Universidade Federal do Pará
Instituto de Ciências Exatas e Naturais
Faculdade de Computação
Análise de Algoritmos

Lista de Exercícios

Questão 1. [POSCOMP 2012] Com base nos paradigmas de projeto de algoritmos, relacione a coluna da esquerda com a coluna da direita.

(I) Tentativa e Erro.	(A) Solução com garantia de distância da ótima.
(II) Divisão e Conquista.	(B) Subdivisão de problemas em partes menores, de tamanho semelhante.
(III) Balanceamento.	(C) Calcula a solução para os subproblemas, dos problemas menores para os maiores, armazenando os resultados parciais durante o processo, reutilizando-os assim que possível.
(IV) Algoritmos Aproximados.	(D) Geralmente exaurem-se todas as possibilidades para se encontrar uma solução. Todos os passos em direção à solução final são registrados. Se alguns dos passos não estiverem relacionados com a solução final, podem ser apagados.
(V) Programação Dinâmica.	(E) Divide problema em partes menores e combina sua solução em uma solução global.

Assinale a alternativa que contém a associação correta.

(A) I-A, II-D, III-B, IV-C, V-E.
(B) I-B, II-A, III-C, IV-E, V-D.
(C) I-B, II-A, III-E, IV-C, V-D.
(D) I-D, II-B, III-E, IV-A, V-C.
(E) I-D, II-E, III-B, IV-A, V-C.

Questão 2. [POSCOMP 2008] Analise as afirmativas abaixo.

I. A programação dinâmica é um método ascendente que aborda um dado problema subdividindo-o em problemas mínimos, soluciona esses subproblemas, guarda as soluções parciais, combina os subproblemas e sub-resultados para obter e resolver os problemas maiores, até recompor e resolver o problema original.

II. A divisão e conquista é um método recursivo e, por isso, descendente que decompõe sucessivamente um problema em subproblemas independentes triviais, resolvendo-os e combinando as soluções em uma solução para o problema original.

III. Um algoritmo guloso sempre faz escolhas que parecem ser as melhores no momento, ou seja, escolhas ótimas locais acreditando que estas escolhas o levem a uma solução ótima global. Por essa estratégia, nem sempre asseguram-se soluções ótimas, mas, para muitos problemas, as soluções são ótimas. Os problemas ideais para essa estratégia não devem ter a propriedade de subestrutura ótima.

A análise permite concluir que

(A) todas as afirmativas são verdadeiras.
(B) todas as afirmativas são falsas.
(C) apenas as afirmativas I e II são verdadeiras.
(D) apenas as afirmativas II e III são verdadeiras.
(E) apenas a afirmativa III é verdadeira.

Questão 3. Sobre o princípio conhecido como superposição de subproblemas, é correto afirmar que

(A) a programação dinâmica não segue esse princípio.
(B) esse princípio ocorre quando um algoritmo recursivo reexamina o mesmo subproblema muitas vezes.
(C) é o princípio fundamental para a aplicação de algoritmos força-bruta.
(D) esse princípio diz que, em uma sequência ótima de escolhas ou decisões, cada subsequência também deve ser ótima.
(E) de acordo com esse princípio, as escolhas feitas a cada iteração do algoritmo são definitivas, ou seja, a escolha não pode ser alterada nos passos subsequentes do algoritmo.

Questão 4. Analise as afirmativas abaixo.

I. *Branch-and-Bound* baseia-se na ideia de desenvolver uma enumeração inteligente das soluções candidatas à solução ótima de um problema, o que possibilita abandonar uma candidata parcialmente construída tão logo quanto for possível determinar que ela não pode gerar a solução ótima.

II. *Backtracking* incrementalmente constrói candidatas de soluções e abandona uma candidata parcialmente construída tão logo quanto for possível determinar que ela não pode gerar uma solução válida.

III. *Branch-and-Bound* só pode ser aplicado em problemas de otimização do tipo minimização, ou seja, esse tipo de algoritmo não resolve problemas de maximização.

