

# Introdução aos Algoritmos

Prof. Afonso Paiva

ICMC-USP

# Algoritmos

- Seqüência **finita** e **ordenada** (de forma lógica ) de instruções para resolver um problema.
- Exemplos de algoritmos:
  - receitas de bolo;
  - manuais técnicos;
  - guias de montagem;
  - programas de computador.

# Exemplo: Bolinho de Chuva

*Em uma tigela, bata o açúcar, a manteiga e o ovo.*

*Em outro recipiente misture a farinha, o fermento, a canela, uma pitada de sal, o leite e a outra mistura. Misture bem.*

*Aqueça óleo e pingue colheradas da massa, fritando os bolinhos até dourar.*

*Escorra bem, polvilhe açúcar e sirva.*



# Propriedades de um Algoritmo

- **Garantia de término:** o problema a ser resolvido possui condições específicas que, quando satisfeitas, a execução do algoritmo é encerrada.
- **Exatidão:** a intenção de cada instrução de um algoritmo deve ser suficientemente clara.
- **Efetividade:** cada instrução deve ser básica o suficiente para ser executada.

## Exemplo: Máximo Divisor Comum (MDC)

1. Chame o maior número de ***a*** e o menor de ***b***;
2. Divida ***a*** por ***b*** e chame o resto de ***r***;
3. Se ***r*** é igual a zero então o MDC é igual a ***b*** e a execução das instruções encerra aqui. Caso contrário, siga para a próxima instrução.
4. Atribua o valor de ***b*** a ***a*** e o valor de ***r*** a ***b***;
5. Volte para a instrução 2.

# Exemplo: Máximo Divisor Comum (MDC)

- **Instrução 1:**  $a = 12$  e  $b = 8$ ;
- **Instrução 2:**  $r = 4$ , pois 4 é o resto da divisão de  $a = 12$  por  $b = 8$ ;
- **Instrução 3:** como  $r \neq 0$ , devemos executar a instrução 4;
- **Instrução 4:**  $a = b = 8$  e  $b = r = 4$ ;
- **Instrução 5:** devemos executar a instrução 2;
- **Instrução 2:**  $r = 0$ , pois 0 é o resto da divisão de  $a = 8$  por  $b = 4$ ;
- **Instrução 3:** como  $r = 0$ , devemos parar de executar o algoritmo e acusar como resultado o valor de  $b$  (isto é, 4) como MDC de 12 e 8.

# Exemplos

- **Instrução:** divida  $x$  por  $y$  se todo número inteiro par maior que 2 é a soma de dois números primos.
  - Conjectura de Goldbach (1742)
  - viola a propriedade de efetividade
- **Instrução:** escreva todos os números ímpares.
  - viola a propriedade de garantia de término

# Exemplo: Troca de lâmpada

- Pegar uma escada;
- Posicionar a escada embaixo da lâmpada;
- Buscar uma lâmpada nova;
- Retirar a lâmpada velha;
- Colocar a lâmpada nova.

**Estrutura seqüencial**



# Exemplo: 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**
  - ✓ Retirar a lâmpada velha;
  - ✓ Colocar a lâmpada nova.

**Estrutura seqüencial/condicional**

# Exemplo: 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;
  - Retirar a lâmpada velha;
  - Colocar a lâmpada nova.

**Estrutura seqüencial / condicional**

# Exemplo: Troca de lâmpada com teste e repetição

- Acionar o interruptor
- **Se** a lâmpada não acender, **então**:
  - Pegar uma escada;
  - Posicionar a escada embaixo da lâmpada;
  - Buscar uma lâmpada nova;
  - Retirar a lâmpada velha;
- Colocar a lâmpada nova.
- **Se** a lâmpada não acender, **então**:
  - Buscar uma lâmpada nova;
  - Retirar a lâmpada velha;
  - Colocar outra lâmpada nova;
  - **Se** a lâmpada não acender, **então**:
    - Buscar uma lâmpada nova;
    - Retirar a lâmpada velha; **(Até quando??)**

# Exemplo: 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;
  - Retirar a lâmpada velha;
  - Colocar a lâmpada nova.
  - **Enquanto** a lâmpada não acender, **faça**:
    - Retirar a lâmpada velha;
    - Colocar outra lâmpada nova.

