).

A falta de programadores no mercado era grande, pois mais e mais empresas, estavam se informatizando e a **Cetil** só atendia as grandes empresas, como a **Cia Hering** com mais de 5mil funcionários, a **Teka** com mais de 3mil funcionários, a **Karsten**, a **Sulfabril**, etc

E as pequenas empresas com menos de 500 funcionários já ficavam quase sem suporte, e as empresas com menos de 50 funcionários aí então que era muito complicado.

- Assim nasceram os Bureaux (birôs) de prestação de serviços, empresas que desenvolviam pequenos programas para as áreas de contabilidade, ERP (sistemas de vendas), etc.

E todos os cursos eram voltados para formar programadores, uma função de altos salários (chegando à quase 10 mil dólares/mês). O que gerava um grande interesse nessa nova área.

E quem não conhecia nada da área e queria aprender tinha que naturalmente fazer um curso de informática (programação).

Para se programar um computador um programador tem que possuir alguns conceitos (algum conhecimento) básico necessário. Assim sempre se parte de um problema proposto:

1. Todo problema gera um processo, e como tal todo processo tem:

Inicio – Meio – Fim, e o programador é quem deve montar desvendar esse processo através da lógica etapa por etapa e depois montar o programa pra resolvê-lo.

1. A partir do problema proposto: saber destrinchar ele, e/ou retirar informações que lhes serão úteis. O programador tem responder algumas perguntas como:

* Quais as informações entrarão no processo (manualmente por digitação ou automaticamente por calculo) o que ele nos pede;
* Quais são as variáveis do problema;
* Tem alguma formula pronta ou terei que construir uma;
* O que deverá sair na tela do monitor ou qual a informação (resposta) será apresentado na tela.

1. Conhecer lógica de programação (LÓGICA) para montar o algoritmo corretamente com a ordem das etapas da resolução do problema proposto. O quê deve se fazer primeiro, e depois, e depois, ... , até finalizar a tarefa.
2. Conhecer então uma linguagem de programação, seus comandos/funções e (a sintaxe e a semântica) de cada um desses comandos, para colocar eles na mesma posição correta que o algoritmo pede.

**Primeira resposta:**

Estes problemas foram detectados “in loco” em sala de aula mesmo, aonde eu percebi que ao longo do tempo todos os alunos mesmo os mais interessados tinham os mesmos problemas de entender os conceitos básicos, e apesar de eu explicar várias vezes “a mesma coisa” o mesmo assunto. Pois muitos alunos eram ainda jovens (entre 10 e 18 anos) e nunca tinham trabalhado fora e assim não conseguiam entender, pois lhes faltavam à experiência do trabalho e da luta diária (da batalha) e a necessidade de certas experiências que iriam usar mais pra frente.

Detectei, alguns problemas e um deles era o fato de que a maioria das linguagens de programação, “eram ou ainda são” em INGLÊS (99,99%), pois foram feitas pelos americanos e ingleses.

E os alunos se preocupavam muito em tentar entender o significado das palavras antes de se preocuparem em resolver a lógica em si. O que era um grande problema ou uma grande barreira no Brasil.

Porquê? Muitos brasileiros não gostam muito do inglês e nem dos termos técnicos em inglês e se assustam quando tem que montar o próprio algoritmo "ou um programa de computador" todo em inglês.

E o pior é que a maioria das documentações, exemplos, exercícios, etc, que acompanham essas linguagens são também em inglês. E quando há livros, estes, são mau traduzidos, pois trazem anomalias estranhas.

Outro problema é que a maioria dos professores que ensinam a disciplina chamada **lógica de programação** para os seus alunos “iniciantes”, usam os fluxogramas (diagramas gráficos) e logo em seguida vão diretamente para uma linguagem de programação adotada pelo professor como Pascal, C, Java, pois estes aprenderam dessa forma na sua época. Que eram por métodos mais antigos e por isso só ensinam "do mesmo jeito que aprenderam", e só sabem ou só transmitem do mesmo modo que absorveram.

Eu também aprendi assim, a lógica era fundamentada em cima da estrutura de máquinas de grande porte (MAINFRAMES) e em cima de conceitos puramente matemáticos. Mas, eu evolui com os meus erros, e tive doutrinar meus conceitos.

