Estrutura de Dados - 1o. período de 2007

Gabarito da Segunda Avaliação à Distância

A Questão 1 vale 1,0 ponto, e as demais valem 1,5 ponto cada.

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

   Resposta: A afirmação é verdadeira. Suponha que o percurso em ordem simétrica é dado pela sequência de nós $x_1 x_2 \ldots x_n$, e que o percurso em ordem em nível é dado pela sequência de nós $y_1 y_2 \ldots y_n$. Sabemos que a raiz da árvore é o nó y_1. Seja $y_1 = x_j$, $1 \leq j \leq n$, no percurso em ordem simétrica. Logo, os nós $x_1 \ldots x_{j-1}$ pertencem à sub-árvore esquerda T_E da raiz, e os nós $x_{j+1} \ldots x_n$ pertencem à sub-árvore T_D direita da raiz. Aplicando recursivamente este raciocínio a T_E, basta verificarmos qual dos nós $x_1 \ldots x_{j-1}$ aparece primeiro no percurso em nível (ou seja, qual deles corresponde a um y_i, tal que i é o menor possível), e descobrimos a raiz de T_E. Similarmente para T_D. Assim, conseguimos reconstruir a árvore binária original de forma única.

2. Apresentar um algoritmo que imprima os nós de uma árvore binária em ordem não decrescente de suas alturas. Isto é, primeiro imprime os nós de altura 1 (em qualquer ordem entre si), depois os de altura 2 (em qualquer ordem entre si), e assim por diante, até imprimir por último a raiz. Utilize a representação de árvores binárias por ponteiros descrita no livro.

   Resposta: Seja h a altura da árvore dada. O algoritmo a seguir percorre a árvore $h-1$ vezes, imprimindo e removendo suas folhas (enquanto a raiz não for uma folha). No procedimento busca-folhas, é utilizada uma variável chamada *lado*, configurada com 0 quando o nó é filho esquerdo de seu pai, ou com 1, quando é filho direito.

   enquanto ($ptraiz \uparrow .esq \neq \lambda$) ou ($ptraiz \uparrow .dir \neq \lambda$) faça
       se ($ptraiz \uparrow .esq \neq \lambda$) então
           busca-folhas($ptraiz \uparrow .esq, ptraiz, 0$)
       se ($ptraiz \uparrow .dir \neq \lambda$) então
           busca-folhas($ptraiz \uparrow .dir, ptraiz, 1$)
   $imprimir(ptraiz \uparrow .info)$

procedimento busca-folhas$(no, pai, lado)$

  se $(no \uparrow .esq = \lambda)$ e $(no \uparrow .dir = \lambda)$ então

    se $(lado = 0)$ então $pai \uparrow .esq = \lambda$

          senão $pai \uparrow .dir = \lambda$

    $imprimir(no)$

    $desocupar(no)$

  senão

    se $(no \uparrow .esq \neq \lambda)$ então busca-folhas$(no \uparrow .esq, no, 0)$

    se $(no \uparrow .dir \neq \lambda)$ então busca-folhas$(no \uparrow .dir, no, 1)$

3. Determinar a árvore binária de custo mínimo relativa às seguintes frequências: $f_1 = 1$, $f_2 = 1$, $f_3 = 1$, $f'_0 = 0$, $f'_1 = 0$, $f'_2 = 0$, $f'_3 = 0$. Tente generalizar, caracterizando a árvore binária de busca ótima quando todas as frequências f_i são unitárias e todas as frequências f'_i são iguais a zero.