A análise permite concluir que

(A) apenas a afirmativa II é verdadeira.
(B) apenas as afirmativas I e II são verdadeiras.
(C) apenas as afirmativas I e III são verdadeiras.
(D) apenas as afirmativas II e III são verdadeiras.
(E) todas as afirmativas são falsas.

Questão 5. Analise as afirmativas abaixo.

I. Os algoritmos força-bruta de enumeração total não usam recursividade, já que são aplicados exaustivamente.

II. Na técnica de divisão e conquista, procura-se dividir o problema em subproblemas balanceados. O efeito provocado pelo balanceamento é percebido no desempenho final do algoritmo.

III. No contexto da programação dinâmica, um problema apresenta uma subestrutura ótima quando sua solução ótima contém nela própria soluções ótimas para subproblemas semelhantes de complexidade assintótica linear.

A análise permite concluir que

(A) todas as afirmativas são falsas.
(B) apenas a afirmativa II é verdadeira.
(C) apenas as afirmativas I e II são verdadeiras.
(D) apenas as afirmativas I e III são verdadeiras.
(E) apenas as afirmativas II e III são verdadeiras.

Questão 6. [POSCOMP 2016] Em relação ao projeto de algoritmos, relacione a Coluna 1 à Coluna 2.

Coluna 1

1. Tentativa e Erro.
2. Divisão e Conquista.
3. Guloso.
4. Aproximado.
5. Heurística.

Coluna 2

( ) O algoritmo decompõe o processo em um número finito de subtarefas parciais que devem ser exploradas exaustivamente.

( ) O algoritmo divide o problema a ser resolvido em partes menores, encontra soluções para as partes e então combina as soluções obtidas em uma solução global.

( ) O algoritmo constrói por etapas uma solução ótima. Em cada passo, após selecionar um elemento da entrada (o melhor), decide se ele é viável (caso em que virá a fazer parte da solução) ou não. Após uma sequência de decisões, uma solução para o problema é alcançada.

( ) O algoritmo gera soluções cujo resultado encontra-se dentro de um limite para a razão entre a solução ótima e a produzida pelo algoritmo.

( ) O algoritmo pode produzir um bom resultado, ou até mesmo obter uma solução ótima, mas pode também não produzir solução nenhuma ou uma solução distante da solução ótima.

A ordem correta de preenchimento dos parênteses, de cima para baixo, é:

(A) 1 - 2 - 3 - 4 - 5.
(B) 2 - 3 - 4 - 5 - 1.
(C) 3 - 4 - 5 - 1 - 2.
(D) 4 - 5 - 1 - 2 - 3.
(E) 5 - 1 - 2 - 3 - 4.

Questão 7. [POSCOMP 2007] Considere o problema do caixeiro viajante, definido como se segue.

Sejam S um conjunto de n cidades, $n \geq 0$, e $d_{ij} > 0$ a distância entre as cidades i e j, $i, j \in S$, $i \neq j$. Define-se um percurso fechado como sendo um percurso que parte de uma cidade $i \in S$, passa exatamente uma vez por cada cidade de $S - i$, e retorna à cidade de origem. A distância de um percurso fechado é definida como sendo a soma das distâncias entre cidades consecutivas no percurso. Deseja-se encontrar um percurso fechado de distância mínima. Suponha um algoritmo guloso que, partindo da cidade 1, move-se para a cidade mais próxima ainda não visitada e que repita esse processo até passar por todas as cidades, retornando à cidade 1.

Considere as seguintes afirmativas.

I. Todo percurso fechado obtido com esse algoritmo tem distância mínima.

II. O problema do caixeiro viajante pode ser resolvido com um algoritmo de complexidade linear no número de cidades.

III. Dado que todo percurso fechado corresponde a uma permutação das cidades, existe um algoritmo de complexidade fatorial no número de cidades para o problema do caixeiro viajante.