Mas quê na nossa época, nós, alunos também tínhamos as mesmas dúvidas e as mesmas dificuldades, e por isso tínhamos que obter todas as informações através de livros, artigos, revistas, etc. Vivíamos dentro das bibliotecas sempre pra nos aprimorarmos, em nossos conhecimentos, “bom, era isso ou nada!“.

E eu tive que aprender desse jeito, assim também de forma complexa e com muita dificuldade. E vi muitos de meus colegas desistirem por não conseguir entender nada da disciplina conhecida como: Lógica.

Nós não tínhamos os recursos que existem hoje como a internet banda larga 24horas ligada e com o "google", assim éramos verdadeiros (ratos de bibliotecas) ou como alguns nos chamavam NERD´s.

Na época, nos meados de 1980-85, os computadores profissionais eram só os **de grande porte**, e os micros eram mais um brinquedo, do que uma ferramenta de trabalho, nós nos baseávamos em resolver os problemas (como formulas matemáticas, físicas, etc) por isso tínhamos que pesquisar muito.

**O PROBLEMA:**

Os alunos "jovens” de hoje já são mais lentos e não tem muita paciência pra estudar, pesquisar, ler, etc. E não querem fazer grandes pesquisas com trabalhos com mais de 200 páginas, hoje eles querem tudo mastigado, tudo prontinho na palma da mão ou desistem fácil e a evasão é muito grande, mais de 96 % dos alunos não terminam os cursos iniciados.

E ainda tem muitos alunos iniciantes que aprendem no método antigo, diretamente na própria linguagem de programação, mas poucos absorvem, e quando esbarram em um problema de lógica, se apavoram e desistem de programar ou perdem muito tempo em cima de um só problema.

A disciplina conhecida como: **Lógica** ainda é a vilã dos cursos de programação de computadores, pois geram uma grande evasão de alunos, que trocam de cursos ou apenas desistem.

E, como eu leciono **LÓGICA** para evitar essa grande evasão dos cursos e das escolas eu preocupado com isso criei na época (1980-83) esse método foi pra ajudar aos meus alunos de cursinhos de informática que encaravam esse mesmo dilema, de não entender inglês e também não conheciam tanto de matemática, eletrônica, física, etc. Mas que deu tão certo, que venho aplicando até hoje, e muitos autores de livros usaram para criar os seus livros, e como eu ainda uso esse método "***in loco***", para muitas escolas como no CEDUPHH, UNIASSELVI, e outras. digo que é um método que funciona.

**Segunda pergunta:**

Para criar o Portugol levei só uns 02 (dois) meses pois eu me baseei numa linguagem de programação antiga chamada ALGOL, que é estruturada em blocos de comandos: (https://pt.wikipedia.org/wiki/ALGOL) , que era muito simples e que deu origem ao nome **Algoritmo**.

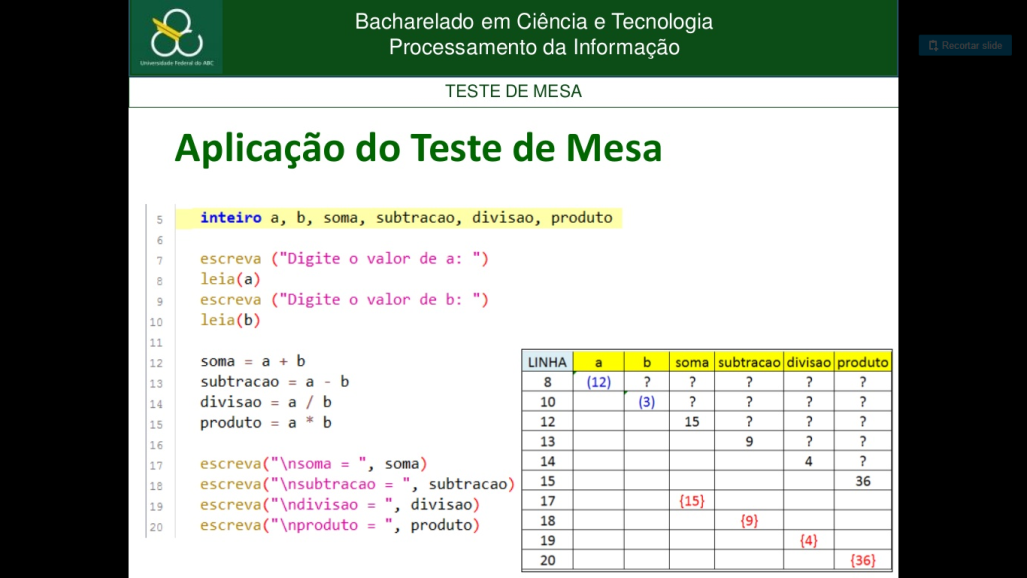
Eu apenas a traduzi para o português, e adotei a idéia da sua estrutura do ALGOL, e o seu formato dos comandos da linguagem PASCAL que é uma descendente do ALGOL.

