INSTITUTO FEDERAL
BAIANO

Algoritmos e Introdução à Programação

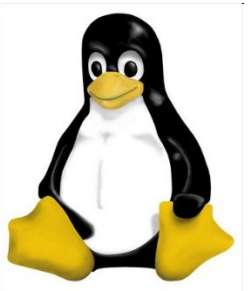
Lógica e Linguagem de Programação

Prof. José Honorato Ferreira Nunes

honoratonunes@softwarelivre.org

http://softwarelivre.org/zenorato/honoratonunes

INSTITUTO FEDERAL
BAIANO



AULA 01:

Introdução, histórico e conceitos de algoritmos

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org
http://softwarelivre.org/zenorato/honoratonunes

Resumo da aula

- Introdução à lógica de programação
- Histórico da programação
- Algoritmos
- Representação de algoritmos
- Atividades

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Introdução à lógica de programação

A lógica é a ciência do pensamento correto. Esta declaração não implica contudo em afirmar que ela seja a ciência da verdade. Mesmo que tudo o que se permita afirmar dentro da lógica seja supostamente verdadeiro em determinado contexto, as mesmas afirmações podem resultar falsas se aplicadas ao mundo real.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Introdução à lógica de programação

Os filósofos da lógica afirmam que, "para entender o que realmente acontece no mundo, precisamos entender o que não acontece", isto é, as propriedades invariantes das entidades ou objetos que o compõem.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Histórico

O uso da lógica na representação dos processos de raciocínio remonta aos estudos de Boole (1815-1864) e de De Morgan (1806-1871), sobre o que veio a ser mais tarde chamado "Álgebra de Boole".

Deve-se ao matemático alemão Göttlob Frege (1879) a primeira versão do que hoje denominamos cálculo de predicados, proposto por ele como uma ferramenta para formalizar princípios lógicos.

No final do século passado a matemática havia atingido um estágio de desenvolvimento mais do que propício à exploração do novo instrumento proposto por Frege.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Histórico

Um passo muito importante foi dado em 1930, em estudos simultâneos, porém independentes, realizados pelo alemão Kurt Gödel e o francês Jacques Herbrand.

Em 1934, Alfred Tarski produziu a primeira teoria semântica rigorosamente formal do cálculo de predicados, introduzindo conceitos precisos para "satisfatibilidade", "verdade" (em uma dada interpretação), "conseqüência lógica" e outras noções relacionadas.

No início da Seg. Guerra Mundial, em 1939, toda a fundamentação teórica básica da lógica computacional estava pronta.

Histórico

Foi somente a partir da metade dos anos 50 que o desenvolvimento da então novíssima tecnologia dos computadores conseguiu oferecer aos pesquisadores o potencial computacional necessário para a realização de experiências mais significativas com o cálculo de predicados.

Em 1958, uma forma simplificada do cálculo de predicados denominada forma clausal começou a despertar o interesse dos estudiosos. Também por essa época, Dag Prawitz (1960) propôs um novo tipo de operação sobre os objetos do cálculo de predicados, que mais tarde veio a ser conhecida por unificação.

Histórico

A programação em lógica em sistemas computacionais, entretanto, somente se tornou realmente possível a partir da pesquisa sobre prova automática de teoremas, particularmente no desenvolvimento do Princípio da Resolução por J. A. Robinson (1965).

A programação em lógica em sistemas computacionais, entretanto, somente se tornou realmente possível a partir da pesquisa sobre prova automática de teoremas, particularmente no desenvolvimento do Princípio da Resolução por J. A. Robinson (1965).

Histórico

A expressão "programação em lógica" (logic programming, originalmente em inglês) é devido a Robert Kowalski (1974) e designa o uso da lógica como linguagem de programação de computadores.

O primeiro interpretador experimental foi desenvolvido por um grupo de pesquisadores liderados por Alain Colmerauer na Universidade de Aix-Marseille (1972) com o nome de Prolog.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Algoritmos

Um algoritmo pode ser definido como um conjunto de regras (instruções), bem definidas, para solução de um determinado problema. Segundo o dicionário Michaelis, o conceito de algoritmo é a "utilização de regras para definir ou executar uma tarefa específica ou para resolver um problema específico."

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Algoritmos

- A palavra algoritmo não é um termo computacional, ou seja, não se refere apenas à área de informática. É uma definição ampla que agora que você já sabe o que significa, talvez a utilize no seu cotidiano normalmente.
- Na informática, o algoritmo é o "projeto do programa", ou seja, antes de se fazer um programa (software) na Linguagem de Programação desejada (Pascal, C, Delphi, etc.) deve-se fazer o algoritmo do programa.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Algoritmos

Um programa, é um algoritmo escrito numa forma compreensível pelo computador (através de uma Linguagem de Programação), onde todas as ações a serem executadas devem ser especificadas nos mínimos detalhes e de acordo com as regras de sintaxe da linguagem escolhida.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Algoritmos

Um algoritmo não é a solução de um problema, pois, se assim fosse, cada problema teria um único algoritmo. Um algoritmo é um 'caminho' para a solução de um problema e, em geral, existem muitos caminhos que levam a uma solução satisfatória, ou seja, para resolver o mesmo problema pode-se obter vários algoritmos diferentes.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Algoritmos

Para resolver um problema no computador é necessário que seja primeiramente encontrada uma maneira de descrever este problema de uma forma clara e precisa. É preciso que encontremos uma sequência de passos que permitam que o problema possa ser resolvido de maneira automática e repetitiva. Esta sequência de passos é chamada de algoritmo.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Algoritmos

- A noção de algoritmo é central para toda a computação. A criação de algoritmos para resolver os problemas é uma das maiores dificuldades dos iniciantes em programação em computadores
- Uma das formas mais eficazes de aprender algoritmos é através de muitos exercícios.