Com relação a essas afirmativas, pode-se afirmar que

(A) I é falsa e III é correta.
(B) I, II e III são corretas.
(C) apenas I e II são corretas.
(D) apenas I e III são falsas.
(E) I, II e III são falsas.

Questão 8. [POSCOMP 2022] Os algoritmos de ordenação MergeSort, da árvore geradora mínima de Kruskal, e o algoritmo Floyd-Warshall que calcula o caminho mais curto entre todos os pares de vértices de um grafo orientado com peso são, respectivamente, exemplos de algoritmos:

(A) Guloso, programação dinâmica e divisão e conquista.
(B) Divisão e conquista, programação dinâmica e guloso.
(C) Guloso, divisão e conquista e programação dinâmica.
(D) Programação dinâmica, divisão e conquista e guloso.
(E) Divisão e conquista, guloso e programação dinâmica.

Questão 9. Analise as afirmativas abaixo.

I. Problemas aparentemente intratáveis ou difíceis são muito comuns na teoria da computação, por exemplo, o problema do caixeiro viajante, cuja complexidade de tempo é $O(n!)$.

II. O problema do caixeiro viajante pode ser resolvido da seguinte forma: (i) passos em direção à solução final são tentados e registrados; (ii) caso esses passos tomados não levem à solução final, eles podem ser retirados e apagados do registro.

III. Existe um algoritmo *branch-and-bound* aplicado ao problema do caixeiro viajante que possui ordem de complexidade $O(n^2)$, em que n corresponde ao número de vértices do grafo.

A análise permite concluir que

(A) I é falsa e III é correta.
(B) I, II e III são corretas.
(C) apenas I e II são corretas.
(D) apenas I e III são falsas.
(E) I, II e III são falsas.

Questão 10. Analise o algoritmo abaixo desenvolvido para obter um caminho solução para o problema do caixeiro-viajante.

Passo 1. Inicie com um vértice arbitrário.

Passo 2. Procure o vértice mais próximo do último vértice adicionado que não esteja no caminho e adicione ao caminho a aresta que liga esses dois vértices.

Passo 3. Quando todos os vértices estiverem no caminho, adicione uma aresta conectando o vértice inicial e o último vértice adicionado.

A análise permite concluir que esse algoritmo

(A) adota uma heurística gulosa.
(B) não pode ser implementado em tempo polinomial.
(C) é ótimo.
(D) é força bruta.
(E) é necessariamente recursivo.

[POSCOMP 2014] Considere o pseudocódigo abaixo para responder as **Questões 11 e 12**.

```
HUFFMAN (C)
1. n = |C|
2. Q = C
3. para i = 1 até n - 1
4.    Aloca um novo nó z
5.    z.esquerda = x = Extract-min(Q)
6.    z.direita  = y = Extract-min(Q)
7.    z.freq = x.freq + y.freq
8.    Insert(Q, z)
9. retorna Extract-min(Q)  // raiz da árvore
```

Questão 11. Sobre o pseudocódigo, é correto afirmar que é um algoritmo

(A) aproximado.
(B) divisão-e-conquista.
(C) guloso.
(D) recursivo.
(E) tentativa e erro.

Questão 12. Sobre o comportamento assintótico desse pseudocódigo, é correto afirmar que sua complexidade é

(A) $O(n^2)$.
(B) $O(n^3)$.
(C) $O(2^n)$.
(D) $O(2n)$.
(E) $O(n\ lg\ n)$.

Questão 13. Quantos *bits* seriam necessários para representar a cadeia de caracteres MISSISSIPI usando um código de Huffman?

(A) 18.
(B) 19.
(C) 20.
(D) 21.
(E) As informações fornecidas no enunciado são insuficientes para realizar a codificação de Huffman.

