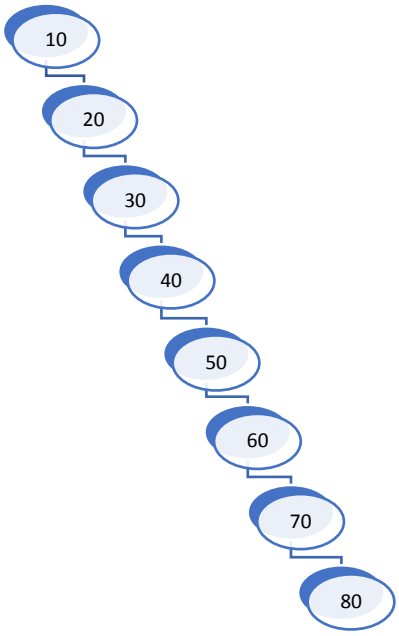
Curso de Tecnologia em Sistemas de Computação
Disciplina: Estrutura de Dados
Gabarito AP2 - 2018/2

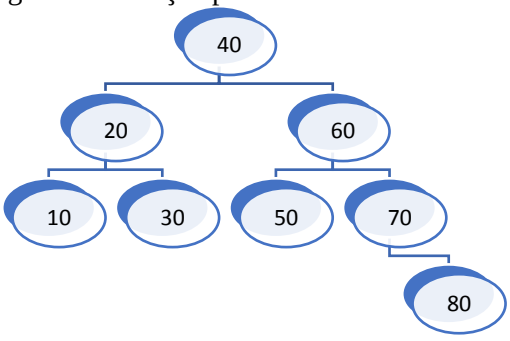
1) (a) (1,0) Desenhe uma árvore binária de busca com 8 elementos e altura máxima possível contendo as chaves 10, 20, 30, 40, 50, 60, 70, 80.

R: Segue uma solução possível:

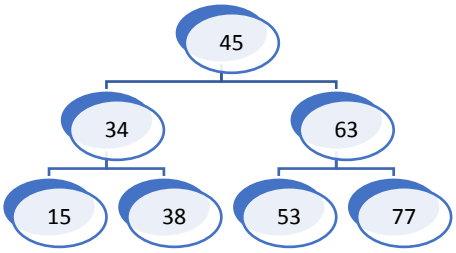


(b) (1,0) Desenhe uma árvore binária de busca com 8 elementos e altura mínima possível contendo as chaves 10, 20, 30, 40, 50, 60 70, 80.

R: Segue uma solução possível:

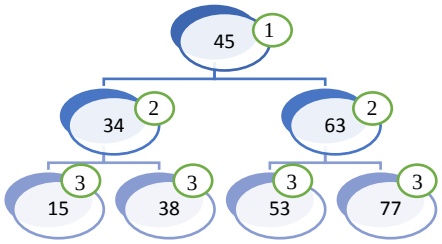


2) **Considere a árvore binária de busca T formada pela inserção dos nós com as seguintes chaves (nesta ordem): 45, 34, 15, 38, 63, 53, 77.**
R: A árvore formada seria:



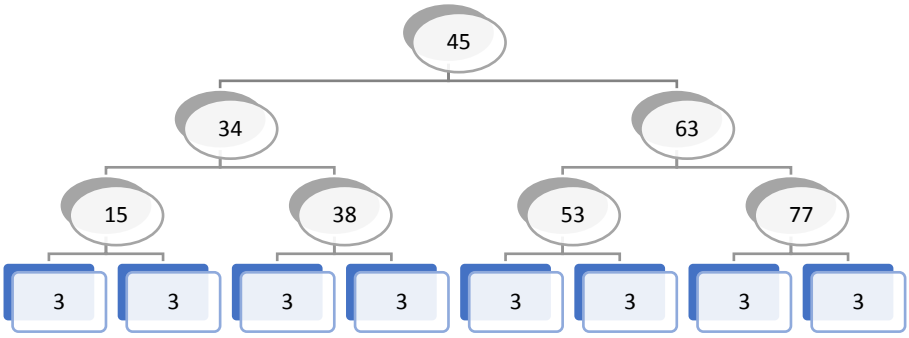
(a) (1,0) Calcule: o comprimento do caminho interno de T, o comprimento do caminho externo de T, e o custo final da árvore.
R: Considerando a árvore em questão, os comprimentos seriam dados por:

- Caminho Interno: 1 + 2 + 2 + 3 + 3 + 3 + 3 = 17



**Os números acima de cada nó representam os elementos considerados na soma do comprimento do caminho interno.*

- Caminho Externo: 3 + 3 + 3 + 3 + 3 + 3 + 3 + 3 = 24



**Os números dentro das folhas vazias (ausência de chaves) mostram os elementos considerados na soma do comprimento do caminho externo.*

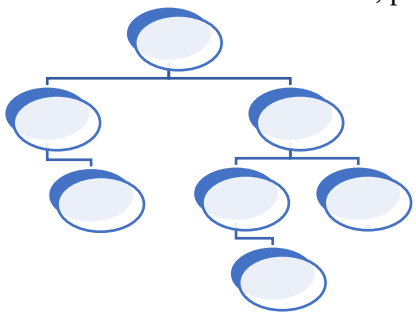
- A árvore binária de busca formada é completa. Por essa razão o custo das operações de inserção, busca e remoção nesta árvore é O (log N), onde N é o número de chaves.