**Estrutura seqüencial/condicional/de repetição**

# Estruturas do Algoritmo

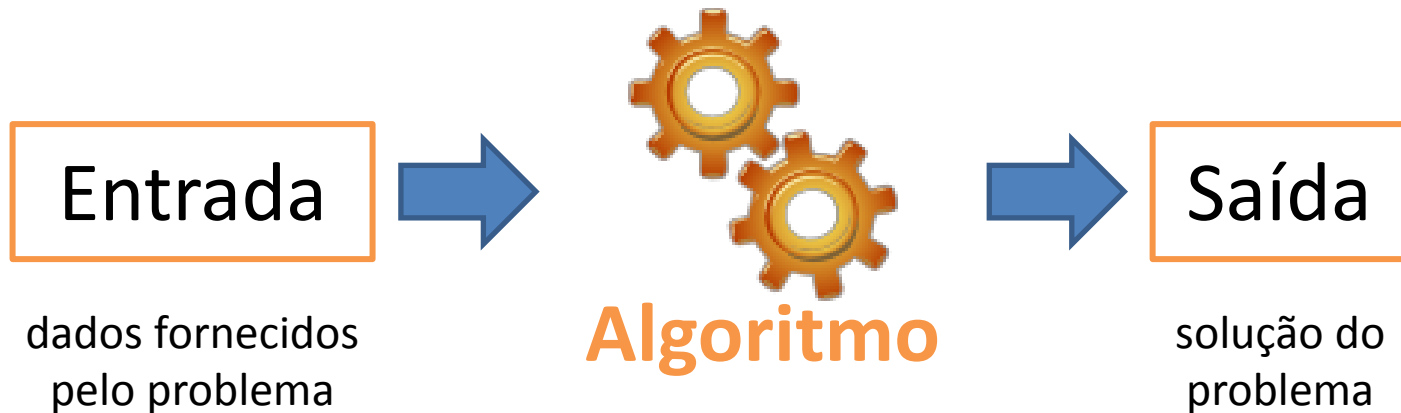
- **Estrutura seqüencial**
  - Os passos são tomados em uma seqüência pré-definida.
  - Ex.: Passos (instruções) do algoritmo
- **Estrutura condicional**
  - Permite a escolha do grupo de ações a ser executado quando determinada condição é ou não satisfeita.
  - Ex.: **Se** a lâmpada não acender **então**
- **Estrutura de repetição**
  - Permite que uma seqüência de comandos seja executada repetidamente até que uma determinada condição de parada seja satisfeita.
  - Ex.: **Enquanto** a lâmpada não acender, **faça**

# Atenção: o diferencial

- Além de estar corretos, os algoritmos devem resolver o problema com **menos esforço** e **maior objetividade** possível.
- O desenvolvimento de algoritmos é um exercício de:
  - criatividade;
  - experiência.
- Apesar de corretos, um algoritmo pode ser melhor do que outro!!!

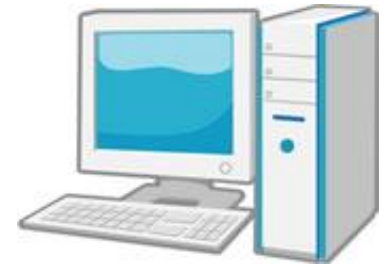
# Problemas Computacionais

- Um problema para o qual existe uma solução pode ser encontrada através de um algoritmo é dito um **problema computacional**.



# Algoritmos Computacionais

- O computador deve executar a tarefa;
- Precisamos de uma linguagem de programação estruturada:
  - seqüência, decisão e repetição
- É preciso transformar a idéia (receita) em um programa (codificação).





# Algoritmos Computacionais

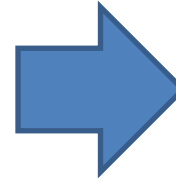
**Algoritmo <nome>**

**início**

**<identificador>**

**<declarações>**

**<comandos>**



Constantes,  
Tipos e  
Variáveis

**fim**

# Tipos de Dados: Numérico

- **Inteiro (`int`):** representa um número inteiro;
  - Exemplos: -35, -20, 0, 7, 14, 34
  - Podem ser usados para idade em anos, número de filhos, etc...
- **Real (`float`):** representa um número real. Também é conhecido como *ponto flutuante*.
  - Exemplos: 3.1415, -234.46, 45.15
  - Podem ser usados para saldo bancário, altura, peso, temperatura, etc...

