Lista de exercícios 1
Cana - 2025.2

Questão 1. Considere o algoritmo X abaixo que recebe como entrada um número natural n.

Algoritmo 1: X(n)

```
1 p ← 1
2 para i ← 1..n faça
3 |  p ← ip
4 Retorna p
```

(a) Simule a execução do Algoritmo X, descreva a sua funcionalidade (o que o algoritmo retorna?) e determine a sua complexidade.

(b) Escreva a versão recursiva do Algoritmo X e prove a sua corretude.

Questão 2. Considere o algoritmo abaixo que recebe um vetor $A[1..n]$ de números inteiros.

Algoritmo 2: Y(A, p, r)

```
1 m ← ⌊(p + r)/2⌋
2 se p = r então
3 |  Retorna p
4 u ← Y(A, p, m)
5 v ← Y(A, m + 1, r)
6 se A[u] < A[v] então
7 |  Retorna u
8 senão
9 |  Retorna v
```

(a) Simule a execução do Algoritmo Y no vetor de 6 posições correspondente ao seu número de matrícula. Neste caso, os valores iniciais de p e r são 1 e 6, respectivamente.

(b) Descreva sucintamente a funcionalidade do Algoritmo Y.

(c) Prove a corretude do Algoritmo Y.

Questão 3. Considere o algoritmo abaixo que recebe um vetor $A[1..n]$ de números inteiros.

(a) Simule a execução do Algoritmo X no vetor $< 15, 42, 21, 50, 33, 65, 40, 43, 20 >$, com os valores inicias de $i = 1$ e $j = 9$.

(b) Descreva sucintamente a funcionalidade do Algoritmo X.

(c) Prove a corretude de X.

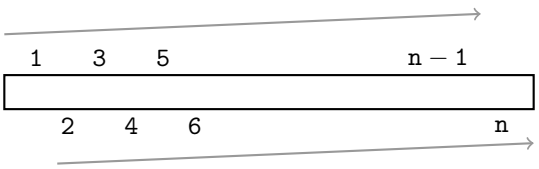
Algoritmo 3: X(A, i, j)

```
1 se A[i] > A[j] então
2   Troca A[i] ↔ A[j]
3 se i + 1 ≥ j então
4   Retorna
5 k ← ⌊(j − i + 1)/3⌋
6 X(A, i, j − k)
7 X(A, i + k, j)
8 X(A, i, j − k)
```

Questão 4. Escreva um algoritmo para realizar a busca linear recursiva em um vetor e mostre a sua corretude.

Questão 5. A *ordenação par-ímpar* consiste em

- ordenar os elementos das posições pares (em separado)
- ordenar os elementos das posições ímpares (em separado)



a) Apresente um algoritmo que realiza essa tarefa.

b) Analise o tempo de execução do seu algoritmo.

Questão 6. Elabore um algoritmo em $\mathcal{O}(n)$ 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 $S[p \ldots r]$ e retornar dois índices q_1 e q_2 satisfazendo as seguintes propriedades:

(a) se $p \leq k \leq q_1$, então $S[k] < piv$;

(b) se $q_1 < k \leq q_2$, então $S[k] = piv$;

(c) se $q_2 < k \leq r$, então $S[k] > piv$.

Questão 7. Considere uma sequência ordenada armazenada no vetor $A[1..n]$, e suponha que queremos encontrar a posição do elemento de valor x. O processo de busca binária resolve este problema de maneira eficiente. A sua análise mostra que o vetor original pode ser dividido em duas metades no máximo $\log_2 n$ vezes. Para cada divisão, é necessário realizar uma comparação para decidir em que metade o elemento x. Portanto, o processo requer $\log_2 n$ comparações.

Considere agora o processo de busca quaternária, que a cada passo divide a lista em quatro partes. Faça um algoritmo que implemente o procedimento de busca quaternária. Apresente uma estimativa do número de comparações requeridas para realizar este processo (no pior caso).

Questão 8. Projete um algoritmo em $\mathcal{O}(\log n)$ que recebe um vetor $A[1..n]$ de números em ordem não decrescente e um número x, e retorna a localização da primeira ocorrência de x em $A[1..n]$, ou o local em que x poderia ser inserido sem violar a ordenação se x não ocorrer no vetor.

