

Lista de Exercícios 1

Versão: 1.5

Prof. Leônidas de Oliveira Brandão
Departamento de Ciência da Computação - IME - USP
<http://www.ime.usp.br/~leo>

1. HUFFMAN

Exercício 1. *Fazer um programa Java que conta as frequências das letras em um texto.*

Exercício 2. *Seja $\{l_i\}_{i=1,\dots,n}$ as letras em um texto e $\{f_i\}_{i=1,\dots,n}$ suas frequências, encontrar as árvores de Huffman das mesmas. No primeiro exemplo, mostre que a árvore encontrada é ótima (enumerando as demais).*

(1) $l_{i=1,\dots,5} = (A, E, N, I, S)$ e $f_{i=1,\dots,5} = (29, 10, 9, 5, 4)$

(2) *Use o programa obtido no exercício 1 para calcular as $l_{i=1,\dots,}$ e $f_{i=1,\dots,}$ sobre o enunciado do próprio exercício 1.*

2. ÁRVORES BINÁRIAS

Exercício 3. *Dado um vetor $V[1, \dots, N]$ ordenado ($V[i] \leq V[i+j]$), construir uma árvore binária T tal que $\text{info}(\text{no}E) \leq \text{info}(\text{no}) \leq \text{info}(\text{no}D)$, $\forall \text{no} \in VT$, $\text{no}E \in \text{sub_arv}(\text{no.esq})$ e $\text{no}D \in \text{sub_arv}(\text{no.dir})$*

Exercício 4. *Usando o exercício acima, construa um método para fazer busca na árvore, com complexidade $\log(n)$, se n é o número de nós da árvore.*

Exercício 5. *Defina uma estrutura de nó (classe Java) para representações de árvores (não necessariamente binária). Defina o método para inserir um novo nó, filho de um determinado nó (os filhos precisam ter uma ordem bem definida, os primeiros estão mais à esquerda).*

Exercício 6. *Para a estrutura definida acima, construa métodos (algoritmos) para:*

- (1) *Passeio em “largura” na árvore (“Breadth First Search”);*
- (2) *Passeio em “profundidade” (“Depth First Search”);*
- (3) *Determinar o número de folhas da árvore a partir do nó $\text{no} \in VT$;*
- (4) *Determinar o número de folhas da árvore a partir do nó $\text{no} \in VT$;*
- (5) *Determinar o número de nós da árvore a partir do nó $\text{no} \in VT$;*
- (6) *Determinar a altura do nó $\text{no} \in VT$;*
- (7) *Determinar o nível do nó $\text{no} \in VT$.*

Exercício 7. *Refazer os exercícios 5 e 6 para árvores binárias, trocando o item (2) por passeis “pré-ordem”, “in-ordem” e “pós-ordem”.*

Exercício 8. *Refazer o passeio “in-ordem” do exercícios 7 para uma árvore binária com costura “in-ordem” (costura em filhos diretos, para o próximo “in-ordem”).*

Exercício 9. *Uma árvore T é do tipo “heap” se, e somente se, $\text{info}(\text{no}) \geq \max\{\text{info}(\text{no.esq}), \text{info}(\text{no.dir})\}$, $\forall \text{no} \in VT$.*

- (1) *Construir um algoritmo para inserir um novo nó numa árvore “heap”, mantendo a propriedade “heap”;*
- (2) *Simular inserções usando a seguinte sequência, (4, 5, 7, 2, 1, 3, 6) (desenhe a árvore após cada inserção).*

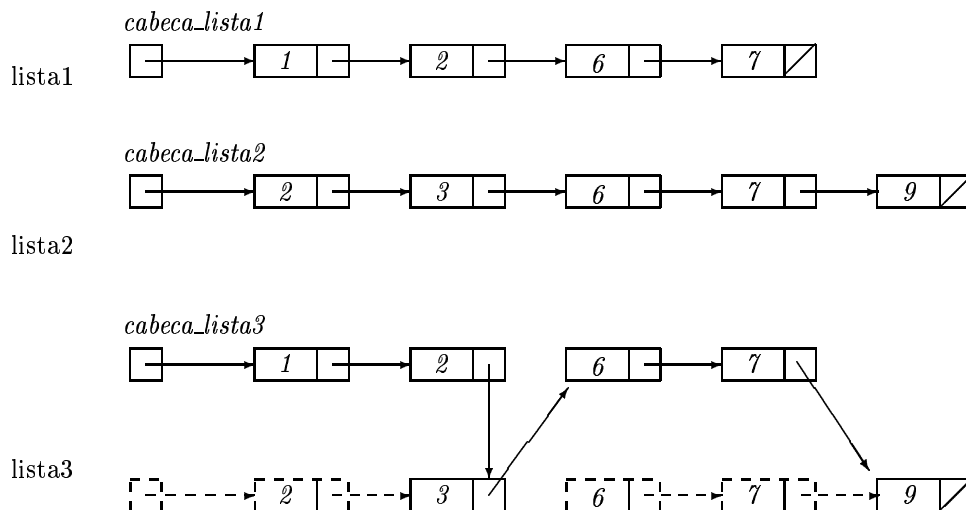
Exercício 10. *Converta a expressão aritmética $A - (B * C - 2 / (X + Y))^F * G$ para a correspondente em notação pós-fixa. Mantenha todas as configurações de saída e correspondente pilha.*

3. LISTAS

Exercício 11. Dadas duas listas ligadas, referenciadas por *lista1* e *lista2*, cujos elementos estejam em ordem crescente, escreva um algoritmo que intercale estas listas, formando uma única lista com os elementos em ordem estritamente crescente e sem repetições.