# Tipos de Dados: Lógico

- Dados lógicos podem assumir apenas dois valores: **verdadeiro** (true ou 1) ou **falso** (false ou 0).
- Também conhecido como *booleano* (**bool**).
- São usados para expressar uma condição:
  - $4 > 5$  é falso;
  - se o cheque número 000123 já foi compensado, ou não.

# Comando de Atribuição

- Pode-se atribuir (ou definir) um valor a uma variável através do operador “←”.

- Sintaxe:

```
identificador ← valor
```

- Exemplos:

– Nome ← “Jose”

– x ← 10 (lê-se: a variável x recebe o valor 10)

# Bloco de Execução

O próprio algoritmo é um bloco de execução. A sintaxe da definição do bloco de um algoritmo é:

```
Algoritmo <Nome_do_Algoritmo>
```

```
início
```

```
    <declaração de variáveis>
```

```
    <comandos>
```

```
fim
```

# Exemplo: Área de um Círculo

Algoritmo Area\_Circulo

{Algoritmo para calcular a área de um círculo}

início

variável:

raio: real      {dado de entrada}

pi: real        {constante}

area: real     {dado de saída}

leia(raio)

pi ← 3.1416

area ← pi\*raio\*raio

escreva(area)

fim

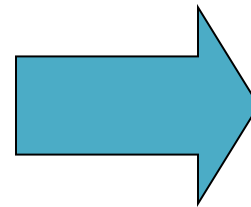
# Estrutura de Controle

- Estruturas de controle permitem o controle do fluxo de execução dos comandos
- Vamos analisar as seguintes estruturas de controle:
  - ❖ seqüencial;
  - ❖ desvio condicional simples/composto;
  - ❖ repetição com teste no início/final/ variável de controle.

# Estrutura Seqüencial

É um conjunto de comandos que serão executados em uma seqüência linear, de cima para baixo

Os comandos serão executados na mesma ordem em que foram escritos



$C_1;$

$C_2;$

...

$C_n;$

Convencionaremos que os comandos serão seguidos por ponto-e-vírgula (;)



# Modelo Geral de um Pseudo-código:

Algoritmo <nome>

início

< declaração de variáveis >

< tarefa 1 >; {corpo do algoritmo}

< tarefa 2 >;

...

< tarefa n >;

fim

# Exemplo 5

## Algoritmo Perimetro\_Triangulo

{Algoritmo para calcular o perímetro de um triângulo}

início

variável:

a,b,c: real;           {dados de entrada}

perim: real;           {dado de saída}

leia(a,b,c);

perim ← a+b+c;

escreva(perim);

fim

# Decisão (Desvio) Condicional

- Um desvio condicional é usado para decidir se um conjunto de instruções deve, ou não, ser realizado;
- Comandos de **decisão (desvio)** para construir estruturas de algoritmos que *não são* totalmente seqüenciais;
- Com as instruções de **desvio** pode-se fazer com que o algoritmo proceda de uma ou outra maneira, de acordo com as decisões lógicas tomadas em função dos dados ou resultados anteriores.

# Desvio Condicional Simples

No desvio condicional simples uma condição é avaliada e, se o resultado for **verdadeiro**, um conjunto de instruções é executado

## PSEUDO-CÓDIGO

```
se (<condição>) então  
    <tarefa única>;  
fim se
```

**Se a condição for falsa  
encerra-se a seleção!**

## PSEUDO-CÓDIGO

```
se (<condição>) então  
    <tarefa 1>;  
    <tarefa 2>;  
    ...  
    <tarefa n>;  
fim se
```

# Exemplo 6

## Algoritmo Numero\_Positivo

{Algoritmo para verificar se um número é positivo}

início

variável:

x: real;

leia(x);

se ( x > 0 ) então

    escreva("O número", x, " é positivo");

fim se

fim

# Desvio Condicional Composto

Quando mais de uma ação depende de uma mesma condição: uma de a condição ser verdadeira e outra de a condição ser falsa.