Questão 9. É dado um vetor infinito $A[\cdot]$ no qual as primeiras n células contêm inteiros em ordem crescente e o restante das células é preenchido com ∞. Não é dado o valor de n. Descreva um algoritmo que toma um inteiro x como entrada e encontre uma posição no vetor contendo x, se tal posição existir, em tempo $O(log\ n)$.

Questão 10. Sejam $X[1 \ldots n]$ e $Y[1 \ldots n]$ dois vetores ordenados. Escreva um algoritmo em $\Theta(\log n)$ para encontrar a mediana de todos os $2n$ elementos dos vetores X e Y. Explique como seu algoritmo obtém tal complexidade.

Questão 11. Imagine que nós temos uma coleção de intervalos $I_j = [x_j, y_j]$.
E imagine que os intervalos estão armazenados como uma lista de pares

I	(x_1, y_1)	(x_2, y_2)	(x_3, y_3)	(x_4, y_4)	(x_5, y_5)	(x_6, y_6)

Nós dizemos que dois intervalos I_j e I_k estão invertidos se

- $j < k$ e $y_k < x_j$

— (*i.e., o intervalo que aparece antes na lista está à direita do outro, sem interseção*)

A tarefa consiste em

→ *Contar o número total de pares de intervalos invertidos*

Você consegue encontrar um algoritmo que resolve esse problema em tempo menor que $O(n^2)$?

Você consegue encontrar um algoritmo que resolve esse problema em tempo $O(n \log n)$?

Questão 12. Elabore um algoritmo em $\Theta(n \log n)$ para resolver o seguinte problema: dado um vetor com n números inteiros positivos e um outro número inteiro positivo x, determinar se existem ou não dois elementos cuja soma é igual a x.

Questão 13. Seja M uma matriz qualquer com n linhas e n colunas (os elementos dessa matriz 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 a matriz M tem um elemento super-majoritário x se mais de três quartos (3/4) de seus elementos são iguais a x. Escreva um algoritmo de tempo $\Theta(n^2\ log\ n)$ que diz se a matriz M possui ou não um elemento super-majoritário. Caso sim, devolva o seu valor. Explique a análise do tempo.

Questão 14. Você está prestando consultoria para uma pequena empresa de investimentos e eles têm o seguinte tipo de problema que eles querem resolver repetidamente. Um exemplo típico do problema é o seguinte. Eles estão fazendo uma simulação em que eles olham para n dias consecutivos de uma determinada ação, em algum momento do passado. Vamos numerar os dias $i = 1, 2, ..., n$; para cada dia i, eles têm um preço $p(i)$ por ação da ação naquele dia. (Vamos supor por simplicidade que o preço foi fixado durante cada dia.) Suponha que durante este período de tempo, eles queria comprar 1.000 ações

em algum dia e vender todas essas ações em algum dia (mais tarde). Eles querem saber: Quando eles deveriam ter comprado e quando eles deveriam ter vendido para ganhar o máximo de dinheiro possível? (Se não havia como ganhar dinheiro durante os n dias, você deve relatar isso). Por exemplo, suponha que $n = 3, p(1) = 9, p(2) = 1, p(3) = 5$. Então você deve retornar "compre no dia 2, venda no dia 3" (comprar no dia 2 e vender no dia 3 significa que eles teriam feito $R\$$ 4 por ação, o máximo possível para esse período). Claramente, existe um algoritmo simples que leva tempo $O(n^2)$: tente todos os possíveis pares de dias de compra/venda e veja qual lhes dá mais dinheiro. Seus amigos investidores esperavam algo um pouco melhor. Mostre como encontrar os números corretos i e j no tempo $O(n\ log\ n)$. (Dica: utilize divisão e conquista).

Questão 15. Elabore um algoritmo em $\Theta(n \log n)$ que, dado um vetor S com $n > 0$ elementos, retorna um vetor V de tamanho n com a seguinte propriedade: $V[i]$ é o número de ocorrências de $S[i]$ em S. Prove esta complexidade.