Questão 14. [POSCOMP 2007] Um sistema de codificação e compressão de imagens consiste de dois blocos, que são: o codificador e o decodificador. Entre as diversas técnicas de codificação, a mais popular é o código de Huffman. Considere a tabela abaixo, em que é apresentado o código resultante de um processo de codificação.

probabilidade	**código**
0,35	1
0,25	01
0,20	010
0,10	0101
0,05	01011
0,03	010110
0,01	0101100
0,01	0101101

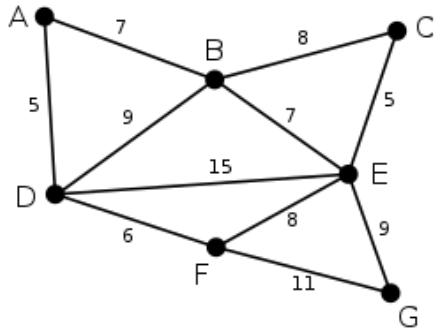
Nesse caso, o comprimento médio do código obtido foi de

(A) 3,15 bits/símbolo.
(B) 1,14 bits/símbolo.
(C) 2,42 bits/símbolo.
(D) 4,38 bits/símbolo.
(E) 3,00 bits/símbolo.

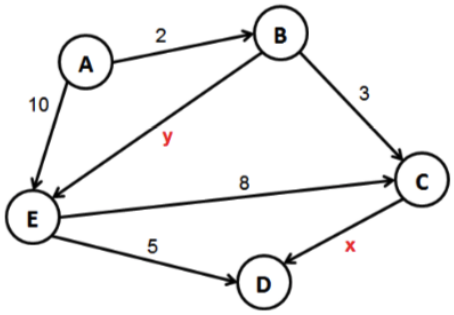
Questão 15. Assinale a alternativa que mostra um código de Huffman para o conjunto de frequências: r:1, s:2, t:5, u:13, v:34.

(A) {r,s,t,u,v} = {111100, 1110, 110, 10, 0}.
(B) {r,s,t,u,v} = {0, 11, 110, 1111, 1110}.
(C) {r,s,t,u,v} = {1111, 1110, 110, 10, 0}.
(D) {r,s,t,u,v} = {1110, 1111, 110, 11, 0}.
(E) {r,s,t,u,v} = {111, 110, 101, 100, 000}.

Questão 16. Encontre uma árvore gerada mínima para o grafo abaixo. Descreva o passo-a-passo do algoritmo usado para resolver o problema.



Questão 17. Dado o grafo G abaixo, onde n é o número de vértices e m o número de arestas, marque a afirmativa **INCORRETA**.



G

(A) Se $x = 4$ e $y = 7$, então o caminho mínimo, considerando o peso das arestas, do vértice A ao vértice D será: A, B, C, D.

(B) Suponha que o algoritmo de Dijkstra seja executado no grafo G, então é possível atingir um tempo de execução da ordem $m + n^2$ se usarmos um vetor não-ordenado como fila de prioridade.

(C) Suponha que o peso da aresta (A, E) seja reduzido para 1, com $y \geq 0$ e $x \geq 7$, então o caminho mínimo, considerando o peso das arestas, do vértice A ao vértice D será: A, E, D.

(D) Se $x \geq y + 3$, então o caminho mínimo, considerando o peso das arestas, do vértice B ao vértice D será: B, E, D.

(E) Se $x \geq 0$, $y \geq 0$ e $x \geq y$, então o caminho mínimo, considerando o peso das arestas, do vértice A ao vértice D será: A, B, E, D.

Questão 18. Considere um tabuleiro abaixo com 3 x 3 quadrículas. Cada quadrícula contém um número.

0	4	3
7	8	6
2	3	1

O objetivo do jogo consiste em deslocar um peão desde o canto superior esquerdo até o canto inferior direito, através de uma sequência de movimentos para a direita ou para baixo, de forma a minimizar o somatório dos pontos correspondentes às quadrículas por onde se passou. Formule o jogo como um problema de caminho mínimo e resolva-o usando o algoritmo de Dijkstra.