Universidade Federal do Vale do São Francisco
Curso de Engenharia de Computação

Introdução a Algoritmos

(baseado no material do prof. Marcelo Linder)

Prof. Jorge Cavalcanti
jorge.cavalcanti@univasf.edu.br
www.univasf.edu.br/~jorge.cavalcanti
www.twitter.com/jorgecav

Introdução a Algoritmos

- **Apresentação da Disciplina**
  - Dicas de (boa) convivência acadêmica
  - Metodologia
  - Bibliografia
  - Avaliação

Introdução a Algoritmos

- **Conteúdo**
  - Conceitos de algoritmo.
  - Algoritmo como representação da solução de problemas.
  - Constantes. Identificadores. Palavras reservadas. Variáveis e tipos primitivos.
  - Operadores. Expressões. Instruções.
  - Lógica de programação.
  - Pseudo-linguagem e seu uso na representação de algoritmos.
  - Comandos de entrada e saída de dados.
  - Estrutura de controle de fluxo (sequencial, condicional e iterativa).
  - Teorema de Böhm-Jacopini.
  - Estruturas de dados homogêneas e heterogêneas.
  - Modularização. Recursão.

Introdução a Algoritmos

- **Objetivo geral**

- Tornar o aluno capaz de visualizar soluções computacionais para problemas através da aplicação dos conceitos da lógica de programação e dotá-lo da capacidade de construção de algoritmos, em pseudo-linguagens, que modelem as soluções vislumbradas.

Introdução a Algoritmos

Objetivos Específicos:

- Desenvolver a habilidade de construir modelos por meio da compreensão da atividade ou tarefa a ser modelada;
- Desenvolver a percepção de que quando se constroem modelos, eles não apenas produzem respostas, mas principalmente criam uma poderosa ferramenta conceitual que pode inclusive ser comunicada a outros e reusada em outras situações;
- Desenvolver o raciocínio lógico e abstrato;
- Familiarizar com o modelo sequencial de computação;
- Apresentar técnicas e pseudo-linguagens para construção e representação de algoritmos;
- Treinar no processo de desenvolvimento de algoritmos na ordem abaixo especificada:
  - Identificar o problema;
  - Fazer suposições e interpretações;
  - Criar um algoritmo para representar as interpretações feitas;
  - Verificar o algoritmo;
  - Fazer o processo de manutenção do algoritmo.

Introdução a Algoritmos

Bibliografia:

Básica

- ASCENCIO, A. F. G.; CAMPOS, E. A. V. Fundamentos da Programação de Computadores. 2ª ed. Editora Pearson Education, 2003.
- OLIVEIRA, A. B.; BORATTI, I. C. Introdução à Programação - Algoritmos. 3 ª ed. Visual Books, 1999.
- CORMEN, T. H. et al. Algoritmos, Teoria e Prática. 2ª ed. Elsevier, 2002.

Complementar:

- MEDINA, M.; FERTIG, C. Algoritmos e Programação- Teoria e Prática. 2ª Ed. Novatec, 2006.
- FORBELLONE, A.; EBERSPÄCHER, H. Lógica de Programação - A construção de algoritmos e estruturas de dados. 3ª ed. Pearson Education, 2005.
- CARBONI, I. F. Lógica de Programação. Cengage Learning, 2003.

Introdução a Algoritmos

- **Metodologia:**
  - **Aulas expositivo-dialogadas.**
    - Fornecer os componentes teóricos e conceituais.
  - **Recursos materiais utilizados:**
    - Quadro branco, pincel, notebook, projetor, dentre outros recursos.
  - **Avaliação:**
    - A avaliação consistirá de 3 (três) provas escritas individuais.
    - A média final do aluno consistirá na média aritmética das três notas.

Introdução a Algoritmos

- **Introdução**
- O computador como ferramenta indispensável:
  - Faz parte das nossas vidas;
  - Por si só não faz nada de útil;
  - Grande capacidade de resolução de problemas;
  - Necessita ser instruído;
  - É extremamente rápido;
  - Possui um comportamento previsível;
  - Não se cansa e pode ser usado à exaustão.

Introdução a Algoritmos

- **Introdução**
  - Computador → É uma máquina capaz de possibilitar variados tipos de tratamento automático de informações ou processamento de dados.
  - O que deve ser feito para que um determinado tratamento automático de informações ocorra?
    - Deve-se instruir o computador para que o mesmo utilizando-se de sua estrutura, execute determinada tarefa.
  - Como?
    - Software (programas) → Sequências de instruções a serem executadas por um computador.

Introdução a Algoritmos

- **Introdução**
  - Software:
    - Parte intangível: conhecimentos e idéias que fazem o hardware exibir um certo comportamento.
    - Quanto mais usado, menos propenso à falhas.
    - Confere funcionalidade ao hardware.
    - Pode ser adquirido ou desenvolvido.
  - Comprar ou desenvolver?
    - Depende do problema que se quer resolver;
    - Avaliar custo x benefício;
    - Diferentes plataformas;
    - Manutenção e customização.

Introdução a Algoritmos

- **Problema** (Dicionário Michaelis):

- Substantivo Masculino.
- Questão matemática proposta para ser resolvida.
- Questão difícil, delicada, suscetível de diversas soluções.
- Qualquer coisa de difícil explicação; mistério, enigma.
- Dúvida, questão.

Introdução a Algoritmos

- **Introdução**
  1. **Problema**
     - Precisa ser conhecido em todos os seus aspectos;
     - É necessário ter resposta para todas as perguntas que dele possam suscitar;
     - É fundamental considerar todas as situações adversas;
     - Nenhum detalhe deve ser omitido.
  2. **Solução**
     - Existe solução para o problema?
     - Qual o custo da sua implementação?
     - Qual o custo da sua execução?
     - Como iremos representá-la?

Introdução a Algoritmos

- **Exemplos de Problema**
- Problemas fazem parte do nosso cotidiano.
- Exemplo de problemas cotidianos:
  - Trocar a resistência de um chuveiro.
  - Definir onde Almoçar.
- Sempre que nos deparamos com um problema buscamos um procedimento para solucionar o mesmo.

