**Aula 4 – Tipos de Dados**

**Apresentação**

Na Aula 3, você viu os conceitos referentes às três principais formas de representação de algoritmos: descrição narrativa, fluxograma e pseudocódigo. Você conheceu as principais vantagens e desvantagens de cada uma dessas formas de representação e viu exemplos de construção de algoritmos nas três formas. Ficou definido, também, que o pseudocódigo vai ser a forma mais utilizada para construir algoritmos ao longo de nossas aulas.

Por fim, serão propostos alguns exercícios. Ao longo desta terceira aula, você aprenderá como ocorre o processo de armazenamento de dados na memória do computador, conhecerá o conceito de variáveis e constantes e a maneira de utilizá-las nos algoritmos. Irá aprender também a reconhecer os tipos de dados que podem ser utilizados, percebendo a importância de sua definição na construção dos algoritmos.

**Objetivos**

Ao final desta aula, você será capaz de:

* Descrever o processo de armazenamento de dados na memória do computador.
* Definir o conceito de variável e constante, identificando as regras básicas para sua criação e utilização nos algoritmos.
* Justificar a importância da definição dos tipos de dados na construção dos algoritmos.
* Identificar os tipos de dados utilizados.

### Armazenamento de dados na memória

Você viu, nas aulas anteriores, que um algoritmo recebe dados de entrada e, ao final da execução, retorna uma ou mais saídas. No exemplo da receita de bolo, você pôde ver que os ingredientes são os dados de entrada e a saída é o bolo pronto.

Na execução de um algoritmo ou de um programa, para receber dados de entrada, processar as instruções e fornecer os dados de saída, é necessário que o computador armazene esses dados e instruções em sua memória, a fim de poder acessá-los posteriormente.

Por exemplo, imagine que você e um amigo desejam calcular o algoritmo da média, "de cabeça‟, sem fazer anotações. Veja os passos descritos a seguir.

1. Você diz a primeira nota.  
2. Seu amigo, então, guarda esse valor na sua memória humana, enquanto aguarda a segunda nota.  
3. Você diz a segunda nota.  
4. Seu amigo guarda o valor da segunda nota em outro local da memória diferente do anterior, para não sobrescrever (e acabar esquecendo) a primeira nota.  
5. O seu amigo recupera os valores das notas em sua memória, faz o cálculo da média e coloca o valor encontrado em um terceiro local de sua memória.  
6. Você pergunta: "Qual o valor da média?"  
7. O seu amigo informa o resultado obtido.

De forma similar, o computador precisa guardar os dados e instruções em endereços específicos de sua memória, durante a execução de um algoritmo, a fim de não "esquecer‟ as informações, podendo, assim, processá-las de maneira eficaz.

### Definição e utilização de variáveis

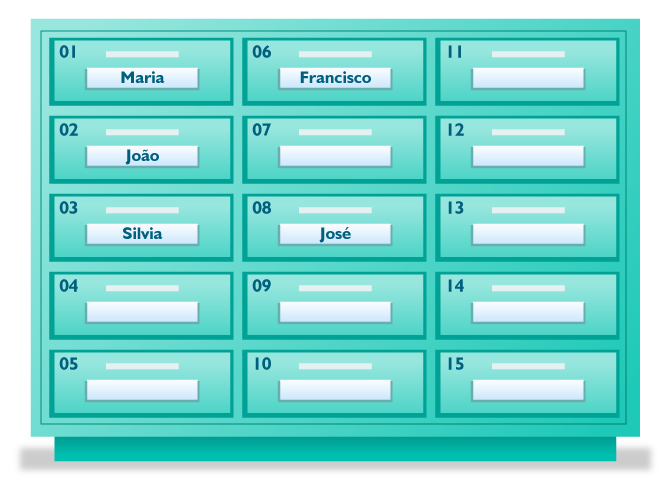
Agora você já sabe que os dados utilizados nos algoritmos são armazenados na memória do computador para serem posteriormente acessados.

Mas de que forma conseguimos acessar esses dados?

Para acessá-los, precisamos descobrir em que posição na memória do computador eles estão armazenados. O conceito de variável foi criado para facilitar essa busca.

Vamos compreender de que forma isso funciona?

Imagine um grande arquivo com várias gavetas. Para conseguirmos acessar o conteúdo dessas gavetas, é importante que o armazenamento seja feito de maneira organizada, de forma que possamos encontrar facilmente o que procuramos. Para isso, seria interessante identificar as gavetas com nomes e/ou números. Cada gaveta identificada representa, dessa forma, o endereço do conteúdo que ela armazena. Observe a Figura 1.



