

Construção e Análise de Algoritmos
Lista de exercícios 1

1. Fixe $k \geq 1$. Uma pessoa sobe uma escada de n degraus, com passos que podem alcançar entre 1 e $k \leq n$ degraus. Escreva equação de recorrência para calcular o número de modos distintos de subir essa escada.
2. Prove as seguintes afirmações sobre notação assintótica:
 - $n^3/100 - 25n^2 - 100n + 7$ é $\Omega(n^2)$ e $\Theta(n^3)$
 - $77n^3 - 13n^2 + 29n - 5$ é $O(n^4)$ e $\Omega(n^3)$
 - $34n \log_7 n^2 + 13n$ é $\Omega(n)$ e $O(n^2)$
3. Resolva as seguintes equações de recorrência segundo o método da árvore de recursão:
 - $T(n) = T(n-1) + 20$
 - $T(n) = 2 \cdot T(n-2) + 1$
 - $T(n) = 3 \cdot T(n/2) + n^2$
 - $T(n) = 4 \cdot T(n/2) + n^2$
 - $T(n) = 5 \cdot T(n/2) + n^2$
 - $T(n) = T(0.99 \cdot n) + 7$
 - $T(n) = T(\sqrt{n}) + 1$
 - $T(n) = T(\sqrt{n}) + \log_2 n$
 - $T(n) = 2 \cdot T(\sqrt{n}) + \log_2 n$
 - $T(n) = 2 \cdot T(n/5) + 3 \cdot T(n/6) + n$
4. Suponha que você está tentando escolher entre os três algoritmos abaixo. Qual o tempo de cada um em notação assintótica e qual você escolheria?
 - Algoritmo A resolve o problema dividindo a entrada em cinco subproblemas com a metade do tamanho, resolve cada subproblema recursivamente e depois combina-os em tempo linear.
 - Algoritmo B resolve o problema dividindo a entrada em dois subproblemas de tamanho $n-1$ (onde n é o tamanho da entrada), resolve cada subproblema recursivamente e depois combina-os em tempo constante.
 - Algoritmo C resolve o problema dividindo a entrada em nove subproblemas com um terço do tamanho, resolve cada subproblema recursivamente e depois combina-os em tempo quadrático.

5. Suponha que você tem k vetores ordenados de tamanho n e deseja combiná-los em um único vetor ordenado de tamanho kn .
- Uma ideia é usar o algoritmo INTERCALA, intercalando o primeiro e o segundo, depois intercalando o resultado com o terceiro, depois com o quarto e etc... Qual a complexidade desse procedimento em termos de k e n ?
 - Mostre uma solução mais eficiente usando divisão e conquista.
6. O algoritmo do k -ésimo mínimo ainda seria $\Theta(n)$ se tomássemos grupos de 3 elementos, ao invés de 5? E se tomássemos grupos de 7 elementos? Justifique pelo método da árvore de recursão.
7. Faça um algoritmo de tempo $\Theta(n \log n)$ para resolver o seguinte problema: dado um vetor com n números inteiros positivos e um outro número inteiro positivo x , determine se existem ou não dois elementos cuja soma é igual a x .
8. Elabore um algoritmo $O(n)$ de decomposição de um vetor S em três subvetores. Esse algoritmo recebe como entrada, além do vetor S , um valor *piv* pertencente a S , e os índices p e r , $1 \leq p \leq r$. O algoritmo deve rearrumar os elementos em $S[p \dots r]$ e retornar dois índices q_1 e q_2 satisfazendo as seguintes propriedades:
- se $p \leq k \leq q_1$, então $S[k] < \textit{piv}$;
 - se $q_1 < k \leq q_2$, então $S[k] = \textit{piv}$;
 - se $q_2 < k \leq r$, então $S[k] > \textit{piv}$.
9. Seja $X[1 \dots n]$ um vetor qualquer (os elementos desse vetor não são necessariamente inteiros ou caracteres; podem ser objetos quaisquer, como frutas ou arquivos executáveis). Suponha que você possui apenas um operador “=” que permite comparar se um objeto é igual a outro. Dizemos que X tem um elemento **majoritário** x se mais da metade de seus elementos são iguais a x . Escreva um algoritmo de tempo $\Theta(n \log n)$ que diz se X possui ou não um elemento majoritário. Caso sim, devolva o seu valor. **Dica:** Se x é majoritário em X , então x é majoritário na primeira ou na segunda metade de X (explique porquê).
10. Sejam $X[1 \dots n]$ e $Y[1 \dots n]$ dois vetores ordenados. Escreva um algoritmo $\Theta(\log n)$ para encontrar a mediana de todos os $2n$ elementos nos vetores X e Y . Prove esta complexidade.
11. Suponha que você possui dois vetores ordenados de tamanhos m e n . Você deseja obter o k -ésimo menor elemento da união desses dois vetores. Mostre um algoritmo de ordem $\Theta(\log m + \log n)$ que faça isso.
12. Seja $X[1 \dots n]$ um vetor de inteiros. Dados $i < j$ em $\{1, \dots, n\}$, dizemos que (i, j) é uma inversão de X se $X[i] > X[j]$. Escreva um algoritmo $\Theta(n \log n)$ que devolva o número de inversões em um vetor X .
13. Em sala de aula, vimos sem detalhes um algoritmo recursivo de divisão e conquista para multiplicar duas matrizes quadradas, dividindo cada matriz em 4 submatrizes quadradas e realizando 8 chamadas recursivas (não estamos falando do algoritmo de Strassen que realiza 7 chamadas recursivas, mas do mais simples visto antes dele). Descreva como seria um algoritmo que divida cada matriz em 9 submatrizes quadradas. Calcule a complexidade de tempo em notação assintótica.
14. Seja $X[1 \dots n]$ um vetor de **números reais**. Dizemos que X tem um elemento **popular** x se mais de **um terço** de seus elementos são iguais a x . Escreva um algoritmo de tempo linear $\Theta(n)$ que diz se X possui ou não um elemento popular. Caso sim, devolva o seu valor. **Dica:** Use o algoritmo de Seleção do k -ésimo mínimo de tempo linear no pior caso.