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

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

0	1	3	5
-	0	1	3
-	-	0	1
-	-	-	0

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

0	1	2	3
-	0	1	2
-	-	0	1
-	-	-	0

   Matriz dos valores minimizantes k:

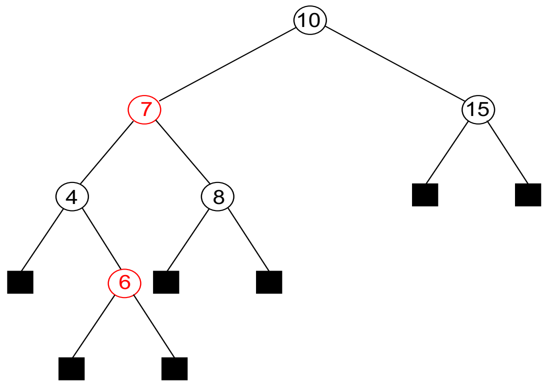
-	1	1 (2)	2
-	-	2	2 (3)
-	-	-	3
-	-	-	-

   Da última matriz acima obtemos a árvore ótima de raiz s_2, filho esquerdo s_1 e direito s_3.

   Geralizando, temos que a árvore binária de busca ótima com n nós, tal que todas as frequências f_i são unitárias e as f'_i são iguais a zero, é uma árvore cheia (caso $n = 2^h - 1, h \geq 1$) ou é uma árvore completa (cheia até o penúltimo nível).

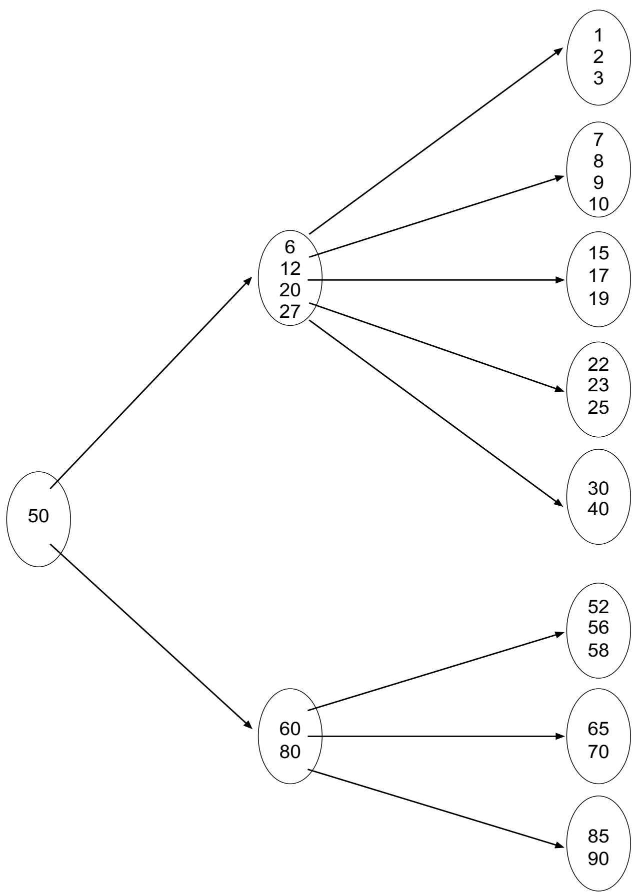
4. A partir de uma árvore inicialmente vazia, desenhe a árvore rubro-negra resultante da inserção de nós com chaves $10, 4, 15, 8, 7, 6$ (nesta ordem). Não esqueça de representar os nós externos, nem de representar as cores dos nós.

   Resposta:

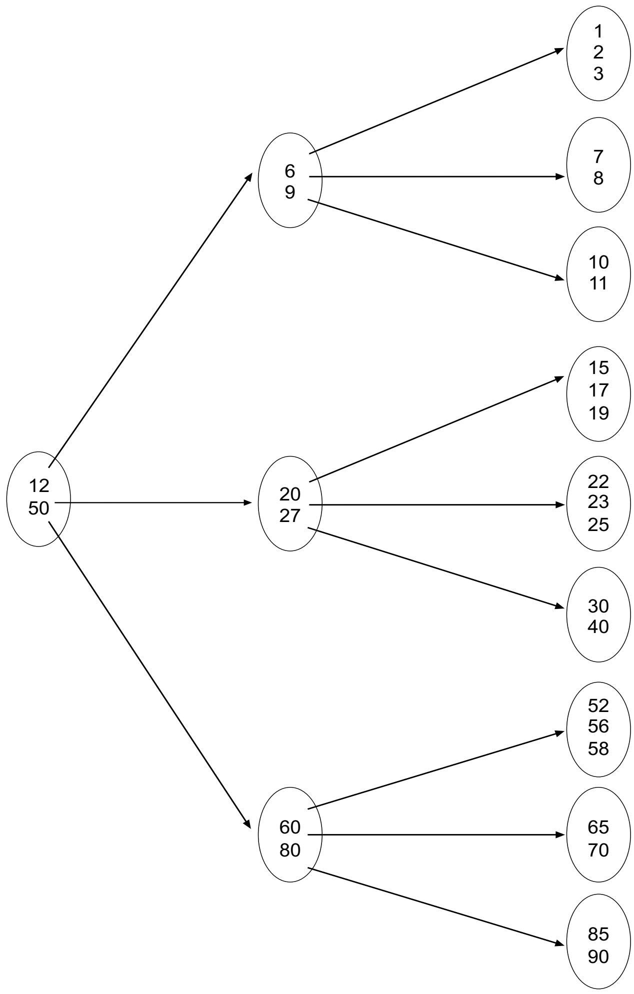


5. Desenhe uma árvore B de ordem $d = 2$ com três níveis. (Os valores nos nós ficam à sua escolha.) A seguir, escolha uma nova chave de forma que a sua *inserção* exija uma cisão propagável. Desenhe a árvore B resultante após a inserção.

   Resposta:



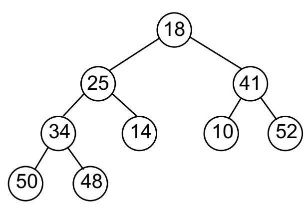
Inserindo a chave 11, temos uma cisão propagável, que resulta na seguinte árvore B:



6. Execute o método de ordenação por heap ("heapsort"), aplicando-o às seguintes prioridades (nesta ordem): $18, 25, 41, 34, 14, 10, 52, 50, 48$. Desenhe as configurações sucessivas da árvore durante o processo de ordenação.

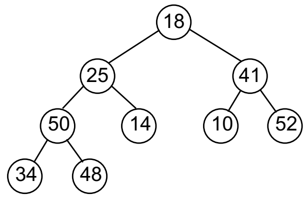
   Resposta:

   Início:

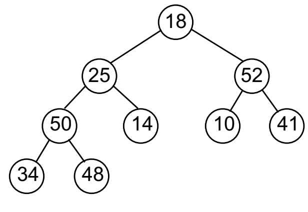


   <u>Construção do heap:</u> (comando *arranjar*(n))

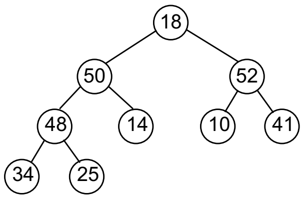
   Descer 34:



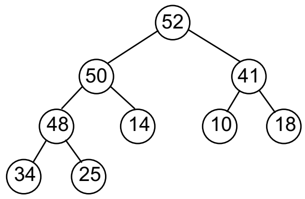
   Descer 41:



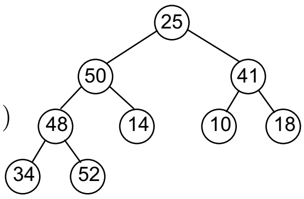
Descer 25:



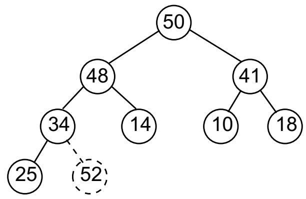
Descer 18: Heap obtido:



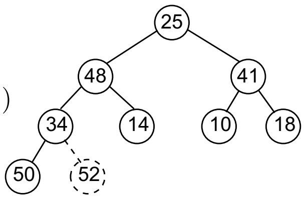
$m := 9$
trocar(TB[1],TB[9])