Assim era uma mistura de **Algoritmo** escrito em português, mas com o formato e as regras do Pascal, e o seu nome é : PORTUGOL

Onde **PORTU***guês* +  *al***GO***l* + *Pasca***L**, também é conhecido como **português estruturado**, por causa de sua estrutura de blocos modulares.

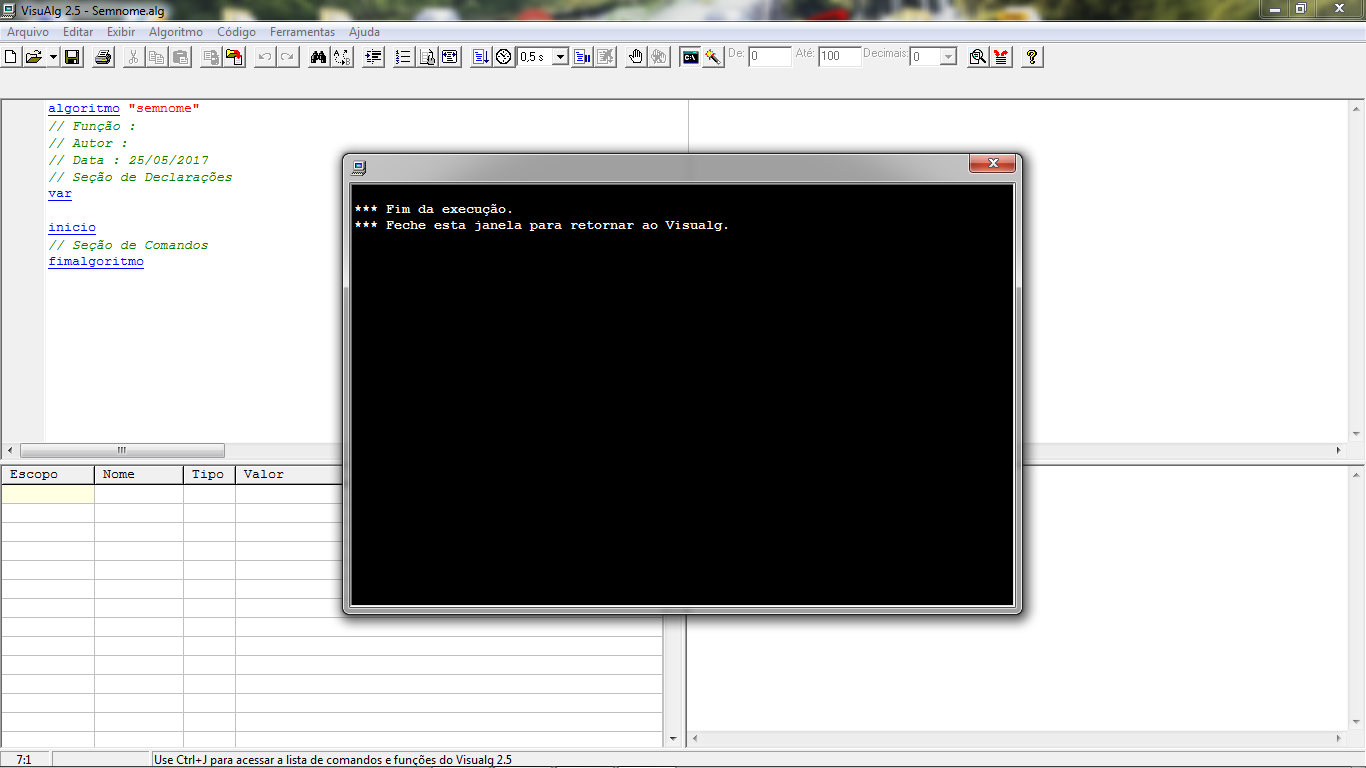
Os comandos, procedimentos e funções, as suas regras e referências são orientadas do mesmo jeito da linguagem PASCAL (micros), que na época era linguagem mais usada nas Universidades junto com a linguagem de programação Cobol (usada só em MAINFRAMES), e assim conforme é na linguagem Pascal: (https://pt.wikipedia.org/wiki/Pascal\_(linguagem\_de\_programação)

Bom, mas para usar o PORTUGOL precisava de um interpretador e assim, pois os algoritmos eram escritos no caderno e testado no próprio caderno (usando a imaginação do aluno) aliado à uma técnica chamada TESTE DE MESA, aonde os alunos montam uma tabela com as variáveis usadas dentro do algoritmo e a cada passo do processo, eles iriam alterando os seus valores.