Algoritmos **não se aprende**	Algoritmos **se aprende**
Copiando algoritmos	Construindo algoritmos
Estudando algoritmos prontos	Testando algoritmos

Tabela 1: Dicas de como aprender e como não aprender algoritmos

Algoritmos

O aprendizado da Lógica é essencial para a formação de um bom programador, servindo como base para o aprendizado de todas as Linguagens de Programação, estruturadas ou não. De um modo geral esses conhecimentos serão de supra importância, pois ajudarão no cotidiano, desenvolvendo um raciocínio rápido.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Formas de Representação de Algoritmos

Os algoritmos podem ser representados de várias formas, como por exemplo:

- Através de uma língua (português, inglês, etc.): forma utilizada nos manuais de instruções, nas receitas culinárias, bulas de medicamentos, etc.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Formas de Representação de Algoritmos

- Através de uma linguagem de programação (Pascal, C, Delphi, etc.): esta forma é utilizada por alguns programadores experientes, que "pulam" a etapa do projeto do programa (algoritmo) e passam direto para a programação em si.

- Através de representações gráficas: são bastante recomendáveis, já que um "desenho" (diagrama, fluxograma, etc.) muitas vezes substitui, com vantagem, várias palavras.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Formas de Representação de Algoritmos

Cada uma dessas formas de representar um algoritmo, tem suas vantagens e desvantagens, cabe a pessoa escolher a forma que melhor lhe convir. As principais formas de representação de algoritmos são:

- Descrição Narrativa
- Diagrama de Nassi-Shneiderman (Diagrama de Chapin)
- Fluxograma (Diagrama de Fluxo)
- Português Estruturado (Pseudocódigo, Portugol ou Pseudolinguagem)

Descrição Narrativa

É a descrição dos passos a serem executados pelo algoritmo, feita diretamente em linguagem natural. Os passos são listados um após o outro, na sequência em que devem ser executados, cada uma em uma nova linha de texto.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Descrição Narrativa

- Exemplo de algoritmo para trocar lâmpada

1. Pegar a escada.
2. Posicionar a escada sob a lâmpada.
3. Pegar a lâmpada nova.
4. Subir na escada.
5. Remover a lâmpada antiga.
6. Colocar a lâmpada nova.
7. Descer da escada.
8. Colocar a lâmpada antiga no lixo.
9. Guardar a escada.

Atividade

- Crie um algoritmo, utilizando a descrição narrativa, que descreva os passos necessários para trocar o pneu de um carro.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Diagrama de Nassi-Shneiderman (Diagrama de Chapin)

Os Diagramas Nassi-Shneiderman, também conhecidos como Diagramas de Chapin, surgiram nos anos 70 como uma maneira de ajudar nos esforços da abordagem de programação estruturada.

A ideia básica deste diagrama é representar as ações de um algoritmo dentro de um único retângulo, subdividindo-o em retângulos menores, que representam os diferentes blocos de sequência de ações do algoritmo.

Diagrama de Nassi-Shneiderman (Diagrama de Chapin)

- Algoritmo para trocar lâmpada utilizando o Diagrama de Chapin

Inicio
Pegar a escada.
Posicionar a escada sob a lâmpada.
Pegar a lâmpada nova.
Subir na escada.
Remover a lâmpada antiga.
Colocar a lâmpada nova.
Descer da escada.
Colocar a lâmpada antiga no lixo.
Guardar a escada.
Fim

Atividade

- Crie um algoritmo, utilizando o diagrama de Chapin, que descreva os passos necessários para trocar o pneu de um carro.

- Crie um algoritmo, utilizando o diagrama de Chapin, que descreva os passos necessários para preparar café.

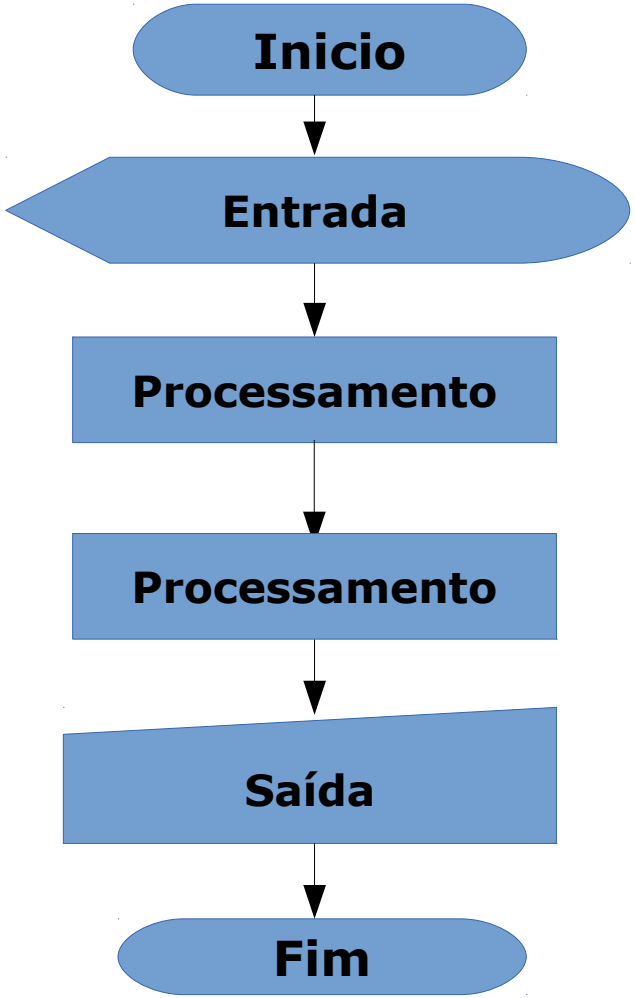
Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Fluxograma (Diagrama de Fluxo)