Introdução a Algoritmos

- **Exemplo de Solução**

- Por exemplo, para trocar a resistência de um chuveiro devemos:
  - Adquirir uma resistência nova;
  - Localizar o chuveiro a ser manipulado;
  - Abrir o chuveiro;
  - Retirar a resistência defeituosa;
  - Colocar a resistência nova;
  - Fechar o chuveiro;
  - Descartar a resistência defeituosa.

- Definir onde Almoçar.
  - ...

Introdução a Algoritmos

3. Conceito de Lógica

O que orientou a obtenção dos procedimentos para as soluções vislumbradas?

A lógica.

O que é lógica?

A lógica é o ramo da Filosofia e da Matemática que estuda os métodos e princípios que permitem fazer distinção entre raciocínios válidos e não válidos, determinando o processo que leva ao conhecimento verdadeiro.

Introdução a Algoritmos

3. Conceito de Lógica

- O uso da lógica é primordial na solução de problemas. Com ela é possível alcançar objetivos com eficiência e eficácia.
- Ninguém ensina outra pessoa a pensar, mas a desenvolver e aperfeiçoar esta técnica, com persistência e constância.

Introdução a Algoritmos

4. Conceito de Algoritmo

- Representação de uma solução para um problema, com algumas características:
  - Seqüência finita de etapas;
  - Individualmente, existe realização possível para cada uma das etapas consideradas;
  - Termina após um tempo finito.
- Diversas são as técnicas e métodos existentes para a construção de algoritmos.
  - No entanto, todas elas possuem um mesmo objetivo, onde as suas variações não passam de pequenos detalhes frente a sua organização geral no atendimento a uma ou outra área mais especificamente.

Introdução a Algoritmos

- **Algoritmos no mundo real**
- Qualquer sequência de etapas para se resolver um problema ou chegar a um objetivo é um algoritmo.
- Exemplos:

ALGORITMO: TROCAR UMA LÂMPADA
PASSO 1: Pegar a lâmpada nova
PASSO 2: Pegar a escada
PASSO 3: Posicionar a escada embaixo da lâmpada queimada
PASSO 4: Subir na escada com a lâmpada nova
PASSO 5: Retirar a lâmpada queimada
PASSO 6: Colocar a lâmpada nova
PASSO 7: Descer da escada
PASSO 8: Ligar o interruptor
PASSO 9: Guardar a escada
PASSO 10: Jogar a lâmpada velha no lixo

ALGORITMO: SACAR DINHEIRO
PASSO 1: Ir até o caixa eletrônico
PASSO 2: Colocar o cartão
PASSO 3: Digitar a senha
PASSO 4: Solicitar o saldo
PASSO 5: Se o saldo for maior ou igual à quantia desejada, sacar a quantia desejada; caso contrário sacar o valor do saldo
PASSO 6: Retirar dinheiro e cartão
PASSO 7: Sair do caixa eletrônico

Introdução a Algoritmos

- **Introdução**
  5. **Programa**
  - Precisamos então:
    - Estabelecer um mecanismo de comunicação com o computador, de modo a fazê-lo entender tudo que quisermos que ele faça;
    - Os computadores entendem somente instruções, postas em uma sequência lógica e seguindo diversas regras;
    - Essas instruções e essas regras não são de fácil uso pelas pessoas, daí a necessidade de uma etapa intermediária, que é a construção de um **programa,** a partir do **algoritmo**;
    - Para isso, vamos usar uma "linguagem de programação".
    - Um pouco mais complexas do que as linguagens usadas para representar algoritmos;
    - Mas mais fáceis de serem entendidas pelo computador.

Introdução a Algoritmos

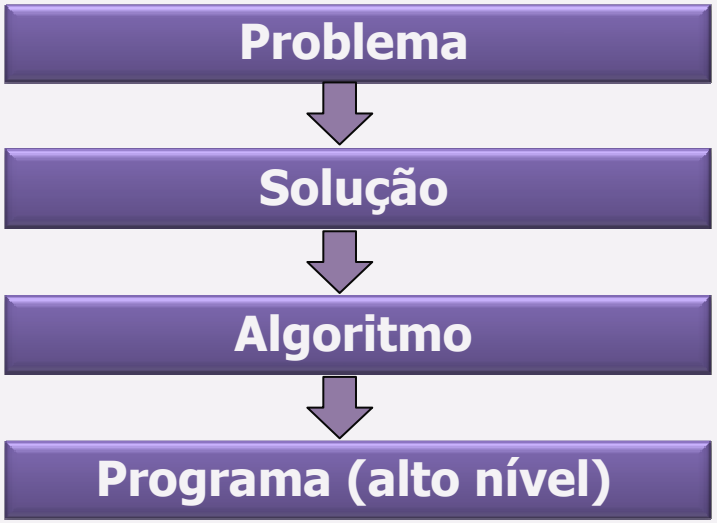
- **Introdução**
  5. **Programa**
  - Precisamos então:
    - Traduzir algoritmos para programas;
    - Conhecer e utilizar (pelo menos) duas linguagens de programação;
    - Entender que erros nas traduções do algoritmo para a linguagem são comuns;

Introdução a Algoritmos

- **Introdução**

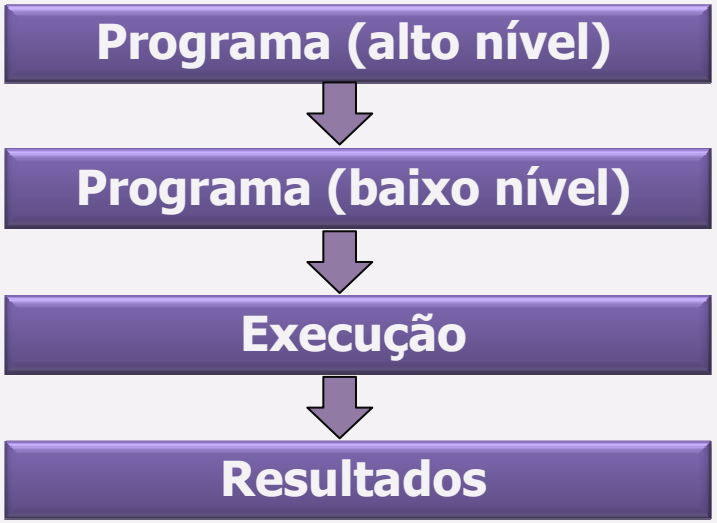
  Revendo e separando bem as coisas a partir daqui pra frente:

  - Sua parte:



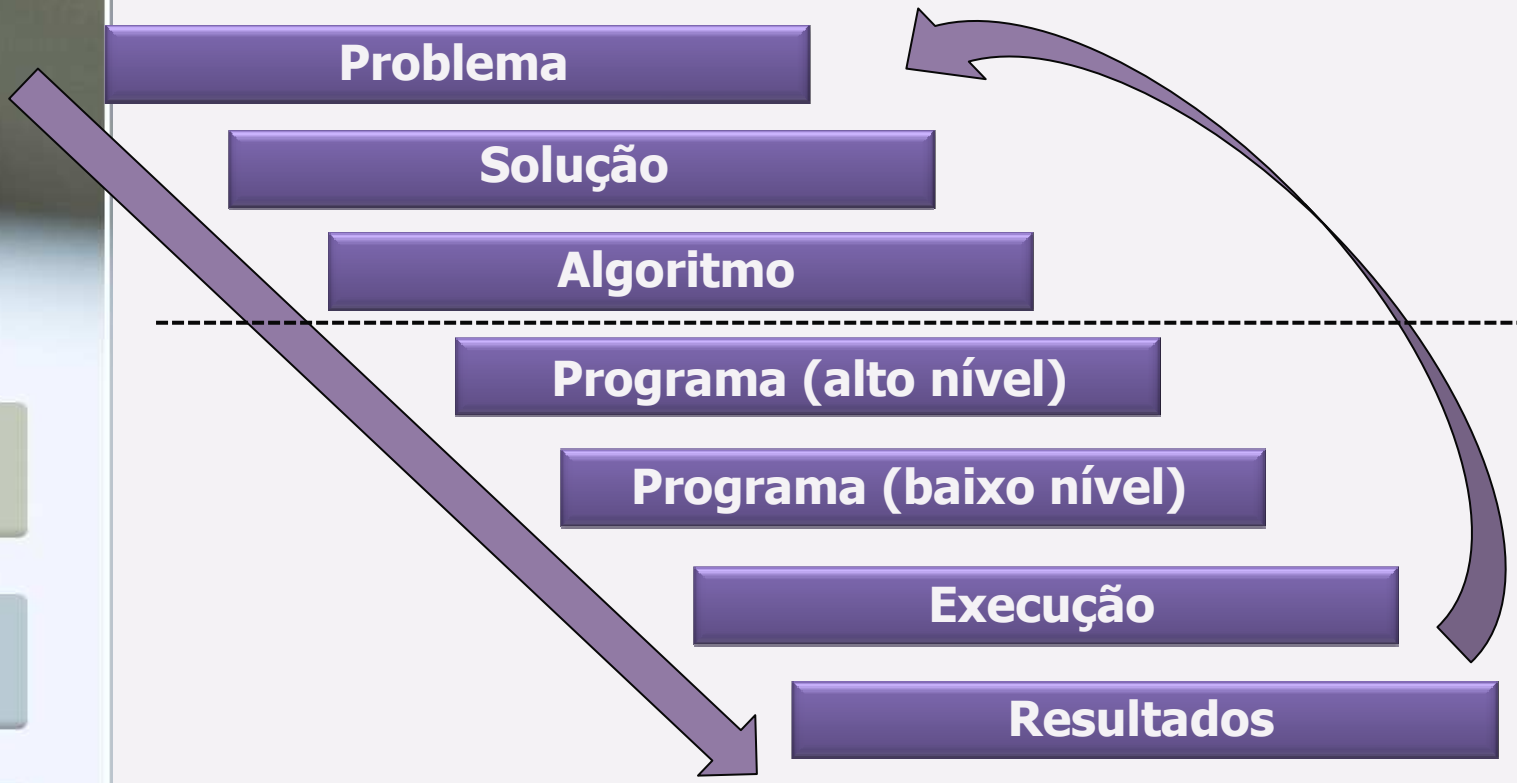
Introdução a Algoritmos

- **Introdução**
  - E se der errado?
    - Voltar a mesa e descobrir onde está o erro.



Introdução a Algoritmos

- **Introdução**
  - Ciclo de desenvolvimento



Introdução a Algoritmos

- **Descrição Narrativa**
  - Conforme vimos, uma descrição narrativa em linguagem natural foi utilizada na descrição dos algoritmos.
  - Qual a vantagem?
    - Não há a necessidade de aprender nenhum novo conceito.
  - Qual a desvantagem?
    - Em virtude da ambiguidade presente na linguagem natural a descrição narrativa é passível de mais de uma interpretação.

Introdução a Algoritmos

- **Descrição Narrativa**
  - Um exemplo de ambiguidade presente em uma sentença na linguagem natural é:
    - O policial escutou o barulho da porta.
      - Esta frase pode ter pelo menos três interpretações:

    1 - O policial escutou o barulho produzido pela porta.

    2 - O policial estava junto à porta e escutou o barulho.

    3 - O policial escutou o barulho que veio através da porta.
  - Outro exemplo: onde você colocaria uma vírgula na sentença abaixo?

    Se o homem soubesse o valor que tem a mulher andaria de quatro à sua procura.

Introdução a Algoritmos

- **Métodos de Representação de Algoritmos**
- Do ponto de vista computacional um algoritmo será implementado em uma linguagem de computação gerando um programa, o qual visa instruir um computador (uma máquina) a executar determinada tarefa.
- Devemos ter consciência que um computador não é dotado da capacidade de tomar decisões com base em premissas. Portanto, não podemos instruir um computador com sentenças dúbias.