O aluno tinha que escrever o algoritmo e interpretar os comandos e testar cada uma das variáveis passo a passo, antes de transcrever esse algoritmo para uma linguagem de programação como o PASCAL para daí testar se estava funcionando, tudo de cabeça.

A HISTÓRIA DO VISUALG :



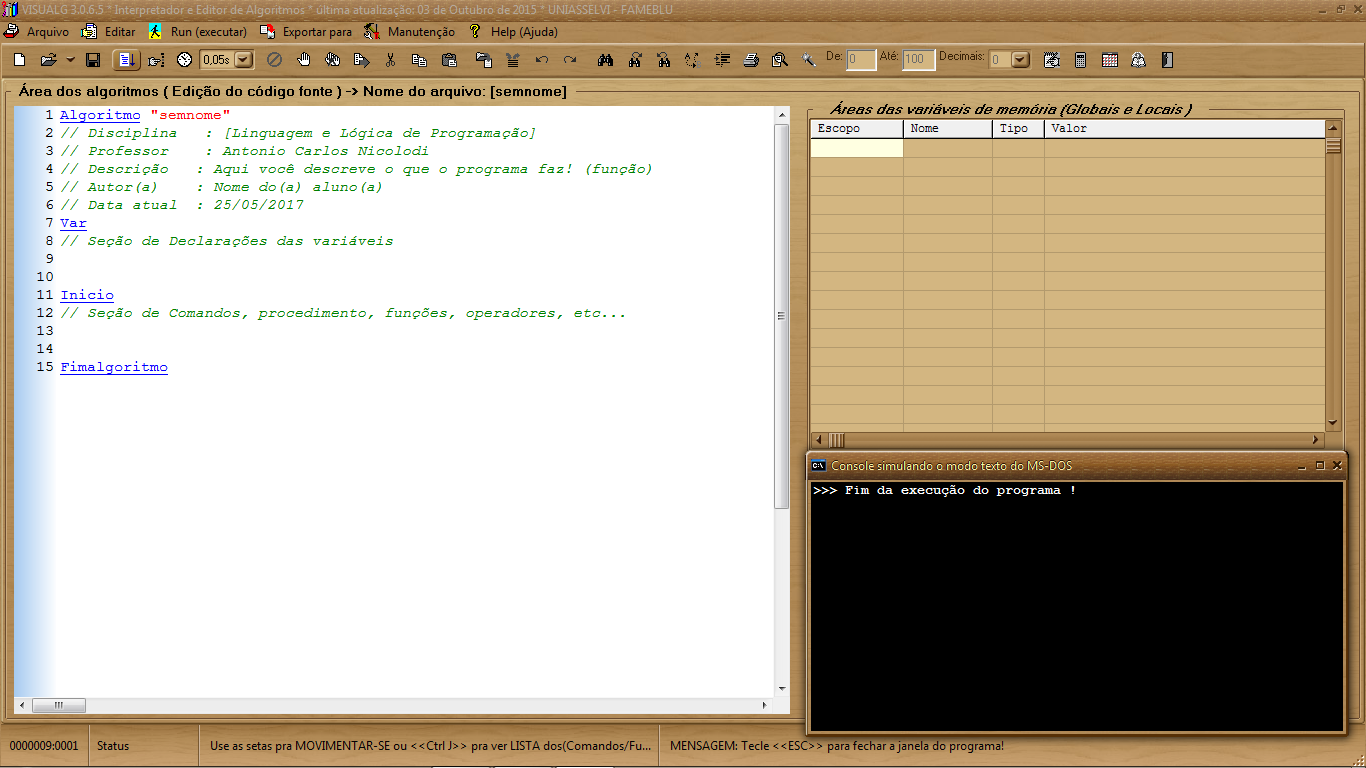
Bem, tudo começou com o professor Cláudio Morgado de Souza que criou o Visualg (**Visu***al* **Alg***oritmo*) em 1996 para um curso de graduação que fez e usou para interpretar os algoritmos e mostrando também as variáveis

Quando nos conhecemos ele já usava o meu método PORTUGOL e ao saber disso e por eu ser um programador da mesma linguagem de programação que ele também programava, o prof. Cláudio convidou-me para participar do seu projeto e me passou os códigos fontes do programa da versão 2.0 para mim do **VISUALG** e para ajuda-lo a melhorar o programa e assim eu o fiz.