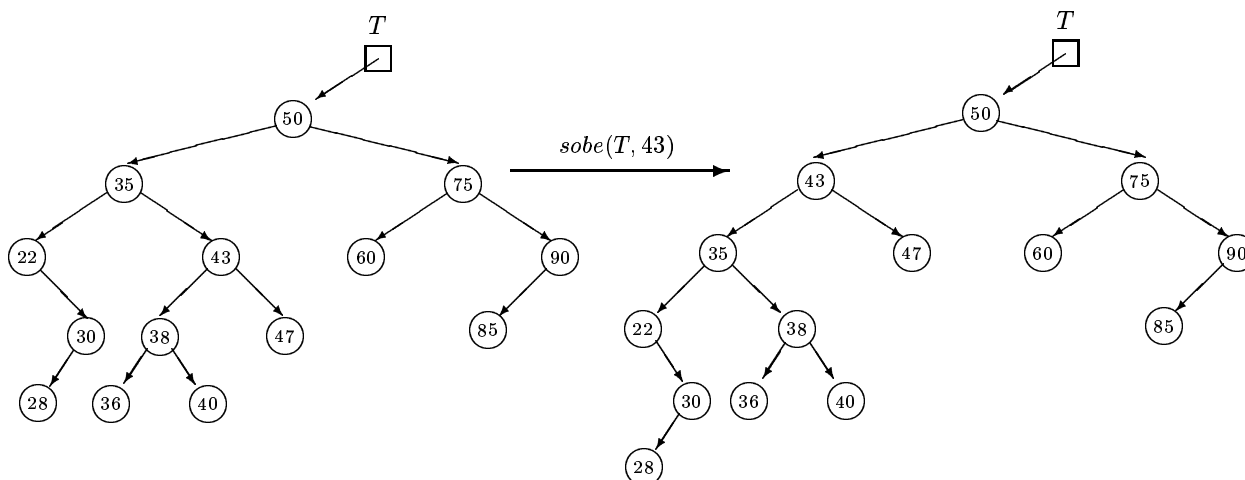
A lista resultante deve ser referenciada por *lista3* (a *lista3* deve ser formada pelas células que faziam parte de *lista1* ou de *lista2* — ao final *lista1* e *lista2* devem estar vazias).

Exemplo:



4. ÁRVORES DE BUSCA BINÁRIAS

Exercício 12. Considere os elementos de um conjunto armazenado numa **árvore de busca binária**. Escreva um algoritmo *Sobe*(*T*, *x*) que recebe uma árvore de busca binária referenciada por *T* e um elemento *x* e, se ele estiver na árvore e não for a raiz, sobe o nó contendo *x* de um nível, mantendo a estrutura de árvore de busca binária como mostra o exemplo a seguir:



5. B-ÁRVORES

Os exercícios 13, 14 e 15 referem-se à **B-árvores**. Uma **B-árvore** de grau *n* é:

- (1) uma árvore de busca *n*-ária (cada nó pode ter até *n* filhos);
- (2) cada nó pode armazenar até *n* - 1 chaves;

- (3) inserções são feitas sempre nas folhas;
- (4) todas as folhas estão no mesmo nível.

Deste modo, se K é um nó da B-árvore, então

- (1) $info_i(K)$ é o i -ésima chave do nó, $i \in \{1, \dots, n-1\}$, com

$$info_1(K) < info_2(K) < \dots < info_{n-1}(K);$$

- (2) denotando por T_{i-1} a sub-árvore esquerda da i -ésima chave do nó K e por T_{i+1} a sua sub-árvore direita, segue que

$$info_j(K_{i-1}) < info_i(K) < info_k(K_{i+1}), \forall (K_{i-1}, K_{i+1}, j, k) \in T_{i-1} \times T_{i+1} \times \{1, \dots, n-1\} \times \{1, \dots, n-1\}.$$

Exercício 13. Construa a **B-árvore** de grau 3, árvore 2 – 3, para a seguinte sequência de entrada: 1, 2, 3, 4, 5, 6, 7.

Exercício 14. Construa a **B-árvore** de grau 4, árvore 3 – 4, para a seguinte sequência de entrada: 1, 2, 3, 4, 5, 6, 7, 8.

Exercício 15. Mostre que em uma **B-árvore** de grau 2, T , não é possível ocorrer algum nó K que tenha uma chave i com $T_{i-1} \neq \emptyset$ e $T_{i+1} = \emptyset$. De modo mais formal:

$$\forall K \in T \Rightarrow \nexists i \in \{1, \dots, n-1\} \text{ tal que } T_{i-1} \neq \emptyset \text{ e } T_{i+1} = \emptyset.$$

6. MAIS HUFFMAN

Exercício 16. Se os caracteres (a, b, c, d, e) aparecem em um texto com frequências $(3, 2, 1, 1, 4)$, desenhe e calcule o custo das seguintes árvores binárias de código (como a de Huffman):

- (a) (0, 10, 110, 1110, 1111)
- (b) (00, 010, 0110, 0111, 1)
- (c) (00, 10, 110, 111, 01)

Exercício 17. Aplique, passo a passo, o algoritmo guloso de Huffman para encontrar a árvore ótima para a “carta” do exercício anterior.

Exercício 18. Seja T uma árvore de Huffman, obtida a partir de uma “carta” com as letras $\{l_i\}_{i=1,n}$, cujas frequências são respectivamente $\{f_i\}_{i=1,n}$. Seja c_l o caminho desde a raiz até a folha com a letra l e $\#c_l$ seu comprimento (o número de dígitos da letra l).

Mostre que: $f_i < f_j \Rightarrow \#c_{l_i} \geq \#c_{l_j}$.