Disciplinas CCM-001 / MCTA003-17
Análise de Algoritmos (e Estruturas de Dados)

Primeira aula: introdução e exercícios de revisão

Profa. Carla Negri Lintzmayer

carla.negri@ufabc.edu.br

www.professor.ufabc.edu.br/∼carla.negri

Centro de Matemática, Computação e Cognição – Universidade Federal do ABC

Conteúdo

Estes slides contêm uma motivação (Seção 1) para o estudo desta disciplina e alguns exercícios de revisão (Seção 2), sobre o conteúdo recomendado.

Motivação

Algoritmo

Sequência finita de passos descritos de forma não ambígua que **corretamente resolvem** um problema.

Algoritmo

Sequência finita de passos descritos de forma não ambígua que **corretamente resolvem** um problema.

Recebe um conjunto de dados *válidos* como *entrada* e devolve um conjunto de dados como *saída*.

Algoritmo

Sequência finita de passos descritos de forma não ambígua que **corretamente resolvem** um problema.

Recebe um conjunto de dados *válidos* como *entrada* e devolve um conjunto de dados como *saída*.

Para toda entrada possível ele produz uma saída que seja solução do problema para aquela entrada.

Análise de algoritmos

Nos permite

- verificar corretude,

Análise de algoritmos

Nos permite

- verificar corretude,
- prever desempenho,

Análise de algoritmos

Nos permite

- verificar corretude,
- prever desempenho,
- comparar soluções.

Análise de algoritmos

Nos permite

- verificar corretude,
- prever desempenho,
- comparar soluções.

Sem que seja necessário implementá-los em um dispositivo específico!

Exemplo

A sequência de Fibonacci é a sequência infinita de números:

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$$

Exemplo

A sequência de Fibonacci é a sequência infinita de números:

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$$

Por definição, o n-ésimo número da sequência, escrito como F_n, é dado por

$$F_n = \begin{cases} 1 & \text{se } n = 1 \\ 1 & \text{se } n = 2 \\ F_{n-1} + F_{n-2} & \text{se } n > 2\,. \end{cases} \tag{1}$$

Exemplo

A sequência de Fibonacci é a sequência infinita de números:

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$$

Por definição, o n-ésimo número da sequência, escrito como F_n, é dado por

$$F_n = \begin{cases} 1 & \text{se } n = 1 \\ 1 & \text{se } n = 2 \\ F_{n-1} + F_{n-2} & \text{se } n > 2\,. \end{cases} \tag{1}$$

Problema: Número de Fibonacci

Dado um inteiro $n \geq 1$, encontrar F_n.

Exemplo

```
1: Função FIB1(n)
2:     Se n ≤ 2 então
3:         Devolve 1
4:     Devolve FIB1(n − 1) + FIB1(n − 2)
```

Exemplo

```
1: Função FIB1(n)
2:     Se n ≤ 2 então
3:         Devolve 1
4:     Devolve FIB1(n − 1) + FIB1(n − 2)
```

1. Esse algoritmo resolve o problema?

Exemplo

```
1: Função FIB1(n)
2:     Se n ≤ 2 então
3:         Devolve 1
4:     Devolve FIB1(n − 1) + FIB1(n − 2)
```

1. Esse algoritmo resolve o problema?

2. Quanto tempo ele leva?

Exemplo

Seja “Máquina 1” um computador que faz 4 bilhões de instruções por segundo.

Seja “Máquina 2” um computador que faz 40 trilhões de instruções por segundo.

Exemplo

Seja “Máquina 1” um computador que faz 4 bilhões de instruções por segundo.

Seja “Máquina 2” um computador que faz 40 trilhões de instruções por segundo.

	$n = 10$	$n = 100$	$n = 200$
Máquina 1	$< 1s$	≈ 6174 anos	$\approx 5 \times 10^{21}$ milênios

Exemplo

Seja “Máquina 1” um computador que faz 4 bilhões de instruções por segundo.

Seja “Máquina 2” um computador que faz 40 trilhões de instruções por segundo.