Mas, por questões pessoais o professor Cláudio, teve que abandonar o seu projeto e me pediu para continuá-lo e substituí-lo, em seu projeto e continuar a alterar, melhorar e desenvolver novas versões com ele, e me contou que estava trabalhando numa nova versão a 2.5 e foi lançada, mas que poderia ter alguns erros ainda, e eu apenas dei a manutenção nela "2.5", e lancei a versão 2.6 do Visualg e dediquei a minha vida em outros assuntos, pois eu estava lecionando outras disciplinas também, na época.

Em 2014, quando voltei a lecionar novamente **LÓGICA** no **CEDUPHH** e na **UNIASSELVI-FAMEBLU** (ambos em Blumenau/SC), resolvi então, mexer novamente no programa **Visualg,** e vi que precisava de muitos ajustes, então comecei a "criar" montar a versão 3.0 e a lancei em 2014, com um novo leiaute, e um monte de novos comandos e novos recursos e nela tanto os alunos como os professores podem ver como cada linha do algoritmo é executada, o quê está acontecendo na tela, e nas variáveis de memória.

O VISUALG 3.0



Com o **Visualg 3.0** não havia mais a necessidade de se criar um **TESTE DE MESA** no caderno, pois o aluno vê na tela do computador as variáveis que estão sendo usadas dentro do próprio algoritmo e pode acompanhá-las *PASSO A PASSO* na execução desse algoritmo, além de ver o que está sendo mostrado na simulação da tela do MS-DOS (telinha preta).

E na versão 3.0 o aluno ainda pode mover essa simulação do MS-DOS, em qualquer lugar da tela do computador.

E em breve estarei lançando a versão 3.5 com muitas novidades também.