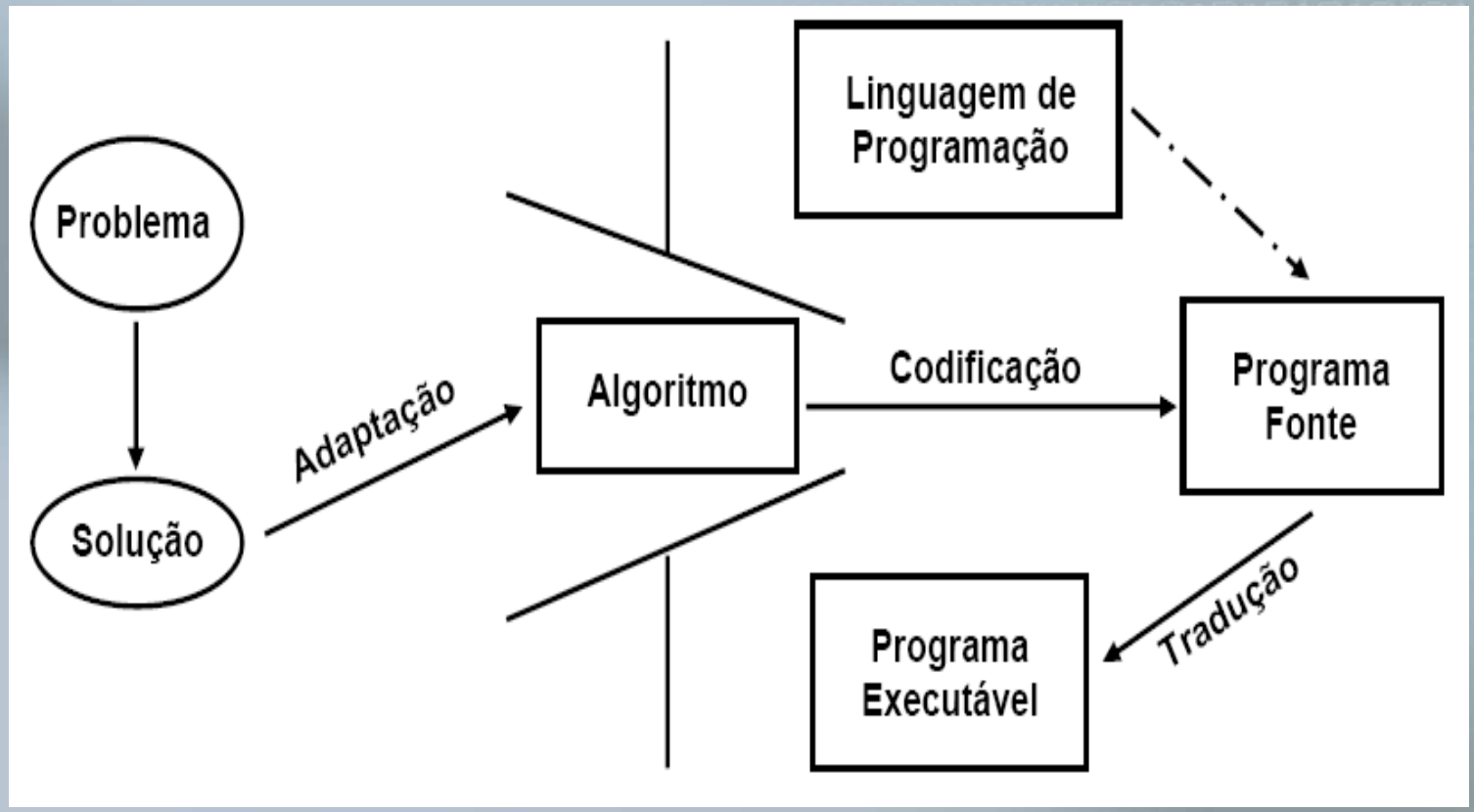
Introdução a Algoritmos

- **Métodos de Representação de Algoritmos**
- Sendo assim, consideraremos que um algoritmo é uma sequência, que não permite ambiguidade, de passos finitos, passível de ser executada com um esforço finito em tempo finito e que acaba para qualquer entrada (inclusive erro).
- O algoritmo tem como papel fundamental ser o elo entre dois mundos (real e computacional). A atividade de programação tem início com a construção do algoritmo.

Introdução a Algoritmos

- **Métodos de Representação de Algoritmos**
  - A elaboração do algoritmo descreve a necessidade dos dados e as suas manipulações durante a execução da lógica proposta por ele, sendo feito por meio de técnicas diferentes que representarão a seqüência desses passos (ou etapas).
  - Entre as várias técnicas existentes, serão abordadas:
    - Fluxograma;
    - Pseudocódigo (Português Estruturado).
  - **O fluxograma** utiliza figuras geométricas predefinidas para descrever as ações (ou instruções) a serem realizadas na resolução de um problema.
  - **Pseudocódigo** consiste na descrição estruturada, por meio de regras pré-definidas, de passos (ou instruções) a serem realizados para a resolução do problema, utilizando a linguagem natural para representar o raciocínio.

Introdução a Algoritmos



Introdução a Algoritmos

4. **Algoritmo**

- Representação:
  - Linguagem natural (vista anteriormente);
  - Pseudocódigo (linguagem textual com poucos símbolos e regras, que são simples);
  - Fluxograma (linguagem visual composta por poucos símbolos e regras)
  - Um algoritmo expressa uma solução para um problema.
- Acontece que:
  - Computadores não entendem (normalmente, ou pelo menos da forma como nós precisamos):
    - Linguagens naturais;
    - Pseudocódigos;
    - Fluxogramas.

Introdução a Algoritmos

- **Pseudocódigo (Portugol)**
  - Descrição narrativa utilizando nosso idioma para descrever o algoritmo.
  - Vantagem – sua transcrição para qualquer linguagem de programação é quase que direta.
  - Desvantagem – é necessário aprender as regras do pseudocódigo.

Exemplo de uma descrição narrativa.

- Receber e efetuar a soma de dois números, exibindo o resultado ao final.