**Figura 1 -** Arquivo de gavetas representando espaços reservados na memória de um computador

Na Figura 1, está representado um arquivo com 15 gavetas que podem armazenar os documentos de diferentes pessoas. Veja que as gavetas estão numeradas e cada gaveta só armazena um documento por vez. Cada número corresponde ao "endereço‟ da gaveta. Algumas delas estão, também, marcadas com nomes de pessoas. Essas são as que estão reservadas.

Por exemplo, a gaveta de número 01 está reservada para armazenar um documento de Maria. Ninguém além de Maria poderá colocar seus documentos na gaveta 01. Veja que tem gavetas reservadas também para João, Sílvia, Francisco e José. E há gavetas que não foram reservadas, ou seja, estão livres para serem reservadas quando necessário (essas gavetas, porém, só podem ser utilizadas se forem previamente reservadas). Dessa forma, se perguntarmos a Maria onde ela guarda os seus documentos, ela responderá: "na gaveta 01 do arquivo". É o "endereço‟ dos documentos dela.

O endereço ao qual nos referimos, em algoritmos, corresponde a uma posição na memória do computador. A essa posição na memória do computador damos o nome de variável.

Uma variável é uma posição na memória do computador, que é reservada para armazenar os dados que o algoritmo vai manipular.

Uma variável precisa ter um nome (ou identificador), um tipo de dado associado a ela (tipo da variável) e a informação que ela armazena. O identificador serve para diferenciar a variável das demais, por isso deve ser único para cada variável.

O processo de criação de uma variável é chamado de declaração da variável. As variáveis devem ser declaradas no algoritmo antes de serem utilizadas, pois a declaração das variáveis permite que seja reservado um espaço na memória para o dado que vai ser armazenado e utilizado.

Para declarar uma variável, devemos proceder da seguinte forma:

<nome\_da\_variável> : <tipo\_da\_variável>

ou:

<lista\_de\_variáveis> : <tipo\_da\_variável>

|  |  |
| --- | --- |
| I – nota | **: real** |
|  |  |
| nome da variável | : tipo da variável |
|  |  |
| II – idade | **: inteiro** |
|  |  |
| nome da variável | : tipo da variável |
|  |  |
| III – sexo | **: caracter** |
|  |  |
| nome da variável | : tipo da variável |
|  |  |
| IV – frase | **: literal** |
|  |  |
| nome da variável | : tipo da variável |
|  |  |
| V – nota1, nota2 | **: real** |
|  |  |
| nome da variável | : tipo da variável |

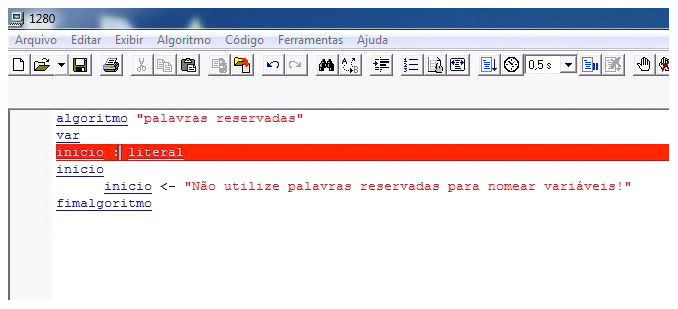
Veja, nos exemplos, que declaramos as variáveis "vazias", isto é, sem informações associadas a elas (e, então, fornecemos essas informações posteriormente, no corpo do algoritmo). Lembre-se do exemplo das gavetas. Para reservar uma gaveta para uma pessoa, não era necessário armazenar o conteúdo, de imediato, na gaveta. Igualmente, só é possível armazenar conteúdo em uma gaveta que já foi previamente reservada. Ou seja, primeiro deve-se reservar uma gaveta e depois armazenar seu conteúdo.

Veja, também, no exemplo V, que duas ou mais variáveis de um mesmo tipo podem ser declaradas juntas, na mesma linha, separadas por vírgula (lista de variáveis).

Uma variável pode assumir valores diferentes, mas só pode armazenar um valor a cada instante. Além disso, o seu conteúdo pode mudar ao longo do tempo na execução do algoritmo.

Existem algumas regras básicas para a definição dos nomes (identificadores) das variáveis. Vamos conhecê-las.