(b) (1,0) Responda: é possível construir outra árvore binária T' com as mesmas chaves de T mas com custo menor que o de T? Justifique.
R: Não seria possível pois a árvore em questão já possui o menor custo possível. Isso acontece porque ela é completa (lembre-se, toda árvore cheia é completa, mas nem toda árvore completa é uma árvore cheia). Essas árvores minimizam o número de comparações efetuadas no pior caso.

3) Responda V ou F, justificando:

(a) (1,0) Toda árvore AVL é também uma árvore binária de busca completa.

R: A afirmação é falsa. Nem toda árvore AVL pode ser considerada uma árvore completa. Uma árvore AVL é uma árvore onde o custo das operações de busca, inserção, remoção mantém-se em O(log n). Isso não necessariamente implica em a árvore ser sempre completa. A estrutura abaixo representa uma AVL nesta situação. Note que todos os nós estão balanceados, porém a árvore não é completa.



(b) (1,0) Se uma árvore B de ordem d e altura h possui 2d chaves em todas as suas páginas então o número de páginas desta árvore é o máximo possível.

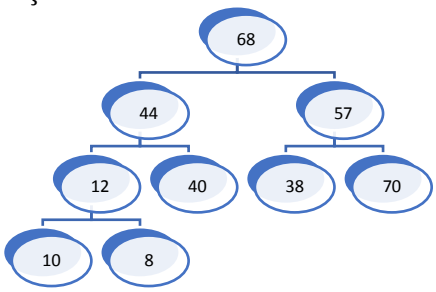
R: Para que uma árvore B tenha a maior quantidade de páginas possíveis, cada página deve possuir 2d+1 filhos. Para que uma página tenha esta quantidade de filhos, é necessário que esta página tenha a quantidade máxima de chaves permitida, que é 2d chaves. Portanto, a afirmação deste exercício é verdadeira. Logo, assumindo que todas as páginas tenham esta quantidade de chaves, teremos então árvore B com o maior número de páginas possível.

4) Seja V um vetor contendo, nesta ordem, os elementos: 68, 44, 57, 12, 40, 38, 70, 10, 8.

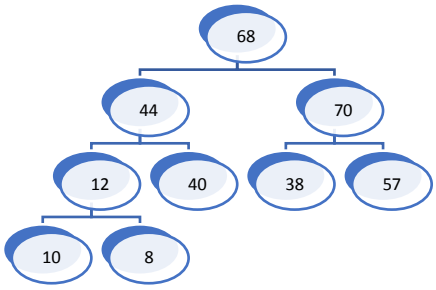
(a) **(1,0) Quantas trocas de elementos são necessárias para que V se converta em um heap, usando o algoritmo de tempo linear de construção de heaps? Justifique.**

R: Considerando a construção de um *heap*, teríamos duas trocas, conforme ilustrado abaixo.

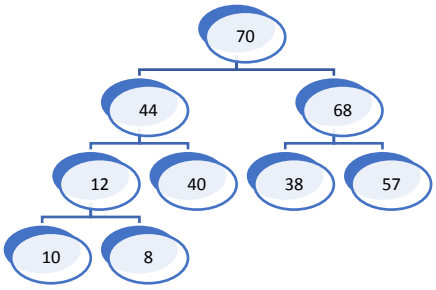
Condição inicial



1) Desce 57:



2) Desce 68:



(b) **(1,0) Se o vetor V tiver exatamente 7 elementos, quantas trocas de elementos são realizadas no pior caso para que V se converta em um heap, utilizando o mesmo algoritmo de construção do item anterior?**

R: São necessárias no máximo 4 trocas pois, no pior caso, cada nó interno deve descer até o último nível.

5) Considere uma tabela de dispersão T com m endereços-base numerados de 0 a m-1, utilizando a função de dispersão h(x) = x mod m. Desejamos inserir em T as chaves correspondentes aos 10 primeiros múltiplos positivos de 5, do menor para o maior. Quantas colisões ocorrerão neste processo de inserção, nos seguintes casos:

(a) (1,0) m = 5.

R: Observe que utilizando esta função de dispersão em uma tabela com 5 endereços bases, todos as chaves a seguir seriam inseridas no mesmo espaço, portanto, ocorreriam M-1 colisões, ou seja, 9 colisões.

Chaves = x	5	10	15	20	25	30	35	40	45	50
h(x) = x *mod* m	0	0	0	0	0	0	0	0	0	0

(b) (1,0) m = 7.

R: Por outro lado, considerando a mesma função de dispersão em uma tabela com 7 endereços, isto é, m=7, teríamos apenas 3 colisões, referentes aos elementos 40, 45 e 50, que ocupariam, respectivamente as posições 5, 3 e 1 da tabela. Observe.

Chaves = x	5	10	15	20	25	30	35	40	45	50
h(x) = x *mod* m	5	3	1	6	4	2	0	5	3	1

Endereço	Chave
0	35
1	15, 50
2	30
3	10, 45
4	25
5	5, 40
6	20

** As chaves em vermelho indicam colisões. Observe que o enunciado não relatou nenhum método de encadeamento como alternativa às colisões, por esta razão, sempre que uma colisão acontece, a chave antiga é removida da tabela e substituída pela nova.