

Gabarito da Segunda Avaliação à Distância

1. (1,0) Prove ou dê um contra-exemplo: Uma árvore binária pode ser construída, de forma única, a partir dos seus percursos em *pré-ordem* e *em nível*.

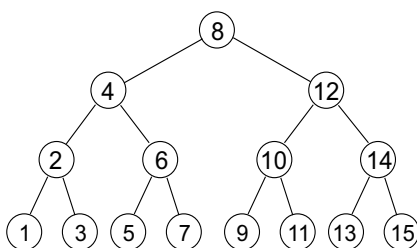
Resposta: A afirmação é falsa. Considere duas árvores binárias T_1 e T_2 , onde cada uma delas contém apenas dois nós A e B de forma que:

- em T_1 , B é filho esquerdo de A ;
- em T_2 , B é filho direito de A .

Para ambas as árvores, os percursos em pré-ordem e em nível são AB . No entanto, elas são distintas.

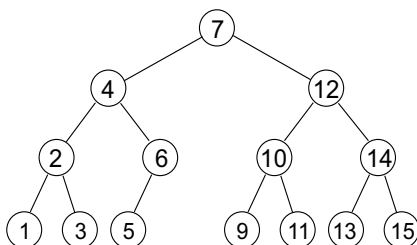
2. (1,0) Desenhe uma árvore binária de busca *cheia* com altura 4, colocando dentro de cada nó o valor de sua chave. As chaves são $1, 2, \dots, k$ (k é o número de nós da árvore, que é um valor que você deve deduzir). A seguir, escreva a sequência de chaves que corresponde ao percurso em *pós-ordem* desta árvore.

Resposta: O percurso em pós-ordem desta árvore é: 1 3 2 5 7 6 4 9 11 10 13 15 14 12 8.

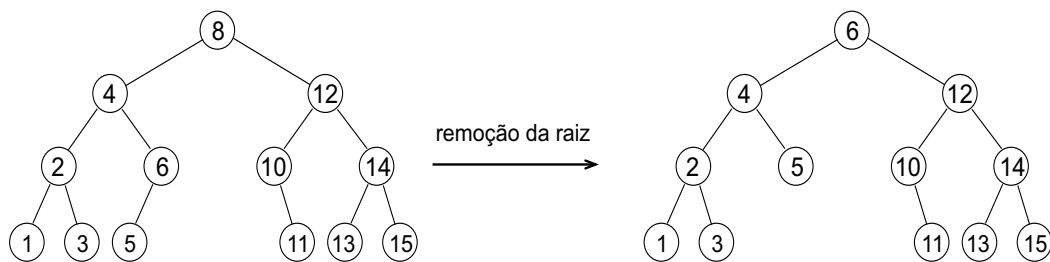


3. (1,5) Suponha que você deseja remover a raiz de uma árvore binária de busca. Após removê-la, como você deve reestruturar a árvore de modo que ela continue sendo uma árvore binária de busca? Dê um exemplo que mostre seu raciocínio.

Resposta: Basta selecionar o nó mais à direita da subárvore esquerda (que é o nó de maior valor desta subárvore), ou o nó mais à esquerda da subárvore direita (o de menor valor) e substituir a raiz por este nó. Utilizando o exemplo da árvore cheia da questão anterior, poderíamos substituir a raiz pela chave 7 (ou pela chave 9).



Caso a árvore binária de busca não seja cheia, e tanto o nó de maior valor da subárvore esquerda quanto o nó de menor valor da subárvore direita não sejam folhas, então basta selecionar um destes nós para substituir a raiz, e “promover” a subárvore deste nó para sua antiga posição, como no exemplo da figura a seguir.



4. (1,5) Construa a árvore binária de busca ótima para o seguinte conjunto de frequências:

j	f_j	f'_j
0	-	2
1	1	0
2	2	1
3	1	0

Resposta: As matrizes do algoritmo de cálculo da árvore ótima são:

Matriz dos custos $c[i, j]$:

0	3	9	12
-	0	3	6
-	-	0	2
-	-	-	0

Matriz dos valores $F[i, j]$:

2	3	6	7
-	0	3	4
-	-	1	2
-	-	-	0

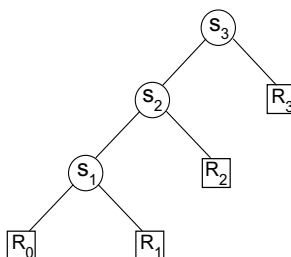
Matriz dos valores minimizantes k :

-	1	1(2)	2
-	-	2	2
-	-	-	3
-	-	-	-

Da última matriz acima, segue que a árvore binária de custo ótimo tem raiz s_2 , e portanto filho esquerdo s_1 e filho direito s_3 .