1. Podem-se utilizar números e letras maiúsculas ou minúsculas.  
2. Não se deve utilizar caracteres especiais, à exceção do caractere sublinhado, ou underline ( \_ ).  
3. O primeiro caractere deve ser sempre letra ou sublinhado, mas nunca um número.  
4. Não deve utilizar acentos gráficos, til ou cedilha.  
5. Além dos símbolos, não é permitido espaço em branco.  
6. Não se devem utilizar as palavras reservadas da linguagem (observe a Figura 1).



**Figura 2 -** Comportamento do VisuAlg durante a tentativa de nomear uma variável com uma palavra reservada

O termo inicio, no VisuAlg (Figura 2), é utilizado para começar o algoritmo. Ao tentar identificar uma variável com o nome inicio, a resposta do programa é o encerramento da execução e a linha em que está o erro é realçada em vermelho.

Vejamos alguns exemplos de acertos e erros na identificação de variáveis:

Identificadores válidos:

a. nome1  
b. ano\_de\_nascimento  
c. salario  
d. nota\_aluno  
e. qtd\_dias

Identificadores inválidos:

a. 1nome – não deve começar por número.  
b. Ano de nascimento – não deve ter espaço em branco.  
c. @salário$ – não deve conter caracteres especiais (contém @ e $).  
d. Nota\*aluno/01 – não deve conter caracteres especiais (contém \* e /).  
e. fim – não devemos utilizar palavras reservadas.  
f. remédio – não deve utilizar acentos gráficos.

Obs.: 1: É importante que, ao identificar uma variável, você utilize palavras ou termos significativos. Por exemplo, para armazenar o valor do salário do mês janeiro, use salário\_janeiro; para armazenar um número de telefone, use num\_fone. Essa prática ajudará no entendimento do algoritmo (para quem lê), tornando-o autoexplicativo.

Obs.: 2: Apesar de não haver limitações para a quantidade de caracteres que comporão o nome de uma variável, não alongue desnecessariamente os nomes. Por exemplo: salario\_do\_mes\_de\_janeiro\_com\_desconto.

Obs.: 3: Ao utilizar nomes de variáveis com mais de uma palavra, separe as palavras com o caractere sublinhado. Por exemplo: salário\_do\_mes.

Obs.: 4: Ao declarar as variáveis de um algoritmo, primeiro analise o problema, depois defina todos os dados de entrada e saída e o tipo de cada um para, então, criar os identificadores. Isso o ajudará a se lembrar de declarar todas as variáveis necessárias à execução do algoritmo.

### Constantes

Além das variáveis, em algoritmos, também lidamos com constantes. Trata-se de valores fixos ou estáveis, que são escritos no programa de forma literal.

No VisuAlg, temos constantes numéricas, caracteres e lógicos.

São exemplos de constantes numéricas:

a. 15  
b. 250  
c. 38.5

Exemplos de constantes caracteres ou literais:

a. "F"  
b. "Maria"  
c. "Avenida São José"

Exemplos de constantes lógicos:

a. Verdadeiro  
b. Falso

Ao atribuir valores às variáveis, podemos utilizar outras variáveis, expressões ou constantes. Veja:

**Var**  
a : inteiro  
b : inteiro  
c : inteiro  
d : lógico  
f : literal  
**Inicio**  
a <- 2 //atribuição de constante  
b <- a + 1 //atribuição de expressão  
c <- b //atribuição de variável  
d <- falso //atribuição de constante  
f <- "Teste" //atribuição de constante  
**fimalgoritmo**

Lógica de Programação

C:\Users\1968604\Documents\IFRN\Metropoli Digital\interface\icone_atividade.jpg01

Esta atividade vai ajudá-lo a fixar o que foi estudado nesta primeira parte da aula. Com base na nossa aula, responda às questões a seguir.

1. Qual a importância da utilização das variáveis na criação de um algoritmo?

2. Por que é importante declarar uma variável antes de utilizá-la?

3. Por que o identificador de uma variável deve ser único?

4. Explique os problemas nos identificadores a seguir, sugerindo, em seguida, uma alternativa válida para cada um:

a. %unidades  
b. anos sem aumento  
c. ah!  
d. Nota\*do\*aluno  
e. Real  
f. Numero\_de\_vezes\_que\_fui\_ao\_shopping\_hoje

5. Explique com suas palavras o que é uma constante.

### Tipos de dados

Vamos relembrar o exemplo do arquivo. Vimos que cada gaveta tem um determinado conteúdo. Para que possamos acessar esse conteúdo, a gaveta precisa estar identificada com um "endereço‟ representado por um nome e/ou número de identificação.