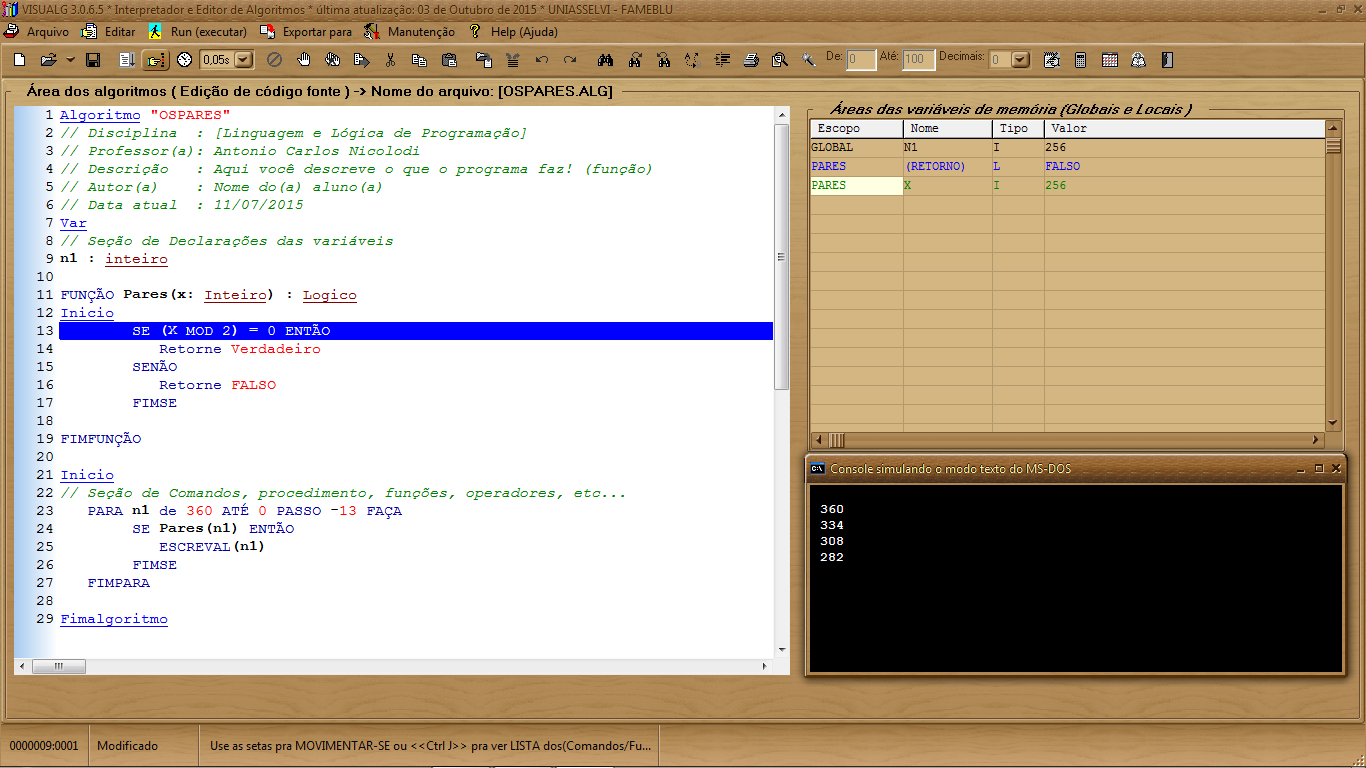
**Terceira pergunta:**

Como foi comentado antes, com o **Visualg 3.0** o aluno não necessita de um teste de mesa escrito num papel ou imaginar mentalmente, ou também não precisa mais imaginar “aonde ou o quê” cada variável está guardando.

Ou “o quê” um determinado comando está executando neste instante, pois o próprio **Visualg 3.0** mostra isso ao aluno, ele aprende vendo o que está acontecendo dentro da memória, e dentro do algoritmo.

Como eu digo o aluno aprende brincando !

Com o recurso da execução PASSO A PASSO como mostra a figura abaixo o aluno vê a linha do algoritmo que está sendo executada, vê as variáveis de memória e vê o que é mostrado na tela do monitor (o resultado), antes mesmo de ter uma codificação em uma linguagem de programação propriamente dita, ou um programa (aplicativo) rodando, é só a Lógica de como o programa irá funcionar. Como mostra na figura abaixo e que a **barra azul** está executando uma linha.



**FUNCIONA ASSIM:**

**Exemplo: (UM PROBLEMA É DADO PELO PROFESSOR).**

Um caracol caiu em um poço de 10 metros de profundidade, e ao subir o caracol, consegue subir apenas 2 metros para cima por dia, mas a noite ao dormir ele escorrega meio metro de volta pra baixo. Pergunta, quantos dias o CARACOL irá levar para subir e sair do poço?

- Monte um programa que calcule quantos dias o caracol irá levar para sair do poço.

BEM O ALUNO COMEÇA AQUI:

Para montar um programa de computador que executa e resolva este simples problema, procede-se assim:

**Primeiro:** Antes de tudo

Vamos ver o problema e definir o que ele pede e o quê ele nos oferece, e :

* definir as variáveis (nome e tipo);
* criar também as fórmulas;
* ver o que deve ser apresentado na tela.

(nada disso o professor fala, o aluno que tem que saber sozinho)

**Segundo:**

Montar uma lógica com o passo a passo do algoritmo:

usando o algoritmo descritivo (passo a passo)

um fluxograma

e a pseudo-linguagem (ou portugol)

**Terceiro:**

Testar o fluxograma contendo o fluxo dos dados, com um TESTE DE MESA.

**Quarto:**

Fazer o TESTE DE MESA para testar e ver se está correto

**Quinto:**

Transformar este algoritmo em código da linguagem de programação **(Pascal)**

**Sexto:**

Executar o programa em PASCAL

**Sétimo:**

Apresentar o resultado ao professor.