1. Receber os dois números.
2. Efetuar a soma dos dois números.
3. Mostrar o resultado.

Introdução a Algoritmos

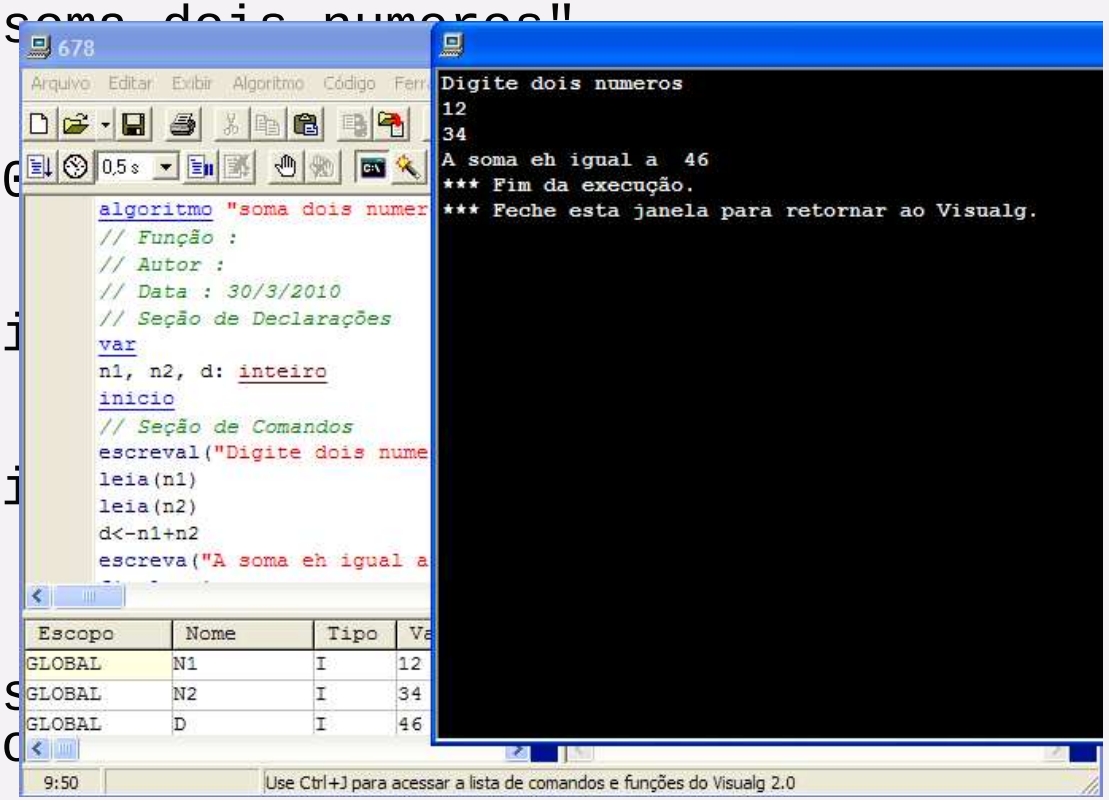
- **Algoritmo em Pseudocódigo**

```
ALGORITMO “SOMA DOIS NÚMEROS”
DECLARE N1, N2, S NUMÉRICO
ESCREVA “Digite dois números”
LEIA N1, N2
S ← N1 + N2
ESCREVA “SOMA = “ , S
FIM_ALGORITMO
```

Introdução a Algoritmos

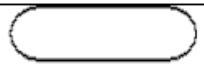
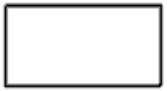
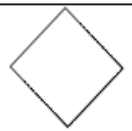
Exemplo – Pseudocódigo (Visualg)

```
Algoritmo "s
// Função :
// Autor :
// Data : 30
// Seção de
var
n1, n2, d: i
inicio
// Seção de
escreval("Di
leia(n1)
leia(n2)
d<-n1+n2
escreva("A s
fimalgoritmo
```



Introdução a Algoritmos

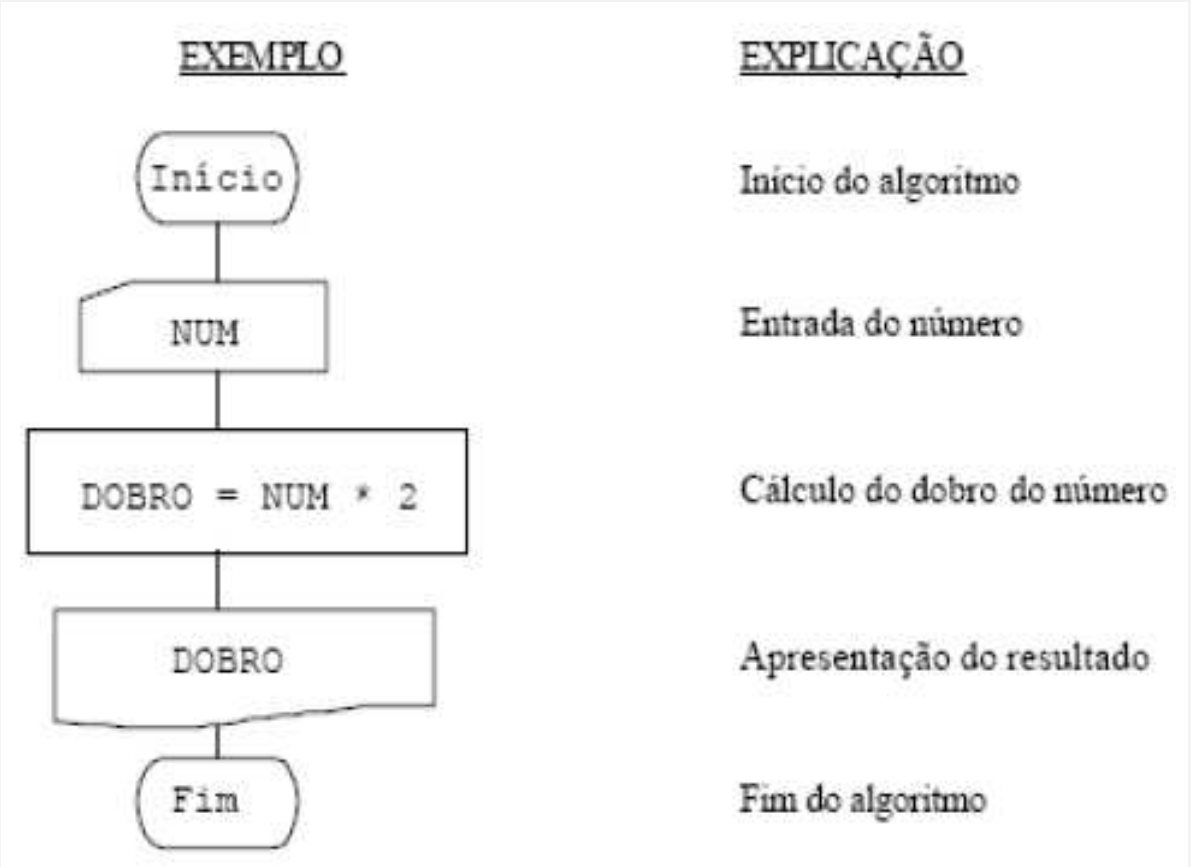
- **Fluxograma**
  - Vantagem – a representação gráfica é mais concisa que a representação textual.
  - Desvantagem – é necessário aprender a simbologia dos fluxogramas

