

```
ANÁLISE DE ALGORITMOS

Definimos um conjunto de operações primitivas de alto nível que são independentes da linguagem de programação usada e podem ser identificadas no pseudocódigo;
As seguintes operações estão incluídas entre as operações primitivas:
atribuição de valores a variáveis
chamadas de métodos
operações aritméticas
comparação de dois números
acesso a um arranjo
seguir uma referência a um objeto
retorno de um método
```

```
CONTANDO OPERAÇÕES
Algoritmo maiorElemento(V, n)
                                           # operações
  maior ← V[0]

para i de 1 até n - 1 faça

se (V[i] > maior) então

maior ← V[i]
                                           2 (n - 1)
2 (n - 1)
       fimse
                                           2(n - 1)
       {incrementa contador i}
   fimpara
   retorna maior
                                           Total 7n - 2
■ Número mínimo de operações:
      2 + 1 + n + 4(n - 1) + 1 = 5n
□ Número máximo de operações:
      2 + 1 + n + 6(n - 1) + 1 = 7n - 2
                      Estrutura de Dados
Prof. Ademar Schmitz, M.Sc
```

### ANÁLISE DO CASO MÉDIO E DO PIOR CASO

- Um algoritmo pode ser mais rápido sobre algumas entradas do que sobre outras.
- □ Podemos desejar expressar o tempo de execução desse algoritmo como uma média calculada com todas as entradas possíveis (caso médio).
- ☐ Uma análise de caso médio requer tipicamente que sejam calculados os tempos de execução baseados em uma distribuição de probabilidade.
- Portanto, a não ser que especificado de outra forma, caracterizaremos os tempos de execução em termos de pior caso.

 Estrutura de Dados

 17/2/2008
 Prof. Ademar Schmitz, M.Sc.
 13

### NOTAÇÃO ASSINTÓTICA

- □ Na análise de algoritmos, é importante concentrar-se na taxa de crescimento do tempo de execução com uma função do tamanho da entrada n, obtendo-se uma quadro geral do comportamento, em vez de concentrarse nos detalhes menores.
- ☐ Freqüentemente basta saber que o tempo de execução de uma algoritmo como maiorElemento cresce proporcionalmente a n, como o verdadeiro tempo de execução sendo n vezes algum pequeno fator constante que depende do ambiente de hardware e software e varia em faixa de valores dependendo da entrada específica.

Estrutura de Dados 17/2/2008 Prof. Ademar Schmitz, M.Sc.

### NOTAÇÃO O (BIG-O)

- □ Sejam f(n) e g(n) funções mapeando inteiros não-negativos em números reais.
  - Dizemos que f(n) é O(g(n)) se existe uma constante real c > 0 e uma constante inteira  $n_0 \ge 1$  tais que f(n) ≤ cg(n) para todo inteiro n  $\ge n_0$ .
  - Esta definição é geralmente chamada de notação **big-O**, pois geralmente se diz "f(n) é big-O de g(n)".

 Estrutura de Dados

 17/2/2008
 Prof. Ademar Schmitz, M.Sc.
 15

### EXEMPLO: $7n - 2 \notin O(n)$

- □ **Justificativa:** pela definição da notação O, precisamos achar uma constante c > 0 e uma constante inteira  $n_0 > 1$  tais que  $7n 2 \le cn$  para todo inteiro  $n \ge n_0$ : c = 7 e  $n_0 = 1$ .
- □ Esta é uma das infinitas escolhas possíveis, pois qualquer número real maior ou igual a 7 será um escolha possível para c e qualquer inteiro maior ou igual a 1 é uma escolha possível para n₀.
- □ Podemos dizer que o tempo de execução do algoritmo maiorElemento para determinar o maior elemento de uma arranjo de n inteiros é O(n).

7/2/2008 Estrutura de Dados Prof. Ademar Schmitz, M.Sc. 16

### **FUNÇÕES COMUNS**

- Algumas funções aparecem com freqüência na análise de algoritmos e estruturas de dados, e seguidamente usamos termos especais para nos referir a elas.
- A tabela abaixo descreve este e outros termos comumentemente usados na análise de algoritmos.