**Só isso...**

Bem primeiro, identificamos as variáveis do problema:

**Sobe, Desce** *serão do tipo Real (poderão receber valores c/ casas decimais)*

**Dias, Total** *serão do tipo* *Inteiro*

Desce = 0,50

Sobe = 0

Total = 10

DIA (1..N)

Definir a Formula: Sobe = (Sobe + 2) - Desce // pra cada dia

// ===============================================

VOU RESUMIR COM ESSE ALGORITMO ABAIXO:

Algoritmo "Caracol"

// Disciplina: [Linguagem e Lógica de programaçao]

// Professor: Antonio Carlos Nicolodi

// Função : Calcula os passos de um caracol para subir o poço

// Autor : Professor Antonio

// Data : 05/04/2015

Var

Sobe, Desce: Real

Dias, Total : Inteiro

Inicio

Sobe <- 0.0

Desce <- 0.5

Dias <- 0

Total <- 10

Limpatela

Escreval("Cálcular os passos de um Caracol")

Escreval("================================")

Escreval(" ")

Para Dias de 0 ate Total faca

se Sobe <= Total entao

Sobe <- Sobe + 2 - Desce

Escreval("Sobe = ",Sobe)

senao

interrompa // pare de executar o algoritmo

fimse

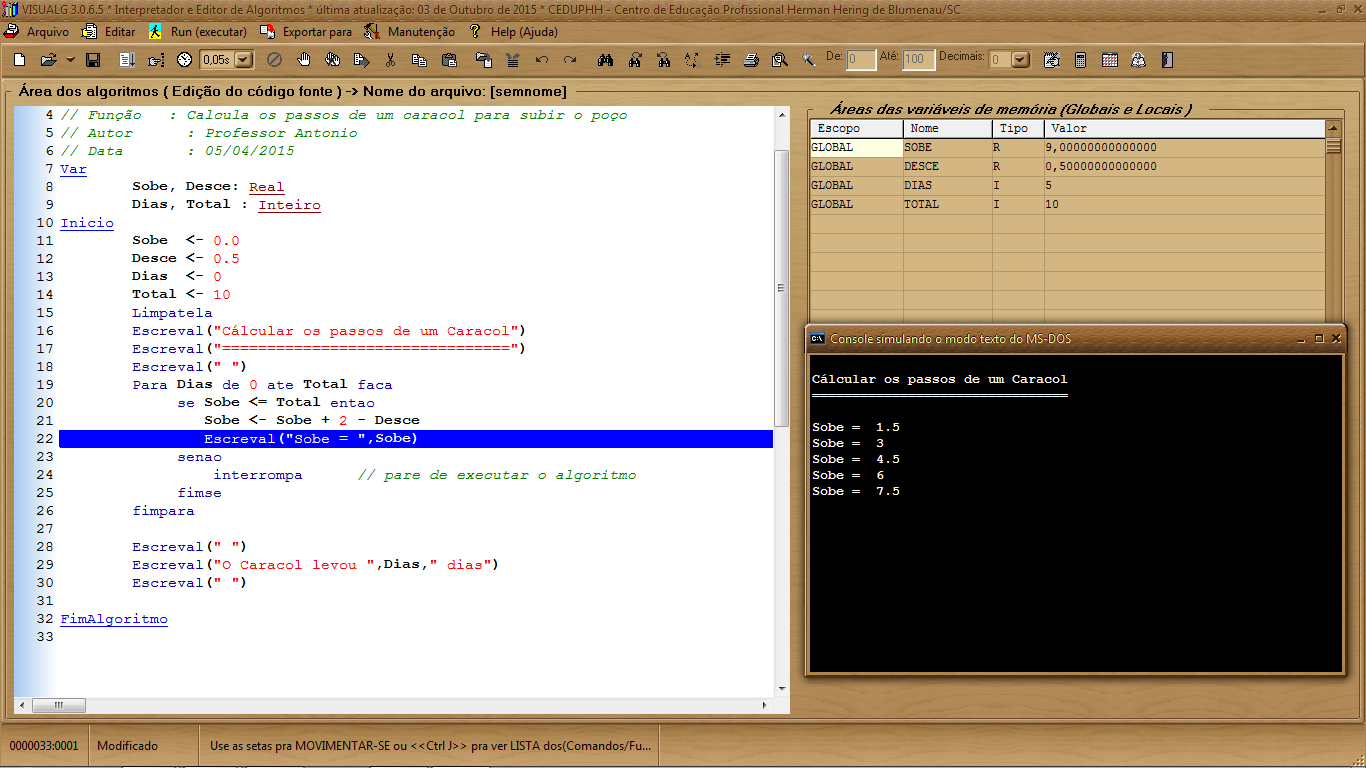
fimpara

Escreval(" ")

Escreval("O Caracol levou ",Dias," dias")

Escreval(" ")

FimAlgoritmo



= = = = = = = = = = = = =

Aqui já está o programa escrito em PASCAL:

program Caracol;

uses Crt;