Imagine agora que, no momento da identificação, seja associado à gaveta um determinado tipo de conteúdo. A partir desse momento, é criada uma regra: essa gaveta só pode armazenar o tipo de conteúdo ao qual ela foi associada. Você viu que, no momento da criação de uma variável, é associado a ela um tipo de dado. A partir desse momento (de forma similar ao que ocorre com a gaveta do nosso exemplo), a variável só pode armazenar o tipo de dado ao qual foi associada.

Os algoritmos lidam com o conceito de tipos de dados para diferenciar dados de naturezas distintas e, assim, saber que operações podem ser realizadas com eles.

Por exemplo, não faria sentido uma operação de soma entre os dados "ana‟ e "5‟. Assim, dependendo da natureza dos dados utilizados, algumas operações podem ou não fazer sentido. Os tipos de dados que um algoritmo pode manipular são: dados numéricos, dados literais e dados lógicos. Vamos conhecer cada tipo e compreender de que forma cada um deles pode ser utilizado.

#### Dados numéricos

No estudo da matemática, você aprendeu que existem diversos conjuntos numéricos (conjunto dos números naturais, inteiros, racionais, irracionais e reais). No estudo dos algoritmos, lidaremos apenas com os números inteiros e os números reais.

#### Números inteiros

Os números inteiros são os números positivos ou negativos que não possuem parte decimal ou fracionária. Ex.: 15, -487, 0, 27835, -14937, 100.

Em algoritmos, os números inteiros são utilizados geralmente para expressar valores, como quantidades ou idade.

Observe o exemplo a seguir:

ALGORITMO QUE CALCULA A IDADE DE UMA PESSOA, DADOS O ANO ATUAL E O ANO EM QUE ELA NASCEU.

**algoritmo** "calcula idade"  
**var**  
ano\_atual : **inteiro**  
ano : **inteiro**  
idade\_hoje : **inteiro**  
**inicio**

ano\_atual <- 2009  
**leia** (ano)  
idade\_hoje <- ano\_atual - ano  
**escreva** (idade\_hoje)

**fimalgoritmo**

Os dados de entrada nesse algoritmo são o ano atual e o ano de nascimento digitado pelo usuário ("ano‟). O dado de saída é a idade atual do usuário ("idade\_hoje‟). Todos os dados nesse algoritmo são numéricos do tipo inteiro. Vamos analisar o código em detalhes.

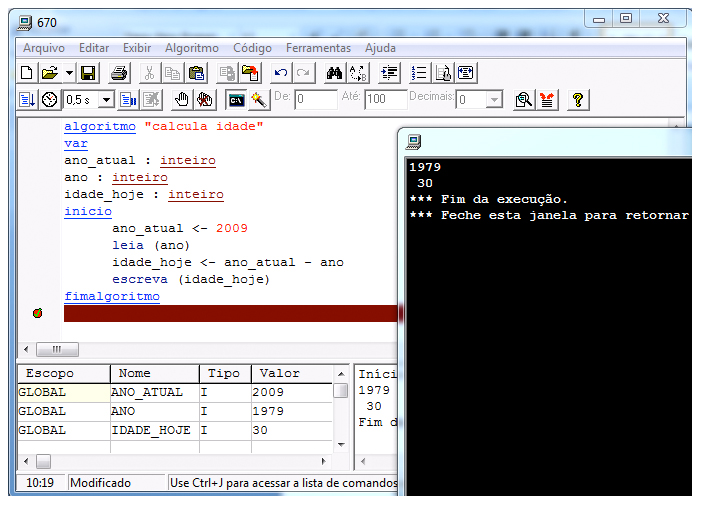
Na primeira linha do algoritmo, temos:

***algoritmo*** "calcula idade"

Nessa primeira linha, estamos nomeando o algoritmo. Veja que o nome do algoritmo está entres aspas duplas.