O(log n)	O(n)	Quadrática O(n²)	Polinomial $O(n^k)$ com $k \ge 1$	Exponencial O(a <sup>n</sup> ) com a > 1	
17/2/2008		Estrutura o Prof. Ademar S			

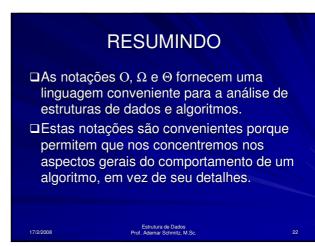
### CRESCIMENTO DE VÁRIAS FUNÇÕES

n	log n	$\sqrt{n}$	n	n log n	n²	n³	2 <sup>n</sup>
2	1	1,4	2	2	4	8	4
4	2	2	6	8	16	64	16
8	3	2,8	8	24	64	512	256
16	4	4	16	64	256	4.096	65.536
32	5	5,7	32	160	1.024	32.768	4.294.967.296
64	6	8	64	384	4.096	262.144	1,84 x 10 <sup>19</sup>
128	7	11	128	896	16.384	2.097.152	3,40 x 10 <sup>38</sup>
256	8	16	256	2.048	65.536	16.777.216	1,15 x 10 <sup>77</sup>
512	9	23	512	4.608	262.144	134.217.728	1,34 x 10 <sup>154</sup>
1.024	10	32	1.024	10.240	1.048.576	1.073.741.824	1,79 x 10 <sup>308</sup>

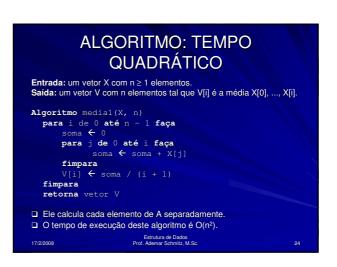
# PARENTES DE BIG-O □ A notação Big-O fornece uma maneira assintótica de dizer que uma função é "menor ou igual a" outra função. □ Outras notações fornecem maneiras assintóticas de fazer outros tipos de comparações. Estrutura de Dados Prof. Ademar Schmitz. M.Sc. 19

# PARENTES DE O (BIG-ÔMEGA) □ Sejam f(n) e g(n) funções mapeando número inteiros em números reais. - Dizemos que f(n) é Ω(g(n)) (dito "f(n) é ômega de g(n)") se g(n) é O(f(n)). - Ou seja, se existe uma constante c > 0 e uma constante inteira n₀ ≥ 1 tais que f(n) ≥ cg(n) para n ≥ n₀. □ Esta função nos permite dizer que uma função é assintoticamente maior que ou igual a outra, exceto por uma fator constante.

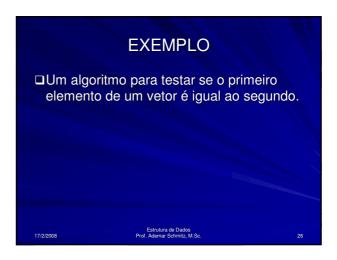




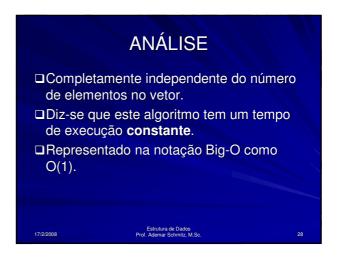
# EXEMPLO: MÉDIA PREFIXADA □ Se tivermos um vetor X armazenando n números, desejamos compor um vetor V tal que V[i] seja a média dos elementos X[0], ..., X[i] para i = 0, ..., n - 1.



# ALGORITMO: TEMPO LINEAR Entrada: um vetor X com n ≥ 1 elementos. Saída: um vetor V com n elementos tal que V[i] é a média X[0], ..., X[i]. Algoritmo media2 (X, n) soma ← 0 para i de 0 até n - 1 faça soma ← soma + X[i] V[i] ← soma / (i + 1) fimpara retorna vetor V Se denotarmos com S, a soma prefixada X[0] + X[1] + ... + X[i], podemos calcular as médias prefixadas como sendo A[i] = S₁/(i + 1). O tempo de execução de média2 é O(n). Estudura de Dados Prof. Ademar Schmiltz, M.Sc. 25



# ALGORITMO Entrada: um vetor V com n ≥ 2 elementos. Saída: mensagem acusando a igualdade ou não dos elementos. Algoritmo primeirosIguais(V, n) se (V[0] = V[1]) então Escreva "são iguais" senão Escreva "não são iguais" fimse fim Estrulura do Dados Piot. Ademir Schmitz, M.Sc. 27



### EXEMPLO Um algoritmo que testa se o primeiro elemento de um vetor é igual a qualquer um dos outros elementos do vetor. Estrutura de Dados Prof. Ademar Schmitz, M.Sc. 29

```
ALGORITMO

Entrada: um vetor V com n ≥ 2 elementos.

Saída: mensagem acusando a igualdade de um elemento do vetor com o primeiro elemento.

Algoritmo temIgual(V, n)

para i de 1 até n - 1 faça

se (V[0] = V[i]) então

Escreva "tem um igual"

fimse

fimpara

fim

Estrutura de Dados

Prof. Ademar Scientiz, M.Sc. 30
```

# ANÁLISE □ Exigirá no máximo n − 1 comparações. Diz-se que um algoritmo que exige um total de n − 1 comparações é O(n). □ Diz-se que um algoritmo O(n) tem um tempo de execução linear.