FIGURA	SIGNIFICADO
	Figura para definir início e fim do algoritmo
	Figura usada no processamento de cálculo, atribuições e processamento de dados em geral
	Figura utilizada na representação de entrada de dados
	Figura utilizada para representação da saída de dados
	Figura que indica o processo seletivo ou condicional, possibilitando o desvio no caminho do processamento
	Símbolo geométrico usado como conector
	Símbolo que identifica o sentido do fluxo de dados, permitindo a conexão entre as outras figuras existentes

Introdução a Algoritmos

- **Fluxograma**

Exemplo de fluxograma: calcular o dobro de um número



Introdução a Algoritmos

Fluxograma

Solução do problema de trocar a resistência de um chuveiro resolvido com um algoritmo representado em pseudocódigo.

Conforme foi mencionado são impostas regras e é definido um número restrito de ações. Neste caso as ações disponíveis são: pegar, largar, abrir, fechar, retirar e colocar.

Descrição Narrativa

Adquira uma resistência nova e localize o chuveiro a ser manipulado. Em seguida abra o chuveiro retirando a resistência defeituosa, coloque a resistência nova e feche o chuveiro.
Após descarte a resistência defeituosa.

Pseudocódigo

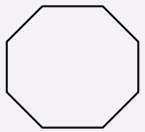
1. **Pegar (resistência nova);**
2. **Pegar (chuveiro);**
3. **Abrir (chuveiro);**
4. **Retirar (resistência def.);**
5. **Colocar (resistência nova);**
6. **Fechar (chuveiro);**
7. **Largar (resistência def.).**

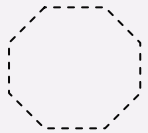
Introdução a Algoritmos

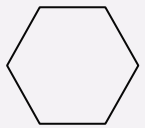
- **Fluxograma**

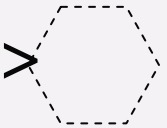
Solução do problema de trocar a resistência de um chuveiro resolvido com um algoritmo representado em fluxograma.

Conforme foi mencionado são impostas regras e é definido um número restrito de ações/símbolos. Neste caso as ações disponíveis e os símbolos a elas associados são:

pegar <=> 

largar <=> 

fechar <=> 

abrir <=> 

colocar <=> 

retirar <=> 

Introdução a Algoritmos

Descrição Narrativa

Adquira uma resistência nova e localize o chuveiro a ser manipulado. Em seguida abra o chuveiro retirando a resistência defeituosa, coloque a resistência nova e feche o chuveiro.
Após descarte a resistência defeituosa.