Os Fluxogramas ou Diagramas de Fluxo, são uma representação gráfica que utilizam formas geométricas padronizadas ligadas por setas de fluxo, para indicar as diversas ações (instruções) e decisões que devem ser seguidas para resolver o problema em questão.

Eles permitem visualizar os caminhos (fluxos) e as etapas de processamento de dados possíveis e, dentro destas, os passos para a resolução do problema.

Fluxograma (Diagrama de Fluxo)



Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividade

- Crie um algoritmo, utilizando o fluxograma, que descreva os passos necessários para trocar uma lâmpada.
- Crie um algoritmo, utilizando o fluxograma, que descreva os passos necessários para trocar o pneu de um carro.
- Crie um algoritmo, utilizando o diagrama de Chapin, que descreva os passos necessários para preparar café, utilizando uma cafeteira.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

INSTITUTO FEDERAL
BAIANO

AULA 02:

Forma de Representação, Variáveis, Tipo de Dados e Constantes

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Resumo da aula

- Formas de Representação de Algoritmos
- Atividades de revisão
- Variáveis
- Tipos de dados
- Constantes

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Formas de Representação de Algoritmos

As principais formas de representação de algoritmos são:

- Descrição Narrativa
- Diagrama de Nassi-Shneiderman (Diagrama de Chapin)
- Fluxograma (Diagrama de Fluxo)
- Português Estruturado (Pseudocódigo, Portugol ou Pseudolinguagem)

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Português Estruturado (Pseudocódigo, Portugol ou Pseudolinguagem)

O Português Estruturado, é uma forma especial de linguagem bem mais restrita que a Língua Portuguesa e com significados bem definidos para todos os termos utilizados nas instruções (comandos).

Essa linguagem também é conhecida como Portugol (junção de Português com Algol, Pseudocódigo ou Pseudolinguagem.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Português Estruturado (Pseudocódigo, Portugol ou Pseudolinguagem)

O Português Estruturado na verdade é uma simplificação extrema da língua portuguesa, limitada a pouquíssimas palavras e estruturas que têm significado pré-definido, pois deve-se seguir um padrão. Emprega uma linguagem intermediária entre a linguagem natural e uma linguagem de programação, para descrever os algoritmos.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Português Estruturado (Pseudocódigo, Portugol ou Pseudolinguagem)

A sintaxe do Português Estruturado não precisa ser seguida tão rigorosamente quanto a sintaxe de uma linguagem de programação, já que o algoritmo não será executado como um programa.

Embora o Português Estruturado seja uma linguagem bastante simplificada, ela possui todos os elementos básicos e uma estrutura semelhante à de uma linguagem de programação de computadores.

Português Estruturado (Pseudocódigo, Portugol ou Pseudolinguagem)

Resolver problemas com português estruturado pode ser uma tarefa tão complexa quanto a de escrever um programa em uma linguagem de programação qualquer só não tão rígida quanto a sua sintaxe, ou seja, o algoritmo não deixa de funcionar porque esquecemos de colocar um ';' (ponto-e-vírgula) por exemplo, já um programa não funcionaria.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Pseudocódigo - algoritmo para trocar lâmpada

Algoritmo TrocaLampada

Inicio

- Pegar a escada
- Posicionar a escada sob a lâmpada
- Pegar a lâmpada nova
- Subir na escada
- Remover a lâmpada antiga
- Colocar a lâmpada nova
- Descer da escada
- Colocar a lâmpada antiga no lixo
- Guardar a escada

FimAlgoritmo

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Padrões para Representação de Algoritmos por meio de Pseudocódigo

- Na primeira linha, após a palavra Algoritmo, é o nome do algoritmo que está sendo descrito.
- O começo dos passos de execução do algoritmo é demarcado pela palavra Inicio.
- O término dos passos de execução do algoritmo é demarcado pela palavra FimAlgoritmo.
- Todos os comandos ente Inicio e FimAlgoritmo estão levemente deslocados para direita.

Regras para Nomenclatura de Algoritmos

- Nomes de algoritmos devem conter apenas letras, números e o caractere *underscore* (_).

- Embora seja possível declarar todo o nome em letra maiúscula, é recomendado, por convenção, utilizar letras maiúsculas apenas no início de cada palavra que compõe o nome do algoritmo. Estas palavras podem ser escritas todas juntas, ou separadas pelo caractere *underscore*.

Atividades

- Crie um algoritmo, utilizando pseudocódigo, que descreva os passos necessários para trocar uma lâmpada.
- Crie um algoritmo, utilizando pseudocódigo, que descreva os passos necessários para trocar o pneu de um carro.
- Crie um algoritmo, utilizando pseudocódigo, que descreva os passos necessários para preparar café, utilizando uma cafeteira.