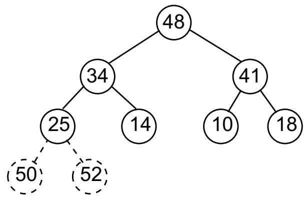
$m := 8$
descer(1,8)



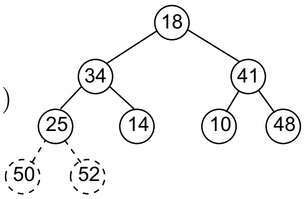
trocar(TB[1],TB[8])



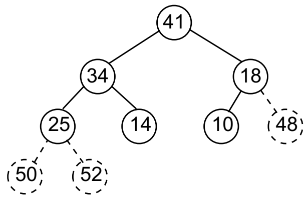
$m := 7$
descer(1,7)



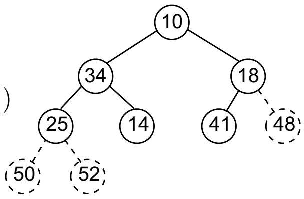
trocar(TB[1],TB[7])



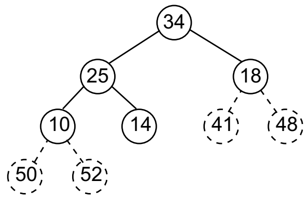
$m := 6$
descer(1,6)



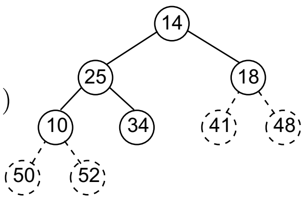
trocar(TB[1],TB[6])



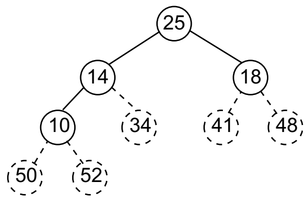
$m := 5$
descer(1,5)



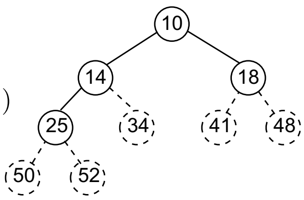
trocar(TB[1],TB[5])



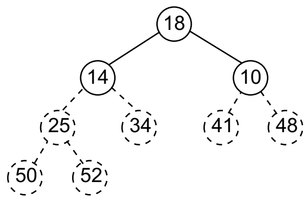
$m := 4$
descer(1,4)



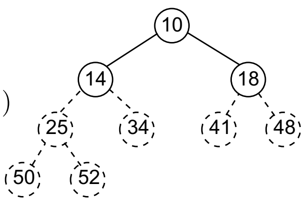
trocar(TB[1],TB[4])



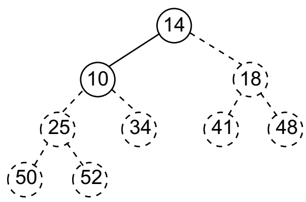
$m := 3$
descer(1,3)



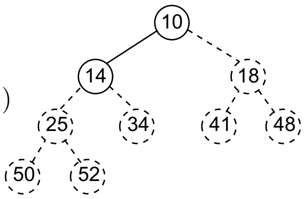
trocar(TB[1],TB[3])



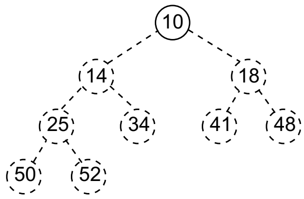
$m := 2$
descer(1,2)



trocar(TB[1],TB[2])



$m := 1$
descer(1,1)



7. Responda: como é a árvore de Huffman relativa a n frequências iguais? (Suponha que n é da forma $n = 2^k$, isto é, n é uma potência de 2.)

   Resposta: É uma árvore cheia de altura $k + 1$ (supondo $n = 2^k$).