	$n = 10$	$n = 100$	$n = 200$
Máquina 1	$< 1s$	≈ 6174 anos	$\approx 5 \times 10^{21}$ milênios
Máquina 2	$< 1s$	≈ 226 dias	$\approx 5 \times 10^{17}$ milênios

Exemplo

3. Dá para fazer melhor?

Exemplo

3. Dá para fazer melhor?

```
1: Função FIB2(n)
2:     Se n ≤ 2 então
3:         Devolve 1
4:     Seja F[1..n] um vetor de tamanho n
5:     F[1] = 1
6:     F[2] = 1
7:     Para i = 3 até n faça
8:         F[i] = F[i − 1] + F[i − 2]
9:     Devolve F[n]
```

Exemplo

3. Dá para fazer melhor?

```
1: Função FIB2(n)
2:     Se n ≤ 2 então
3:         Devolve 1
4:     Seja F[1..n] um vetor de tamanho n
5:     F[1] = 1
6:     F[2] = 1
7:     Para i = 3 até n faça
8:         F[i] = F[i − 1] + F[i − 2]
9:     Devolve F[n]
```

1. Esse algoritmo resolve o problema?

2. Quanto tempo ele leva?

Exemplo

Seja “Máquina 1” um computador que faz 4 bilhões de instruções por segundo.

Seja “Máquina 2” um computador que faz 40 trilhões de instruções por segundo.

Exemplo

Seja “Máquina 1” um computador que faz 4 bilhões de instruções por segundo.

Seja “Máquina 2” um computador que faz 40 trilhões de instruções por segundo.

	$n = 10$	$n = 100$	$n = 200$
Máquina 1	$< 1s$	$< 1s$	$< 1s$
Máquina 2	$< 1s$	$< 1s$	$< 1s$

Exemplo

Seja “Máquina 1” um computador que faz 4 bilhões de instruções por segundo.

Seja “Máquina 2” um computador que faz 40 trilhões de instruções por segundo.

	$n = 10$	$n = 100$	$n = 200$	$n = 1mi$
Máquina 1	$< 1s$	$< 1s$	$< 1s$	$< 1s$
Máquina 2	$< 1s$	$< 1s$	$< 1s$	$< 1s$

Exemplo

Seja “Máquina 1” um computador que faz 4 bilhões de instruções por segundo.

Seja “Máquina 2” um computador que faz 40 trilhões de instruções por segundo.

	$n = 10$	$n = 100$	$n = 200$	$n = 1mi$	$n = 10bi$
Máquina 1	$< 1s$	$< 1s$	$< 1s$	$< 1s$	$2.5s$
Máquina 2	$< 1s$	$< 1s$	$< 1s$	$< 1s$	$< 1s$

Exemplo

Seja “Máquina 1” um computador que faz 4 bilhões de instruções por segundo.

Seja “Máquina 2” um computador que faz 40 trilhões de instruções por segundo.

	$n = 10$	$n = 100$	$n = 200$	$n = 1mi$	$n = 10bi$
Máquina 1	$< 1s$	$< 1s$	$< 1s$	$< 1s$	$2.5s$
Máquina 2	$< 1s$	$< 1s$	$< 1s$	$< 1s$	$< 1s$

3. Dá para fazer melhor?

O que veremos no curso?

0. Uma vez compreendido um problema, como resolvê-lo?

1. O algoritmo resolve o problema?

2. Quanto tempo ele leva?

3. Dá para fazer melhor?

O que veremos no curso?

0. Uma vez compreendido um problema, como resolvê-lo?

- Técnicas de solução de problemas (divisão e conquista, gulosos, programação dinâmica).
- Outros problemas (redução).
- Complexidade (problemas P, NP e NP-completos).

1. O algoritmo resolve o problema?

2. Quanto tempo ele leva?

3. Dá para fazer melhor?

O que veremos no curso?