Na segunda linha do algoritmo, o termo var indica o campo onde ficarão as declarações das variáveis. Nas linhas 3, 4 e 5, as variáveis são declaradas. Veja:

Linha 2 - ***var***  
Linha 3 - ano\_atual : ***inteiro***  
Linha 4 - ano : ***inteiro***  
Linha 5 - idade\_hoje : ***inteiro***



**Figura 3 -** Exemplo de algoritmo no VisuAlg

No VisuAlg (Figura 3), a declaração das variáveis é feita fora do bloco de execução dos algoritmos. O bloco de execução tem seu início em seguida, na linha 6, indicado pela palavra reservada inicio, e é finalizado com a palavra reservada fimalgoritmo.

Na primeira linha do bloco de execução, temos a seguinte instrução:

ano\_atual <- 2009

O termo ano\_atual é o nome que foi dado à variável. O símbolo "<-‟ (seta para a esquerda) é um símbolo que indica atribuição. Estamos dizendo, portanto, nessa linha de código, que está sendo atribuído o valor 2009 à variável do tipo inteiro, que foi nomeada ano\_atual.

Na linha seguinte a do código, temos:

***leia*** (ano)

A instrução leia é uma solicitação ao usuário para que insira, no programa, alguma informação. No momento da execução dessa linha de código, no VisuAlg, uma caixa de diálogo aparece na tela, com um campo onde o usuário digita a informação pedida (caso o software esteja configurado para executar em modo DOS, aparecerá uma tela escura semelhante ao DOS e você digitará a informação nessa tela). Essa caixa de diálogo é mostrada na Figura 3. Nela o usuário digitou o valor 1979.

Quando o usuário insere a informação, o programa a recebe e guarda na variável que foi previamente reservada (ano). Essa variável foi criada para armazenar o dado inserido pelo usuário (o seu ano de nascimento).

Na linha seguinte, temos outra atribuição:

idade\_hoje <- ano\_atual – ano

Esse comando indica que estamos armazenando o valor encontrado na operação "ano\_atual – ano‟ (operação de subtração) no espaço da memória representado pela variável idade\_hoje.

Nesse momento, o objetivo do algoritmo foi atingido. A última instrução exibe na tela o resultado encontrado:

***escreva*** (idade\_hoje)

O comando escreva é utilizado para exibir, na tela, alguma informação. É através dele que o computador comunica-se com você, informando o resultado encontrado no algoritmo. Na Figura 3, é possível ver que o algoritmo escreveu o valor 30 como resultado do comando **escreva**. Note na Figura 3 que o VisuAlg mostra no canto inferior esquerdo as variáveis declaradas e seus respectivos valores.

### Números reais

Os números reais são os números positivos ou negativos que englobam números decimais ou fracionários. Ex.: 15, -487, 1.78, 0.254, 27835, 100, 8.50.

São também chamados de pontos flutuantes, nas linguagens de programação. Valores reais são aplicáveis em algoritmos que manipulam dados que expressam valores fracionários, como salário, média, preço, porcentagem, entre outros.

O algoritmo da média, que vimos nas duas primeiras aulas, utiliza dados numéricos reais. Observe:

ALGORITMO QUE CALCULA A MÉDIA DO ALUNO

**algoritmo** "calcula média"  
**var**  
nota1 : **real**  
nota2 : **real**   
media : **real**  
**inicio**

**leia** (nota1)   
**leia** (nota2)  
media <- (nota1 + nota2)/2   
**escreva** (media)

**fimalgoritmo**

Note que os dados de entrada (as notas do aluno) e o dado de saída (a média) são do tipo real, pois a nota de um aluno pode assumir valores fracionários, como 8.5 ou 5.2, por exemplo.

Obs.: As casas decimais, nos números reais, devem ser separadas por ponto (.) e não por vírgula (,). Por exemplo: use "8.5" e não "8,5".

### Dados literais

Os dados literais são formados por um único caractere ou uma sequência de caracteres, que podem ser letras (maiúsculas ou minúsculas), números ou símbolos especiais (como #, $, @, ?, &, entre outros). Os números, quando representados como caracteres, não podem ser utilizados para cálculos. Uma sequência de caracteres pode ser chamada também de cadeia de caracteres ou string.

Exemplos de dados literais: "Fone: 3222-2222", "Av. Senador Salgado Filho, Nº 1550", "João Silveira", "M", "152", "CEP: 59052-250", "F".

Nos nossos algoritmos, representaremos todos os dados literais sempre entre aspas duplas, sejam eles caracteres isolados ou strings, pois essa é a convenção utilizada no VisuAlg.

Observe o exemplo a seguir, utilizando uma string (cadeia de caracteres).