Ações/Símbolos

pegar <=>

largar <=>

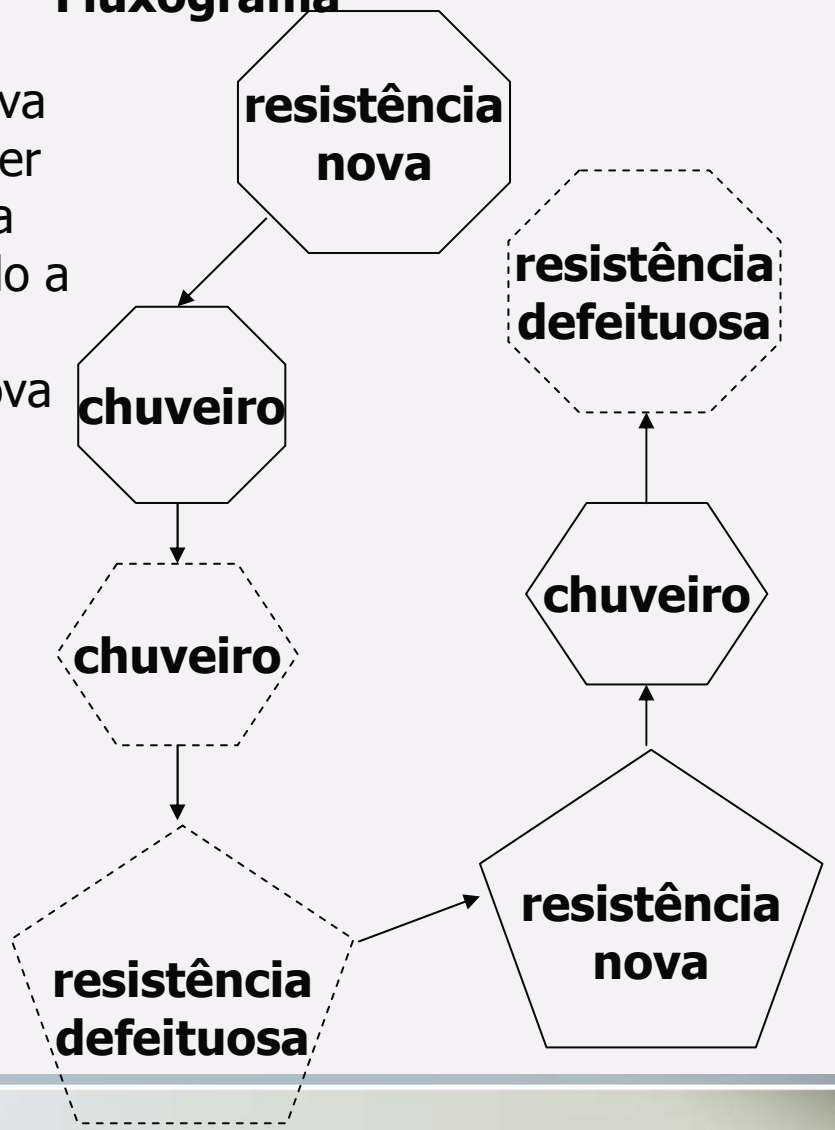
fechar <=>

abrir <=>

colocar <=>

retirar <=>

Fluxograma



Introdução a Algoritmos

Melhorando um algoritmo

- Tomando como exemplo uma operação simples de trocar uma lâmpada, podemos perceber uma série de detalhes que nos ajudam a compreender melhor o uso de um algoritmo e ainda perceber que podemos torná-lo mais eficiente.

ALGORITMO 1.1 Troca de lâmpada

- pegar uma escada;
- posicionar a escada embaixo da lâmpada;
- buscar uma lâmpada nova;
- subir na escada;
- retirar a lâmpada velha;
- colocar a lâmpada nova.

- E se a lâmpada não estivesse queimada?

Introdução a Algoritmos

- **Refinamento de um algoritmo**
- Solução – Efetuar um teste. Uma nova versão do algoritmo seria:

ALGORITMO 1.2 Troca de lâmpada com teste

- pegar uma escada;
- posicionar a escada embaixo da lâmpada;
- buscar uma lâmpada nova;
- acionar o interruptor;
- se a lâmpada não acender, então
  - subir na escada;
  - retirar a lâmpada queimada;
  - colocar a lâmpada nova.

- Embora correto, o algoritmo ainda pode ser melhorado. Como?

Introdução a Algoritmos

- **Refinamento de um algoritmo**
- Testar antes de pegar a escada e a lâmpada nova.

ALGORITMO 1.3 Troca de lâmpada com teste no início

- acionar o interruptor;
- se a lâmpada não acender, então
  - pegar uma escada;
  - posicionar a escada embaixo da lâmpada;
  - buscar uma lâmpada nova;
  - acionar o interruptor;
  - subir na escada;
  - retirar a lâmpada queimada;
  - colocar a lâmpada nova.

- E se a lâmpada nova não funcionar?

Introdução a Algoritmos

- **Refinamento de um algoritmo**
- Usar um teste seletivo com repetição

ALGORITMO 1.4 Troca de lâmpada com teste e repetição indefinida

- acionar o interruptor;
- se a lâmpada não acender, então
  - pegar uma escada;
  - posicionar a escada embaixo da lâmpada;
  - buscar uma lâmpada nova;
  - acionar o interruptor;
  - subir na escada;
  - retirar a lâmpada queimada;
  - colocar a lâmpada nova;
  - se a lâmpada não acender, então
    - retirar a lâmpada queimada;
    - colocar outra lâmpada nova;
    - se a lâmpada não acender, então
      - retirar a lâmpada queimada;
      - colocar outra lâmpada nova;

- ... E não para mais?

Introdução a Algoritmos

- **Refinamento de um algoritmo**
- Usar uma condição de parada.

ALGORITMO 1.5 Troca de lâmpada com teste e condição de parada

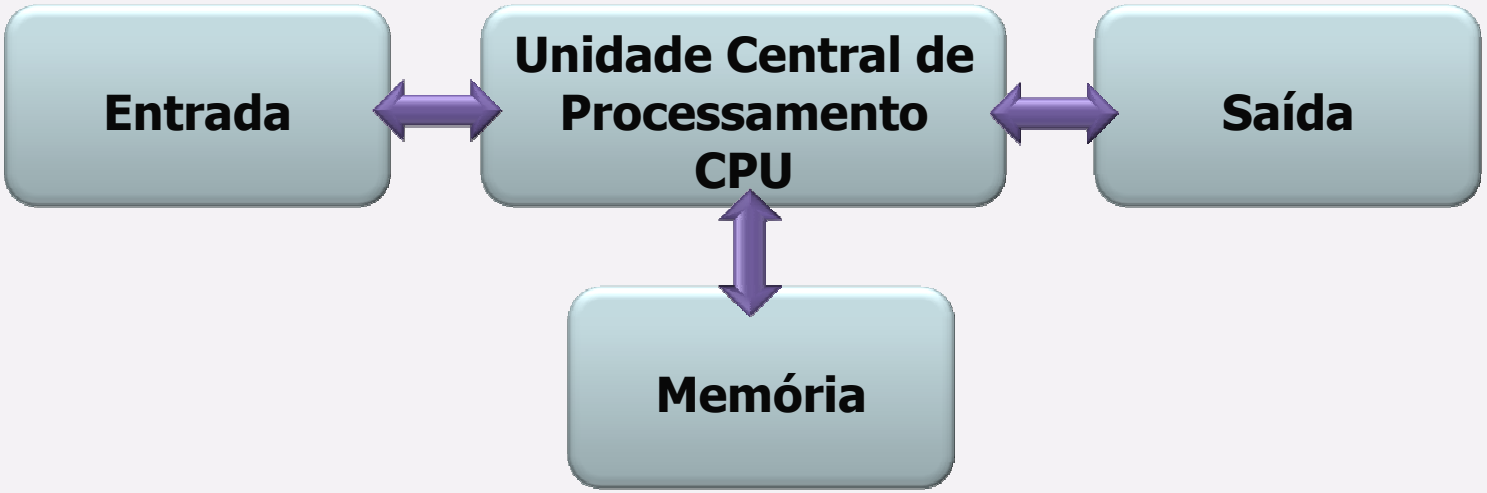
- acionar o interruptor;
- se a lâmpada não acender, então
  - pegar uma escada;
  - posicionar a escada embaixo da lâmpada;
  - buscar uma lâmpada nova;
  - acionar o interruptor;
  - subir na escada;
  - retirar a lâmpada queimada;
  - colocar uma lâmpada nova;
  - enquanto a lâmpada não acender, faça
    - retirar a lâmpada queimada;
    - colocar uma lâmpada nova;

- O número de repetições é indefinido, porém é finito.