0. Uma vez compreendido um problema, como resolvê-lo?

- Técnicas de solução de problemas (divisão e conquista, gulosos, programação dinâmica).
- Outros problemas (redução).
- Complexidade (problemas P, NP e NP-completos).

1. O algoritmo resolve o problema?

- Como demonstrar corretude de algoritmos (prova por indução).

2. Quanto tempo ele leva?

3. Dá para fazer melhor?

O que veremos no curso?

0. Uma vez compreendido um problema, como resolvê-lo?

- Técnicas de solução de problemas (divisão e conquista, gulosos, programação dinâmica).
- Outros problemas (redução).
- Complexidade (problemas P, NP e NP-completos).

1. O algoritmo resolve o problema?

- Como demonstrar corretude de algoritmos (prova por indução).

2. Quanto tempo ele leva?

- Vocabulário e técnicas de análise de algoritmos (notação assintótica e recorrências).

3. Dá para fazer melhor?

O que veremos no curso?

0. Uma vez compreendido um problema, como resolvê-lo?

- Técnicas de solução de problemas (divisão e conquista, gulosos, programação dinâmica).
- Outros problemas (redução).
- Complexidade (problemas P, NP e NP-completos).

1. O algoritmo resolve o problema?

- Como demonstrar corretude de algoritmos (prova por indução).

2. Quanto tempo ele leva?

- Vocabulário e técnicas de análise de algoritmos (notação assintótica e recorrências).

3. Dá para fazer melhor?

- Técnicas de solução de problemas (divisão e conquista, gulosos, programação dinâmica).
- Estruturas de dados (não básicas).

Sobre as aulas

- Conteúdo dado no quadro, com alguns slides de apoio eventualmente
  - Notas de aula/livro estão disponíveis no site
  - Recursos extras estão disponíveis no site

Sobre as aulas

- Conteúdo dado no quadro, com alguns slides de apoio eventualmente
  - Notas de aula/livro estão disponíveis no site
  - Recursos extras estão disponíveis no site
  - Videoaulas não substituem as aulas presenciais!

Sobre as aulas

- Conteúdo dado no quadro, com alguns slides de apoio eventualmente
  - Notas de aula/livro estão disponíveis no site
  - Recursos extras estão disponíveis no site
  - Videoaulas não substituem as aulas presenciais!

- Atendimentos extraclasse: basta aparecer!
  - Qualquer pergunta e feedback são **sempre** bem-vindos
  - Não deixe dúvidas acumularem

Sobre as aulas

- Conteúdo dado no quadro, com alguns slides de apoio eventualmente
  - Notas de aula/livro estão disponíveis no site
  - Recursos extras estão disponíveis no site
  - Videoaulas não substituem as aulas presenciais!

- Atendimentos extraclasse: basta aparecer!
  - Qualquer pergunta e feedback são **sempre** bem-vindos
  - Não deixe dúvidas acumularem

- Espero que você
  - Participe dos atendimentos
  - Seja autor das suas atividades
  - Se divirta :)

Informações

professor.ufabc.edu.br/~carla.negri/cursos/2025Q1-AAED/

- Problemas com o Google Chrome
- Estude bem o conteúdo do site.
- Verifique-o com frequência!

Exercícios de revisão

Apoio

- Os exercícios a seguir foram retirados da lista 0.
- Existem materiais de revisão no site sobre os conteúdos necessários, principalmente sobre indução.

Exercício 1

Escreva um algoritmo recursivo que devolva o elemento de maior valor contido em um vetor de n elementos.

Exercício 2

Avalie $\sum_{k=0}^{40}(\log_2 4^k + 1)$.

Exercício 3

Prove que a quantidade máxima de nós em uma árvore binária de altura h é $2^{h+1} - 1$.

Exercício 4

Qual o erro?

Exercício 5

Denote por F_n o n-ésimo número da sequência de Fibonacci. Definimos $F_1 = 1$, $F_2 = 1$ e $F_n = F_{n-1} + F_{n-2}$ para $n \geq 3$. Prove que $F_n \geq 2^{0.5n}$ para todo $n \geq 6$.