Atividade de revisão

- Crie um algoritmo que descreva os passos necessários para escovar os dentes, utilizando as 4 principais formas de representação de algoritmos, trabalhadas nessa disciplina.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Variáveis

Variáveis

- O bom entendimento do conceito de variável é fundamental para elaboração de algoritmos e, consequentemente de programas.
- Uma variável, é um espaço da memória do computador que "reservamos" para guardar informações (dados).

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Variáveis

- Como o próprio nome sugere, as variáveis, podem conter valores diferentes a cada instante de tempo, ou seja, seu conteúdo pode variar de acordo com as instruções do algoritmo.

- As variáveis são referenciadas através de um nome (identificador) criado por você durante o desenvolvimento do algoritmo.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Variáveis

- O conteúdo de uma variável pode ser alterado, consultado ou apagado quantas vezes forem necessárias durante o algoritmo.

- Ao alterar o conteúdo da variável, a informação anterior é perdida, ou seja, sempre "vale" a última informação armazenada na variável.

- Uma variável armazena 'apenas' um conteúdo de cada vez.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Variáveis

EXEMPLOS DE NOMES VÁLIDOS	EXEMPLOS DE NOMES INVÁLIDOS
nome_candidato	nome candidato
endereco	endereço
RG	R.G.
mes_ferias	mês_férias
dataNasc	data-Nasc
fone1	1fone

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Tipos de Dados

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Tipos de dados

Quando declaramos uma variável, precisamos identificar o tipo de informação que desejamos armazenar nela. Existem diversos tipos de dados e muitos deles são comuns na grande maioria das linguagens de programação. No nosso estudo de lógica de programação, porém, utilizaremos apenas alguns dos principais.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Tipos de dados

Os tipos de dados básicos com os quais iremos trabalhar são:

Inteiro: permite armazenar números inteiros, positivos ou negativos.
Ex.: {5; 0; 2; -2; -1500}

Real: permite armazenar números inteiros ou fracionários, positivos ou negativos.
Ex.: {5; 0; 2,5; -2,654; -1500}

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Tipos de dados

Caractere: permite armazenar caracteres alfanuméricos (ou seja: letras, números, espaços, sinais de pontuação e outros símbolos). Esse tipo também é chamado de **Literal** ou **String**. Valores do tipo caractere são sempre representados entre (“”).

“Av. Brasil, 1500”

“5”

“%”

“O tipo caractere aceita tudo! @#$%¨&*”

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Tipos de dados

Lógico: permite armazenar valores lógicos, do tipo Verdadeiro ou Falso, os quais representaremos respectivamente por **V** e **F**.

Ex.:
V
F

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Declaração de Variáveis

Para que uma variável passe a existir e possa ser utilizada no contexto de um algoritmo, é necessário que ela seja declarada. A declaração nada mais é do que a definição de uma variável, aonde informamos seu nome e tipo de informação que ela deverá ser capaz de armazenar.

Para declarar variáveis em pseudocódigo, adotamos o seguinte padrão:

nome_da_variável : tipo_de_dado

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Declaração de Variáveis

A declaração deverá ser feita no começo do algoritmo, em um bloco nomeado **Var**, antes do demarcador **Inicio**.

```
Algoritmo "ExemploVariaveis"
Var
      nome : Caractere
      endereco : Caractere
      altura : Real
      peso :  Real
      telefone : Caractere
 Inicio
 …
 FimAlgoritmo
```

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Declaração de Variáveis

É possível agrupar variáveis do mesmo tipo em uma mesma linha, declarando-as todas juntas.

```
Algoritmo "ExemploVariaveisAgrupadas"
Var
    nome, endereco, telefone : Caractere
    altura, peso : Real
Inicio
...
FimAlgoritmo
```

Prof. José Honorato F. Nunes
eu@honorato.eu

Atribuição e Inicialização de Variáveis

Atribuição é o ato de definir o valor de uma variável. Tecnicamente, isso significa escrever uma nova informação no espaço de memória identificado pelo nome que demos à variável no momento de sua declaração.

Para atribuir um novo valor a uma variável adotamos o seguinte padrão:

nome_da_variável := valor

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atribuição e Inicialização de Variáveis

```
Algoritmo “ExemploAtribuicao”
Var
    ano : Inteiro
    nomeAluno : Caractere
    altura : Real
Inicio
    ano := 2010
    nomeAluno := “Pedro da Silva”
    altura := 1,72
FimAlgoritmo
```

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atribuição e Inicialização de Variáveis

A operação de atribuição apaga qualquer informação existente na variável, sobrepondo-a com o novo valor.

```
Algoritmo “ExemploAtribuicao2”
Var
    x : Inteiro
Inicio
    x := 1
    x := 5
FimAlgoritmo
```

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividades

- Crie um algoritmo para armazenar seu nome, sobrenome, idade e sexo.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Constantes

São chamadas de constantes, as informações (dados) que não variam com o tempo, ou seja, permanecem sempre com o mesmo conteúdo, é um valor fixo (invariável). Como exemplos de constantes pode-se citar: números, letras, palavras etc.