5. (1,0) Seja T uma árvore binária de custo mínimo relativa às frequências $f_1, f_2, f_3, f'_0, f'_1, f'_2, f'_3$. Escreva valores para estas frequências de modo que T seja uma árvore zigue-zague.


Resposta: Para as frequências $f_1 = 1, f_2 = 2, f_3 = 4, f'_0 = f'_1 = f'_2 = f'_3 = 0$ temos a seguinte árvore binária de custo mínimo.



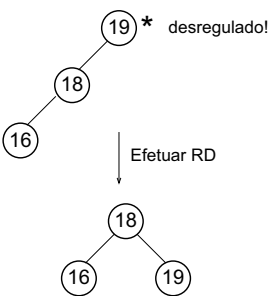
6. (1,0) Desenhe a árvore AVL obtida pela sequência de inserções das chaves 19, 18, 16, 15, 17, 2, 6, nessa ordem. Explique as operações realizadas, passo a passo.

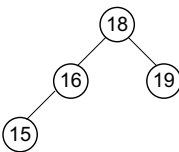
Resposta:

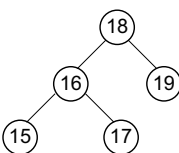
Início: árvore vazia

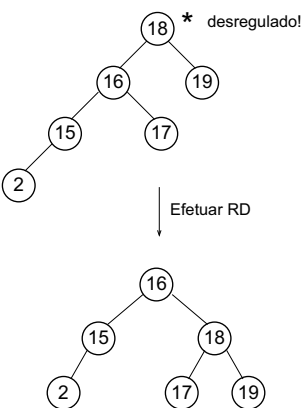
Inserir 19: 

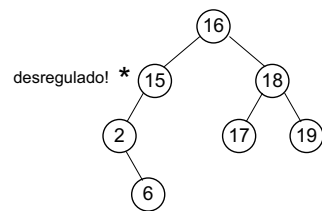
Inserir 18: 

Inserir 16: 

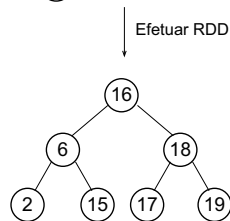
Inserir 15: 

Inserir 17: 

Inserir 2: 

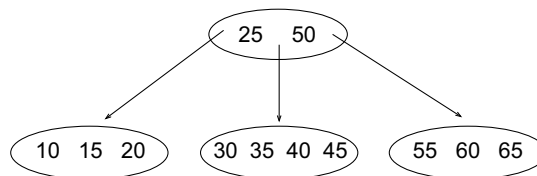


Inserir 6:



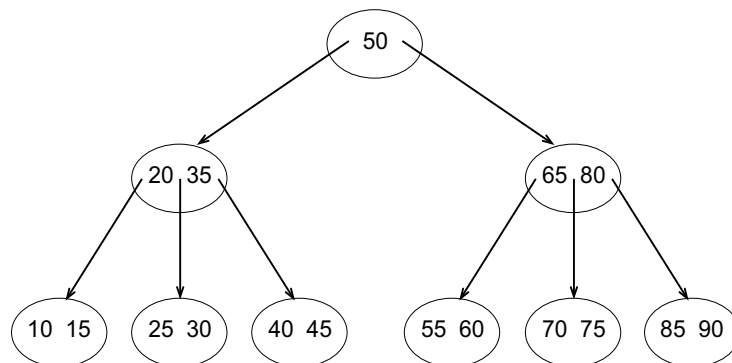
7. (1,0) Desenhe uma árvore B de ordem 3 que contenha as seguintes chaves: 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65.

Resposta:



8. (1,0) Desenhe uma árvore B de ordem $d = 2$ com três níveis e o menor número possível de chaves. (Os valores das chaves ficam à sua escolha.)

Resposta:



9. (CANCELADA) Construa um heap com as seguintes prioridades: 18, 25, 41, 34, 14, 10, 52, 50, 48. A seguir, redesenhe o heap após a remoção do nó com prioridade 34.

10. (1,0) Determine o heap obtido pela aplicação do algoritmo de construção à seguinte lista de prioridades: 18, 25, 41, 34, 14, 10, 52, 50, 48. Explique passo a passo.

Resposta: Os passos do algoritmo de complexidade $O(n)$ são os seguintes:

Início: 18, 25, 41, 34, 14, 10, 52, 50, 48

Descer 34: 18, 25, 41, 50, 14, 10, 52, 34, 48

Descer 41: 18, 25, 52, 50, 14, 10, 41, 34, 48

Descer 25: 18, 50, 52, 48, 14, 10, 41, 34, 25

Descer 18: 52, 50, 41, 48, 14, 10, 18, 34, 25 \rightarrow heap final!