// { Disciplina: [Linguagem e Lógica de programaçao]}

// { Professor: Antonio Carlos Nicolodi}

// { Função : Calcula os passos de um caracol para subir o poço}

// { Autor : Professor Antonio}

// { Data : 05/04/2015}

Var

Sobe, Desce: real;

Dias, Total : integer;

begin

Sobe := 0.0;

Desce := 0.5;

Dias := 0;

Total := 10;

ClrScr;

writeln('Cálcular os passos de um Caracol');

writeln('================================');

writeln(' ');

for Dias := 0 to Total do begin

if Sobe <= Total then begin

Sobe := Sobe + 2 - Desce;

writeln('Sobe = ',Sobe);

end else begin

Exit // { pare de executar o algoritmo};

end;

end;

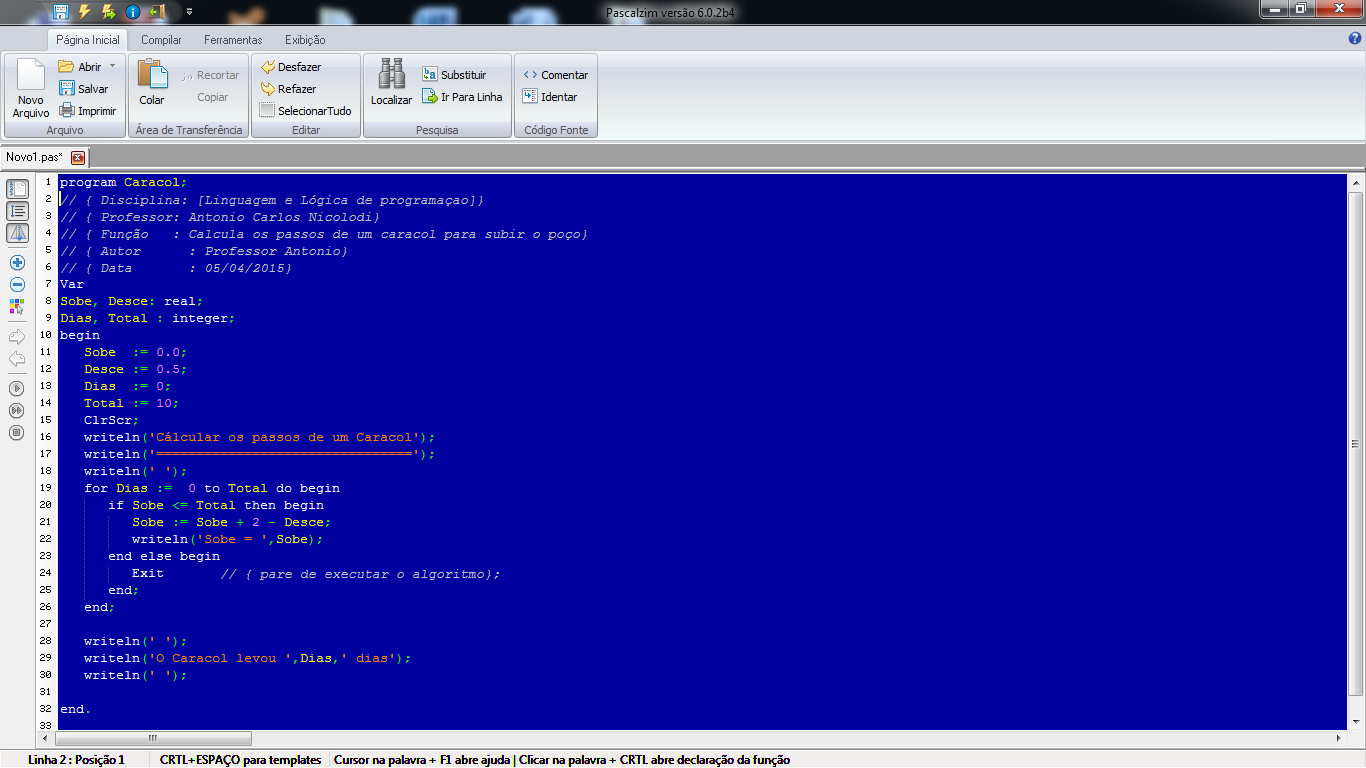
writeln(' ');

writeln('O Caracol levou ',Dias,' dias');

writeln(' ');

end.

Programa executando no PASCALZIM (um compilador Pascal)



Só Isso !