Usamos letras maiúsculas para declarar constante como forma de diferencia-las das demais variáveis.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Constantes

```
Algoritmo “ExemploConstantes”
Constantes
    PI := 3,14
    MAX_GRAUS := 90
    INSTITUICAO := “IF Baiano”
    RELATORIO_ATIVADO:= F
Var
    ...
Inicio
    ...
FimAlgoritmo
```

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

INSTITUTO FEDERAL
BAIANO

AULA 03:

Operadores e Expressões

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

RESUMO DA AULA

- Atividade de Revisão
- Operadores e Expressões:
  - Operadores Aritméticos
  - Expressões
  - Operadores Relacionais
  - Operadores Lógicos
  - Operador Literal

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Operadores Aritméticos

Muitas vezes, ao desenvolvermos algoritmos, é comum utilizarmos expressões matemáticas para a resolução de cálculos. Neste capítulo são apresentados os operadores aritméticos necessários para determinadas expressões.

Operação	Símbolo	Prioridade
Multiplicação (Produto)	*	1a.
Divisão	/	1a.
Adição (Soma)	+	2a.
Subtração (Diferença)	-	2a.

Operadores Aritméticos

Nas linguagens de programação e, portanto, nos exercícios de algoritmos que iremos desenvolver, as expressões matemáticas sempre obedecem às regras matemáticas comuns, ou seja:

Quando duas ou mais expressões tiverem a mesma prioridade, a solução é sempre iniciada da expressão mais à esquerda até a mais à direita.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Operadores Aritméticos

As expressões dentro de parênteses são sempre resolvidas antes das expressões fora dos parênteses. Quando existem vários níveis de parênteses, ou seja, um parêntese dentro de outro, a solução sempre inicia do parêntese mais interno até o mais externo (de dentro para fora).

- Soma := ((2+2*4) – 20 / (1+1))

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Expressões

Para o desenvolvimento de algoritmos que possuam cálculos matemáticos, as expressões aritméticas devem estar horizontalizadas, ou seja, linearizadas e também não esquecendo de utilizar os operadores corretamente.

Matemática Tradicional	Algoritmo
$M = \frac{N1 + N2}{2}$	M := (N1 + N2) / 2

Expressões

- Soma := ((2+2*4) – 20 / (1+1))
- Soma := ((2+8) – 20 / 2)
- Soma := (10 – 20 / 2)
- Soma := (10 – 10)
- Soma := 0

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Expressões

Desta forma, veja os seguintes exemplos e os respectivos resultados:

- ExemploA : 2 + (6 * (3 + 2)) := 32
- ExemploB : 2 + 6 * (3 + 2) := ?

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividades

- Crie um algoritmo para calcular a média de consumo de combustível de um veículo qualquer. O usuário deverá informar: quilometragem inicial, quilometragem final e a quantidade de litros consumida durante a viagem. Represente seu algoritmo utilizando pseudocódigo e fluxograma.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Resposta da Atividade: Portugol

```
Algoritmo  "MediaConsumo"
Var
    kmInicial, kmFinal, qtdConsumida, mediaConsumo : Real
Inicio
    kmInicial := 0
    kmFinal := 0
    qtdConsumida := 0
    mediaConsumo := 0
    Escreva("Informe a quilometragem inicial :")
    Leia (kmInicial)
    Escreva("Informe a quilometragem final :")
    Leia (kmFinal)
    Escreva("Informe o consumo de combustível :")
    Leia (qtdConsumida)
    mediaConsumo :=  (kmFinal - kmInicial) / qtdConsumida
    Escreva ("A média de consumo é:")
    Escreva (mediaConsumo)
FimAlgoritmo
```

Leia \ Escreva

O exemplo anterior apresenta duas novidades, os comandos de leitura e escrita que simulam a interação com o usuário.

Para simular entradas do teclado, utilizamos o comando **Leia**.