## PSEUDO-CÓDIGO

```
se (<condição>) então  
  <tarefa 1>;  
senão  
  <tarefa 2>;  
fim se
```

# Exemplo 7

## Algoritmo Calculo\_da\_media\_final

{Algoritmo para calcular a média final dos alunos, baseado nas notas de provas e listas}

início

variável

```
P1,P2,L: real;           {Dados de entrada}
MF: real;                {Dados de saída}
leia(P1,P2,L);          {Lendo os dados}
MF ← (P1 + P2 + L)/3;   {Cálculo da média final}
```

se (MF < 7.0) então

```
    escreva ("Reprovado com média final", MF);
```

senão

```
    escreva ("Aprovado com média final", MF);
```

fim se

fim

# Desvio Condicional Composto

## PSEUDO-CÓDIGO

```
se (<condição>) então  
  <tarefa V1>;  
  ... {Bloco Verdade}  
  <tarefa Vn>;  
senão  
  <tarefa F1>;  
  ... {Bloco Falsidade}  
  <tarefa Fn>;  
fim se
```



# Exemplo 8

## Algoritmo Calculo\_da\_media\_final

{Algoritmo para calcular a média final dos alunos, baseado nas notas de provas e listas}

início

variável

```
P1,P2,L: real ;           {Dados de entrada}
MF: real;                 {Dados de saída}
leia(P1,P2,L);           {Lendo os dados}
MF ← (P1 + P2 + L)/3;    {Cálculo da média final}
se (MF < 7.0) então
    escreva ("Reprovado com média final", MF);
    escreva ("Terá que estudar mais");
senão
    escreva ("Aprovado com média final", MF);
    escreva ("Pode ir para praia relaxar");
fim se
```

fim

# Desvio Condicional: **se-então-se**

## PSEUDO-CÓDIGO

```
se (<condição 1> ) então  
  se (<condição 2>) então  
    se (<condição 3>) então  
      se (<condição 4>) então  
        <tarefa 1>;  
      fim se  
    fim se  
  fim se  
fim se
```

Não há **senão** após então!

A ação vai ocorrer quando todas as condições forem ao mesmo tempo satisfeitas!

# Operadores Lógicos

- Proposições compostas: geradas a partir da combinação de proposições simples, através do uso de conectivos lógicos.

- Conectivos Lógicos:

- ❖ e (and):  $\wedge$

- ❖ ou (or):  $\vee$

- ❖ não (not):  $\sim$

- ❖ **Atenção aos símbolos!!!**

# Conectivo “E” (AND)

● **Exemplo:** Se chover **e** relampejar, eu fico em casa.

● Quando eu fico em casa?

❖  $p$ : Está chovendo

❖  $q$ : Está relampejando

❖  $p \text{ e } q \leftrightarrow p \wedge q$

Conectivo E Conjunção		
$p$	$q$	$p \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

# Conectivo “OU” (OR)

- **Exemplo:** Se acabar café **ou** acabar o açúcar, irei ao mercado.

- Quando irei ao mercado?

- ❖  $p$ : Acabou o café

- ❖  $q$ : Acabou o açúcar

- ❖  $p \text{ ou } q \leftrightarrow p \vee q$

Conectivo OU Disjunção		
$p$	$q$	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

# Conectivo “Não” (NOT)

Conectivo NÃO Negação	
p	$\sim p$
V	F
F	V

## Exemplo:

- p: O Sol é uma estrela
- $\sim p$ : O Sol **NÃO** é uma estrela

# Desvio Condicional: **se-então-se**

## PSEUDO-CÓDIGO

```
se (<condição 1> ) então  
  se (<condição 2>) então  
    se (<condição 3>) então  
      se (<condição 4>) então  
        <tarefa 1>;  
      fim se  
    fim se  
  fim se  
fim se
```

Não há **senão** após então!

A ação vai ocorrer quando todas as condições forem ao mesmo tempo satisfeitas!