**algoritmo** "literais"   
**var**  
nome : **literal**  
sobrenome : **literal**  
**inicio**

nome <- "Maria"  
**escreva** ("Digite o sobrenome:")  
**leia** (sobrenome)  
**escreva** (nome," ",sobrenome)

**fimalgoritmo**

Note que utilizaremos a notação literal quando nos referirmos a strings. Veja outro exemplo com um caractere isolado:

**algoritmo** "caractere"  
**var**  
sexo : **caractere**   
**inicio**

sexo <- "F"  
**escreva** (sexo)

**fimalgoritmo**

Para um caractere isolado, utilizaremos a notação caracter ou caractere e não literal, como no exemplo anterior.

Execute, também, esses dois códigos no VisuAlg (salve em arquivos separados).

#### Dados lógicos

Os dados lógicos são também chamados de booleanos, por ter sua origem na álgebra booleana. Os valores que esses tipos de dados podem assumir são Verdadeiro ou Falso, podendo representar apenas um desses dois valores.

Um exemplo simples:

**algoritmo** "logicos"  
**var**  
igualdade : **logico**  
**inicio**

igualdade <- 10 = 12  
**escreva** (igualdade)

**fimalgoritmo**

Veja que utilizamos a notação logico para representar o dado lógico. Sabemos que 10 não é igual a 12. Logo, a saída para esse algoritmo será "FALSO".

Cole o código no editor de textos do VisuAlg e execute. Veja a saída, no simulador de saída. Agora, no lugar do número 12, coloque o número 10 (de forma que fique "10=10") e execute novamente. Observe a nova saída no simulador de saída.

### Resumo

Nesta aula, você viu o processo de armazenamento de dados na memória do computador. Você aprendeu o conceito de variáveis e as regras para a sua declaração e utilização. Viu também a importância da definição de tipos de dados, identificando os tipos existentes para a construção de algoritmos.

### Referências

ASCENCIO, Ana F. Gomes; CAMPOS, Edilene A. V. de. **Fundamentos de programação de computadores**: algoritmos, Pascal e C/C++. São Paulo: Prentice Hall, 2002.

CRUZ, Adriano J. O.; KNOPMAN, Jonas. Variáveis. **Projeto e desenvolvimento de algoritmos**: Núcleo de Computação Eletrônica. Rio de Janeiro, 2002a. Disponível em <<http://equipe.nce.ufrj.br/adriano/algoritmos/apostila/indice.htm>>. Acesso em: 25 dez. 2009.

\_\_\_\_\_\_. Tipos de dados. **Projeto e Desenvolvimento de Algoritmos**: Núcleo de Computação Eletrônica. Rio de Janeiro, 2002b. Disponível em <<http://equipe.nce.ufrj.br/adriano/algoritmos/apostila/indice.htm>>. Acesso em: 25 dez. 2009.

MOREIRA, Robson Antonio. **Lógica de programação**: aula 2. Faculdades integradas de Jaú: Curso Sistemas de Informação. Disponível em: <<http://robson.fjaunet.com.br/fundacao/logica/downloads/pdf/aula_02_logica.pdf>>. Acesso em: 25 dez. 2009.

SANTOS, Adriano B. **Aulas de algoritmo e programação de computadores**: plano de aula. Natal: IFRN, 2008. Disponível em: <[http://www.cefetrn.br/~adrianobs/algest/Aulas%20de%20Algoritmos%20\_1a%20Etapa\_.pdf](http://www.cefetrn.br/%7Eadrianobs/algest/Aulas%20de%20Algoritmos%20_1a%20Etapa_.pdf)>. Acesso em: 25 dez. 2009.

SCHROEDER, Rebeca. **Tipos de dados, expressões e variáveis**. Joinville: UDESC, 2008. Disponível em: <<http://www.joinville.udesc.br/portal/professores/rebeca/materiais/alp_4.pdf>>. Acesso em: 25 dez. 2009.

SOUZA, Alessandro J. **Algoritmos**. Natal: IFRN, 2006. Disponível em: <[http://www.cefetrn.br/~adrianobs/algest/aula2.pdf](http://www.cefetrn.br/%7Eadrianobs/algest/aula2.pdf)>. Acesso em: 25 dez. 2009.

SOUZA, Cláudio M. de. **VISUALG**: editor e interpretador de algoritmos. Disponível em: <<http://apoioinformatica.inf.br/visualg/objetivos.htm>>. Acesso em: 28 dez. 2009.