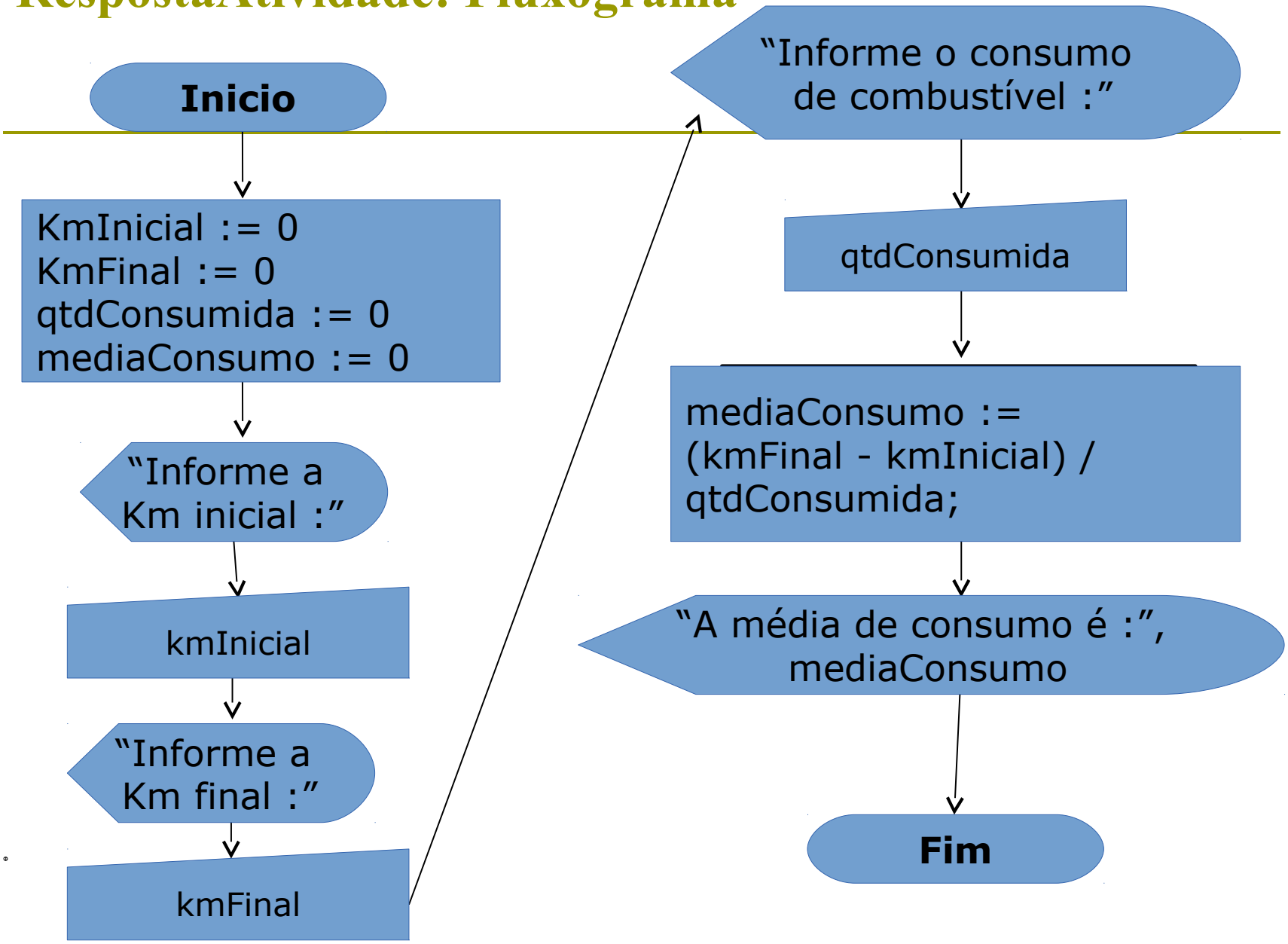
Para simular saídas na tela, utilizamos o comando **Escreva**.

Leia \ Escreva - Fluxograma

Leia\ENTRADA
Representa a entrada de dados via teclado

Escreva\SAÍDA
Representa a saída de dados via teclado

RespostaAtividade: Fluxograma



Atividade

Crie um algoritmo, utilizando Portugol, que calcule o valor total da compra de um cliente. Considere que a loja vende tudo a R$ 1,99 e que o cliente deve informar a quantidade de itens da compra.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividade

Crie um algoritmo, utilizando Portugol, que calcule a nota final de um aluno. Considere que a avaliação consiste de 3 notas (trabalho valendo 5 pontos, seminário valendo 5 pontos e prova valendo 10 pontos).

O programa deve receber as notas dos alunos e calcular a média final;

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividade

Crie um algoritmo, utilizando Portugol, que calcule quantidade de latas de cervejas necessárias para uma festa.

Considere que a média de consumo para mulheres é de 8 latas e para homens de 10 latas.

O usuário vai informar a quantidade de homens e a quantidade de mulheres na festa e o sistema deve informar a quantidade de latas de cervejas para a festa.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Operadores Relacionais

Operações relacionais são as comparações permitidas entres valores, variáveis, expressões e Constantes.

Símbolo	Significado
>	maior
<	menor
=	igual
> =	maior ou igual
< =	menor ou igual
< >	diferente

Operadores Relacionais

Comparação Válida	Exemplo
variável e constante	X = 3
variável e variável	A <> B
variável e expressão	Y = W + J
expressão e expressão	(X + 1) < (Y + 4)

Operadores Lógicos

Os operadores lógicos permitem que mais de uma condição seja testada em uma única expressão, ou seja, pode-se fazer mais de uma comparação (teste) ao mesmo tempo.

Operação	Operador
Negação	**não**
Conjunção	**e**
Disjunção (não-exclusiva)	**ou**
Disjunção (exclusiva)	**xou** (lê-se: *"ou exclusivo"*)

Operadores Lógicos

Note que a Tabela anterior, apresenta os operadores lógicos já ordenados de acordo com suas prioridades, ou seja, se na mesma expressão tivermos o operador ou e o operador não, por exemplo, primeiro devemos executar o *não* e depois o *ou*.

Operadores Lógicos - Tabela Verdade

A	B	A e B	A ou B	não A	A xou B
F	F	F	F	V	F
F	V	F	V	V	V
V	F	F	V	F	V
V	V	V	V	F	F

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Operadores Lógicos - Tabela Verdade

Expressão	Quando eu não saio?
Se chover *e* relampejar, eu não saio.	Somente quando chover e relampejar ao mesmo tempo (apenas 1 possibilidade).
Se chover *ou* relampejar, eu não saio.	Somente quando chover, somente quando relampejar ou quando chover e relampejar ao mesmo tempo (3 possibilidades).
Se chover *xou* relampejar, eu não saio.	Somente quando chover, ou somente quando relampejar (2 possibilidades).

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Operador Literal

O operador literal tem a função de unir duas informações do tipo texto. Esse processo é chamado de concatenação.