# Desvio Condicional: **se-então-se**

## PSEUDO-CÓDIGO

```
se (<condição1>^<condição2>^<condição3>^<condição4>) então  
  <tarefa 1>;  
fim se;
```

Condição 1	Condição 2	Condição 3	Condição 4
V	V	V	V



**Executa a tarefa**



# Desvio Condicional: **se-senão-se**

## PSEUDO-CÓDIGO

```
se ( X = V1 ) então  
  <tarefa 1>;  
senão  
  se ( X = V2 ) então  
    <tarefa 2>;  
  senão  
    se ( X = V3 ) então  
      <tarefa 3>;  
    senão  
      se ( X = V4 ) então  
        <tarefa 4>;  
      fim se  
    fim se  
  fim se  
fim se
```

Após cada **senão**, existe outro comando **se**.

Depois do **então**, existe um comando.

# Desvio Condicional Heterogêneo

Quando **não conseguimos identificar** um padrão lógico de construção de um estrutura de seleção encadeada

## PSEUDO-CÓDIGO

```
se (<condição 1>) então  
  se (<condição 2>) então  
    <tarefa 1>;  
  
  senão  
    se (<condição 3>) então  
      <tarefa 2>;  
  
    senão  
      <tarefa 3>;  
  
    fim se  
  fim se  
  
  senão  
    <tarefa 4>;  
  
  fim se
```

# Estrutura de Repetição

- Repetição com teste no início
  - **enquanto** <condição> **faça**
- Repetição por contagem
  - **para** V **de** Vi **até** Vf **passo** p **faça**

# Repetição com Teste no Início

- Permite repetir várias vezes um mesmo trecho do algoritmo, porém sempre verificando antes de cada execução se é permitido executar o trecho, ou seja, enquanto o valor da **condição for verdadeiro**.
- Ex.: Lembra do algoritmo de trocar um pneu?

enquanto <houver parafuso p desapertar> faça

Desparafusar a roda;

fim do enquanto

## PSEUDOCÓDIGO

enquanto <condição> faça

< tarefa 1 >;

< tarefa 2 >;

...

< tarefa n >;

fim do enquanto

# Definições

- **Contador**: representado por uma variável com um dado valor inicial, o qual é incrementado a cada repetição.
- **Incrementar**: mesmo que somar um valor constante.
- Exemplo:

início

variável

```
cont: inteiro;           {declaração do contador}
cont ← 1;                {inicializando o contador}
cont ← cont+1;          {incrementando o contador}
```

fim

**Pode se incrementar mais que um !!!**

# Exemplo 12

Algoritmo ler\_e\_somar\_n\_numeros

{Algoritmo para ler N números naturais e exibir a soma desses números.}

início

variável

cont, n, soma: inteiro;

cont ← 1;                    {Inicializando o contador}

leia(n);                    {Lendo os dados}

soma ← 0;                    {Inicializando a soma}

enquanto (cont ≤ n) faça

    soma ← soma + cont;

    cont ← cont + 1;    {Incrementar o contador em um}

fim do enquanto

escreva ("A soma de", n, " números é ", soma);

fim

# Repetição por Contagem

- Utilizaremos a estrutura:

para V de Vi até Vf faça

- Sempre repete a execução do bloco um número predeterminado de vezes. Possui limites de fluxos.
  - **V:** *variável*
  - **Vi:** *valor inicial*
  - **Vf:** *valor final*

## PSEUDO-CÓDIGO

```
para V de Vi até Vf faça  
    <tarefa 1>;  
    <tarefa 2>;  
    ...  
    <tarefa n>;  
fim para
```

# Repetição por Contagem

- Outra forma de estrutura:

para V de vi até vf passo p faça

- p: valor do *incremento* dado a variável V

## PSEUDO-CÓDIGO

para V de Vi até Vf passo p faça

<tarefa 1>;

<tarefa 2>;

...

<tarefa n>;

fim para



# Exemplo 13

Algoritmo ler\_e\_somar\_20\_numeros

{Algoritmo para somar 20 números naturais e exibir a soma desses números.}

início

variável

V, soma: inteiro;

cont ← 1;                    {Inicializando o contador}

soma ← 0;                    {Inicializando a soma}

para V de 1 até 20 (passo 1) faça

    soma ← soma + V;

fim para

    escreva ("A soma de 20 números é ", soma);

fim