Introdução a Algoritmos

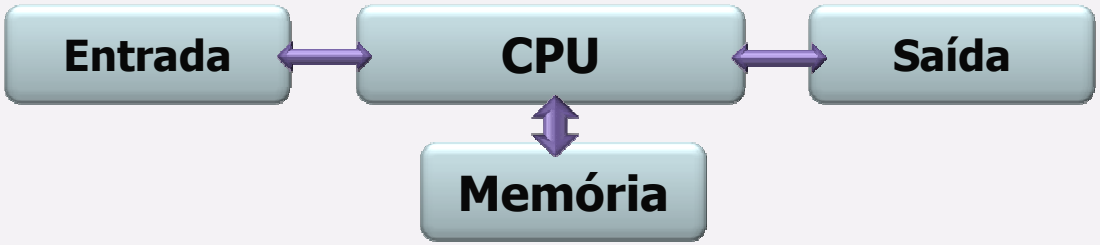
- **Conceitos básicos para construção de algoritmos.**

Modelo de Von Neumann



Introdução a Algoritmos

Modelo de Von Neumann



- Os dados e os programas são armazenados na memória, em regiões distintas.
- Os programas são formados, essencialmente, por comandos (instruções sobre o que fazer);
- Os comandos são lidos sequencialmente da memória, um após o outro;
- A execução de um novo comando inicia apenas depois que a execução do anterior tiver terminado (execução sequencial).
- Eventualmente, um comando pode modificar o valor de um dado existente na memória, solicitar novos dados ao usuário ou enviar dados para a saída.

Introdução a Algoritmos

- **Conceitos básicos para construção de algoritmos**
  - Constantes;
  - Variáveis;
  - Identificadores;
  - Palavra-reservada.
- **Constantes**: São valores fixos, que não podem ser alterados pelas instruções do algoritmo, ou seja, é um espaço de memória cujo valor não deve ser alterado durante a execução de um algoritmo.

Exemplos:

Inteiro → 10, -23768, ...

Real → -2.34, 0.149, ...

Caractere → "k", "computador"

Introdução a Algoritmos

Conceitos básicos (cont...)

- **Variáveis:** Elementos de dado cujo valor pode ser modificado ao longo de sua execução.
- São espaços alocados na memória que recebeu um nome (identificador) e pode ter tipo (inteiro, caractere e real), armazena um valor que pode ser modificado durante a execução do algoritmo.
- Um programa pode manipular várias variáveis distintas;
- Cada variável pode armazenar vários valores, mas apenas um de cada vez;
- "Variáveis" são criadas no início da execução do programa e destruídas ao término da sua execução;
- O conjunto de "variáveis" que um programa necessita precisa ser definido antes de se iniciar a execução do programa.

Introdução a Algoritmos

- **Conceitos básicos (cont...)**
- **Regras para criar nomes de variáveis.**
  - Os nomes das variáveis devem representar o que será guardado dentro dela.
  - O primeiro caractere de um nome deverá ser sempre alfabético.
  - Não podem ser colocados espaços em branco no nome de variáveis, usar o UNDERSCORE "_".
  - A declaração de uma variável é feita no algoritmo informando o seu nome, seguido por : e terminado com o seu tipo.

Exemplos:

a: inteiro
nome_do_aluno: caractere
sinalizador: logico
valor1, valor2: real

Introdução a Algoritmos

- **Conceitos básicos (cont...)**
- **Palavras reservadas (palavras-chave):**
  - São identificadores predefinidos que possuem significados especiais para o interpretador do algoritmo.

inicio	**senao**	**para**	**repita**
var	**logico**	**se**	**ate**
faca	**inteiro**	**caractere**	**real**

  - Algumas das palavras reservadas definem os **tipos de dados.**

Introdução a Algoritmos

- **Conceitos básicos (cont...)**
- **Tipos de dados**
  - Toda "variável" precisa estar associada a algum "tipo" de dados;
  - O "tipo" de uma variável determina a coleção finita de valores que podem ser atribuídos à mesma;
  - O "tipo" de uma variável é fixo durante toda a execução do programa.
  - Os "tipos" de todas as "variáveis" precisam ser definidos antes de se iniciar a execução do programa.

Introdução a Algoritmos

- **Conceitos básicos (cont...)**
- **Tipos Primitivos**
  - **logico** - define variáveis do tipo booleano, ou seja, com valor VERDADEIRO ou FALSO.
  - **caractere** – define variáveis do tipo string, ou seja, cadeia de caracteres.
  - **inteiro** - define variáveis numéricas do tipo inteiro, ou seja, sem casas decimais.
  - **real** - define variáveis numéricas do tipo real, ou seja, com casas decimais.

Introdução a Algoritmos

- **Conceitos básicos (cont...)**
- **Declarações**
  - Seqüência de instruções que servem para informar quais variáveis estarão sendo usadas pelo programa e quais os seus respectivos tipos;
  - Não é possível mudar o tipo de uma variável durante a execução do programa;
  - Não é possível criar ou destruir variáveis durante a execução do programa;
  - Tudo precisa ser planejado antes – durante a elaboração do algoritmo.

Introdução a Algoritmos

- **Conceitos básicos (cont...)**
- **Comandos**
  - Determinam quando e quais ações "primitivas" devem ser executadas;
  - São exemplos de ações "primitivas": leitura de dados, saída de dados, atribuição de valor a uma variável;
  - Além disso, os comandos podem ser "estruturados";
  - A "estruturação" dos comandos permite que eles sejam executados numa determinada ordem, que a sua execução seja repetida ou que se opte pela escolha de um ou outro comando subordinado.
  - Basicamente, a "estruturação" dos comandos permite o estabelecimento de um "fluxo de controle", ou seja, uma seqüência de execução de ações primitivas através do qual se pretende alcançar a solução do problema original.