Operador: ,

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Operador Literal

```
Algoritmo “ExemploConcatenacao”
Var
    nome,sobrenome,nomeCompleto : Caractere
Início
    Escreva(“Digite o nome”)
    Leia(nome)
    Escreva(“Digite o sobrenome”)
    Leia(sobrenome)
    nomeCompleto := nome + “ ” + sobrenome
    Escreva(“O nome completo é : ”, nomecompleto)
FimAlgoritmo
```

Prof. José Honorato F. Nunes
eu@honorato.eu

Atividade

Crie um algoritmo que solicite o nome e a idade do usuário e, em seguida, imprima estas informações de forma concatenada em uma frase de boas vindas. Represente seu algoritmo em pseudocódigo.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Resposta da atividade anterior

```
Algoritmo  “ExemploConcatenacao”
Var
    idade : Inteiro
    nome : Caractere
Inicio
    Escreva(“Informe o nome do usuário:”)
    Leia (nome)
    Escreva(“Informe a idade do usuário:”)
    Leia (idade)
    Escreva (“Bem vindo: ”, nome, “ sua idade é ”, idade)
FimAlgoritmo
```

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividade

Crie um algoritmo em pseudocódigo para calcular a taxa de serviço do garçom, a partir da entrada do valor da conta. A taxa de serviço é fixa em 10%. O sistema deverá escrever na tela o valor da taxa de serviço e depois o valor total a ser pago.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

INSTITUTO FEDERAL
BAIANO

AULA 04:

Estruturas de Controle: Seleção

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Resumo da aula

- Estruturas de Controle
- Seleção Simples
- Seleção Composta
- Seleção Aninhada
- Atividades

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividade de Revissão

Crie um algoritmo para calcular a velocidade média atingida por um veículo durante uma viagem. Reflita sobre os dados que serão necessários solicitar ao usuário. Represente seu algoritmo utilizando pseudocódigo e fluxograma.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Resposta da Atividades: Portugol

```
Algoritmo "VelocidadeMedia"
Var
      kmInicial, kmFinal : Inteiro
      tempoInicial, tempoFinal, media : Real
Inicio
      Escreva("Informe a quilometragem inicial :")
      Leia (kmInicial)
      Escreva("Informe a quilometragem final :")
      Leia (kmFinal)
      Escreva("Informe o tempo\hora inicial :")
      Leia (tempoInicial)
      Escreva("Informe o tempo\hora final :")
      Leia (tempoFinal)
      media :=  (kmFinal - kmInicial)/(tempoFinal - tempoInicial)
      Escreva ("A velocidade média é:")
      Escreva (media)
FimAlgoritmo
```

Estruturas de Controle

Estávamos trabalhando com algoritmos puramente sequenciais, ou seja, todas as instruções eram executadas seguindo a ordem do algoritmo (normalmente, de cima para baixo). Começaremos a estudar estruturas de seleção. Uma estrutura de seleção, como o próprio nome já diz, permite que determinadas instruções sejam executadas ou não, dependendo do resultado de uma condição (teste), ou seja, o algoritmo vai ter mais de uma saída, uma opção que será executada de acordo com o teste realizado.

Estruturas de Controle

Nos exemplos e exercícios que vimos, sempre foi possível resolver os problemas com uma sequência de instruções que eram executadas apenas uma vez. Existem três estruturas básicas para a construção de algoritmos, que são:

- algoritmos sequenciais;
- algoritmos com seleção;e
- algoritmos com repetição.

Estruturas de Controle

A combinação dessas três estruturas permite-nos a construção de algoritmos para a resolução de problemas extremamente complexos. Nesta aula veremos as estruturas de repetição possíveis em algoritmos e existentes na maioria das Linguagens de Programação.

Estruturas de Controle - Seleção

CLASSIFICAÇÃO:

- Estrutura de Seleção Simples;
- Estrutura de Seleção Composta;
- Estrutura de Seleção Aninhadas;

Estruturas de Controle: Seleção Simples

A *estrutura de seleção simples* permite definir um bloco de instruções que serão executadas apenas se forem atendidos os critérios definidos. Esta estrutura também é conhecida como *desvio condicional simples*.

No pseudocódigo, a *estrutura de seleção simples* é representada pelo comando Se, que utiliza a seguinte estrutura:

Se (condição) Entao

...

FimSe

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Estruturas de Controle: Seleção Simples

```
Algoritmo "ExemploSe"
Var
    idadeAluno : Inteiro
Inicio
    Escreva("Informe a idade do aluno:")
    Leia (idadeAluno)

    Se (idadeAluno < 18) Entao
        Escreva("O aluno é menor de idade.")
    FimSe

    Escreva ("A idade do aluno é: ", idadeAluno)
FimAlgoritmo
```

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Estruturas de Controle: Seleção Composta

A *estrutura de seleção composta* permite definir dois blocos de instruções, sendo que um deles será executado e o outro não, de acordo com o atendimento ou não dos critérios definidos. Esta estrutura também é conhecida como *desvio condicional composto*.

No pseudocódigo, a *estrutura de seleção simples* é representada pelo comando Se...Senao, que utiliza a seguinte estrutura:

Se (condição) Entao

...

Senao

...

FimSe

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Estruturas de Controle: Seleção Composta

```
Algoritmo  "ExemploSeSenao"
Var
      idadeAluno : Inteiro
Inicio
      Escreva("Informe a idade do aluno:")
      Leia (idadeAluno)

      Se (idadeAluno < 18) Entao
            Escreva("O aluno é menor de idade.")
      Senao
            Escreva("O aluno é maior de idade.")
      FimSe

      Escreva ("A idade do aluno é: ", idadeAluno)
FimAlgoritmo
```

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividade

Crie um algoritmo em pseudocódigo para calcular a média final do aluno. O sistema deve mostrar o nome do aluno e receber duas notas (trabalho e prova). Depois de calcular a média final, se o aluno tiver média >= 7 informar que ele foi aprovado, senão informar que ele foi reprovado.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Resposta da atividade anterior

```
Algoritmo “ExemploSe”

Var
    nome : Caractere
    notaTrabalho, notaProva, media : Real
Inicio
    Escreva(“Digite o nome”)
    Leia(nome)
    Escreva(“Digite a nota do trabalho”)
    Leia(notaTrabalho)
    Escreva(“Digite a nota da prova”)
    Leia(notaProva)
    media := (notaTrabalho + notaProva) / 2
    Se (media >= 7) Entao
        Escreva(“Aluno ”, nome,  “ foi APROVADO com média = ”, media)
    Senao
        Escreva(“Aluno ”, nome, “ foi REPROVADO com média = ”, media)
    FimSe
FimAlgoritmo
```

Estruturas de Controle: Seleção Aninhadas

Muitas vezes, dentro de um fluxo condicional, será necessário tomar uma nova decisão. Ou pode ser que tenhamos mais de duas opções de fluxo de execução. Em ambos os casos podemos utilizar *estrutura de seleção aninhadas*, que nada mais são do que uma estrutura de seleção dentro de outra.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

```
Algoritmo  “ExemploSeAninhado”
Var
      nota1, nota2, media : Real
      nome, resultado : Caractere
Inicio
      Escreva(“Informe o nome do aluno:”)
      Leia (nome)
      Escreva(“Informe a primeira nota do aluno:”)
      Leia (nota1)
      Escreva(“Informe a segunda nota do aluno:”)
      Leia (nota2)
      media := (nota1 + nota2) / 2
      Se (media >= 7) Entao
            resultado := “aprovado”
      Senao
            Se (media >= 3) Entao
                  resultado := “na final”
            Senao
                  resultado := “reprovado”
            FimSe
      FimSe
      Escreva (“O aluno é ”, nome, “ está ”, resultado)
FimAlgoritmo
```

Atividades

- Faça um algoritmo que receba um número e mostre uma mensagem caso este número seja maior que 10.
- Escrever um algoritmo que leia dois valores inteiro distintos e informe qual é o maior.
- Faça um algoritmo que receba um número e diga se este número está no intervalo entre 100 e 200.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividades

- Faça um algoritmo que receba três valores inteiros e organize esses valores em ordem decrescente.

- Faça um algoritmo que solicite o nome, idade e sexo do aluno e caso ele seja homem e maior de idade solicitar ao aluno o número da carteira de reservista, caso seja homem menor de idade emitir a mensagem: ”aguardando idade para o exercício militar”, caso seja mulher emitir a mensagem: ”dispensada do exercício militar”.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividades

- Escrever um algoritmo que leia o nome e as três notas obtidas por um aluno durante o semestre. Calcular a sua média (aritmética), informar o nome e sua menção aprovado (media >= 7), Reprovado (media <= 5) e Recuperação (media entre 5.1 a 6.9).

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividades

- Crie um algoritmo, utilizando pseudocódigo, que leia dois números informados pelo usuário e, em seguida, exiba na tela uma mensagem informando se o maior deles é o primeiro, o segundo, ou se são iguais.

- Crie um algoritmo, utilizando pseudocódigo, que:
  - Leia três números informados pelo usuário;
  - Multiplique o menor valor lido pelo maior valor e some o resultado com o valor do meio;
  - Exiba na tela o resultado da soma.

Prof. José Honorato Ferreira Nunes
honoratonunes@softwarelivre.org

Atividade

Crie um algoritmo em pseudocódigo para aplicar um percentual de desconto sobre o valor de uma compra informado pelo usuário. Os percentuais de desconto são:

- 15% para compras acima de R$ 500,00;
- 10% para compras entre R$ 200,00 e R$ 499,99;
- 5% para compras abaixo de R$ 200,00.

Mostre na tela uma mensagem informando: valor antes do desconto, valor do desconto e valor a ser pago.

Bibliografia

- BENEDUZZI, Humberto M. e METZ, João A. **Lógica e Linguagem de Programação – Introdução ao Desenvolvimento de Software (1ª edição)**. Editora do Livro Técnico, 2010
- MANZANO,Wilson Y. Yamaturni-São Paulo-SP. **Lógica estruturada para programação de computadores**, Ed. Érica 1997 e 2001.
- MORAES, Celso Roberto. **Estruturas de Dados e Algoritmos**. Ed. Érica, São Paulo
- LOPES, Anita. **Introdução à programação**. Rio de Janeiro: Campus,2002.
- SEBESTA, Robert W. **Conceitos de linguagens de programação**. 9. ed. Porto Alegre: Bookman, 2003.
- CORMEN, Thomas H. **Algoritmos: teoria e prática**. Rio de Janeiro: Campus, 2002.
- ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi de. **Fundamentos da programação de computadores: algoritmos, Pascal e C/C++ e Java**. 2. ed. São Paulo: Pearson Prentice Hall, 2008.
- ZIVIANI, Nivio. **Projeto de algoritmos com implementações em Pascal e C.** São Paulo: Pioneira Thomson Learning.