EDA - Home Work 3

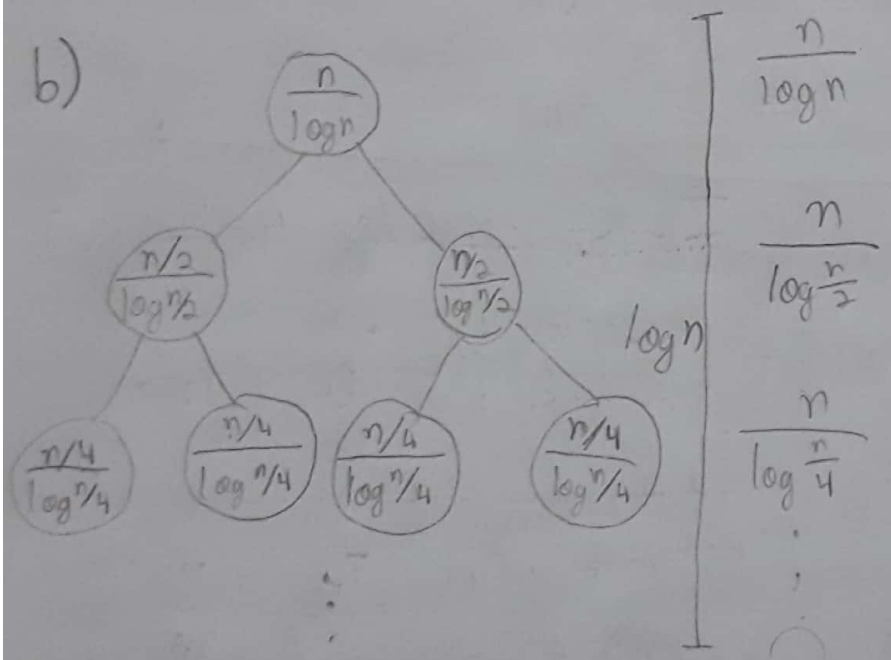
João Pedro de Abreu Marciano

① B, E, D, C, B, A

② a) Utilizando o método Mestre:

$$T(n) = \begin{cases} O(n^k \log n) \text{ se } 2 = 2^k \Rightarrow k = 1 \\ O(n^k) \text{ se } 2 < 2^k \Rightarrow k > 1 \\ O(n^{\log_2 2}) = O(n) \text{ se } 2 > 2^k \Rightarrow 0 < k < 1 \end{cases}$$

b)



Supondo $n = 2^k \Rightarrow \log n = k$

Utilizando a árvore:

$$T(n) = \sum_{i=1}^{k-1} \frac{n}{\log \frac{n}{2^i}} + \frac{n}{\log n} + O(n)$$

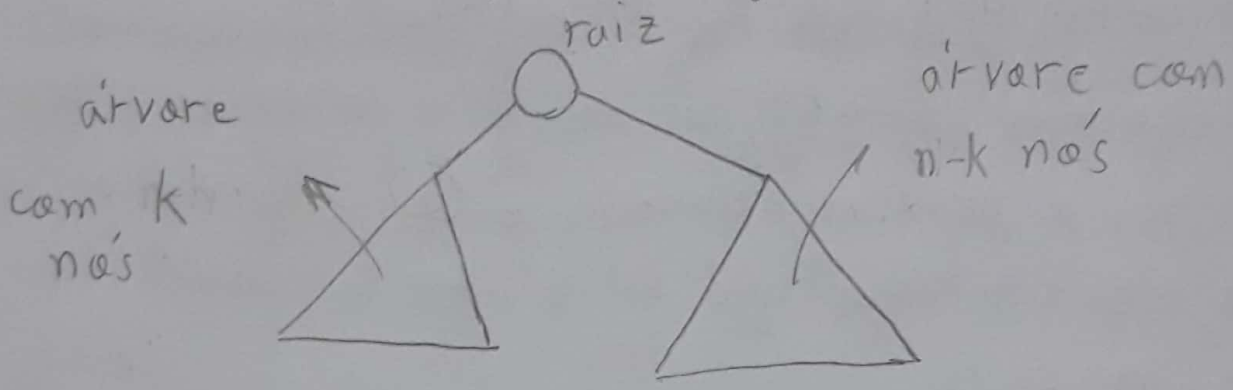
$$= n \cdot \sum_{i=0}^{k-1} \frac{1}{k-i} + O(n) = n \cdot \sum_{j=1}^{k} \frac{1}{j} + O(n) = O(n \log \log n)$$

③ Base: $n=1$ é trivial ver que funciona.

Vamos supor que é válido para todo $1 \leq k \leq n$.

Dado uma árvore de $n+1$ nós e um conjunto com $n+1$ chaves distintas.

Vamos supor que o seguinte ocorre:



A chave da raiz deve ser maior que os k elementos da esquerda e menor que os $n-k$ elementos da direita. Logo, ordenamos a lista de chaves e escolhemo a da $(k+1)^a$ posição. Os k nós da esquerda com os k primeiros elementos formam uma árvore única, pela hipótese de indução. O mesma vale para os $n-k$ nós da direita com os $n-k$ últimos elementos da lista de chaves.

Então, por indução, a árvore de pesquisa binária é única.

---

④ a) Postorder: M, W, Y, I, P, S, E, B, O

Postorder reverso: O, B, E, S, P, I, Y, W, M

b) {{O}, {B, E, Y}, {S, W}, {I, M, P}}

Logo: O, B, E, Y, S, W, I, M, P

c) Topological sort utilizado será o preorder:

O, B, E, S, I, W, M, Y, P.

---

⑤ a) O primeiro relaxado foi 3., pois dele começamos. Após, seguimos 3 → 0, pois este é o de menor peso, ou seja, o próximo relaxado é O. Seguindo a mesma ideia, temos que os 5 vértices são:

3, 0, 10, 5, 2

b)

	v	distTo[ ]	edgeTo[ ]
	0	1	3 → 0
	1	17	5 → 1
	2	6	5 → 2
	3	0	null
	4	7	0 → 4
	5	5	10 → 5
→	6	11	4 → 6
→	7	10	4 → 7
→	8	8	4 → 8
→	9	22	4 → 9
	10	3	3 → 10

Valores que mudaram são os →

⑥ a) Errado, pois claramente vale que

$$P_{h \in H}\left(h(u_x) = h(u_y)\right) = \frac{1}{n}$$

b) Errado, já que armazenar um elemento de H_1 tem a mesma custo de memória que armazenar um elemento de H_2.

c) Certo, pois o número de elementos de H_1 é $n^{|U|} = n^m$, enquanto $|H_2| = p(p-1) = m(m-1)$ Ou seja, $|H_1| = O(n^m)$ e $|H_2| = O(m^2)$.

---

⑦ Se $n = 1$ e $A[1] \neq$: retorna 0

Se $A[\lceil \frac{n}{2} \rceil] = \lceil \frac{n}{2} \rceil$: retorna $\lceil \frac{n}{2} \rceil$

Se $A[\lceil \frac{n}{2} \rceil] < \lceil \frac{n}{2} \rceil$:

chama o mesmo algoritmo denovo só que em $A[\lceil \frac{n}{2} \rceil + 1 : n]$ e retorna esse valor mais $\lceil \frac{n}{2} \rceil$.

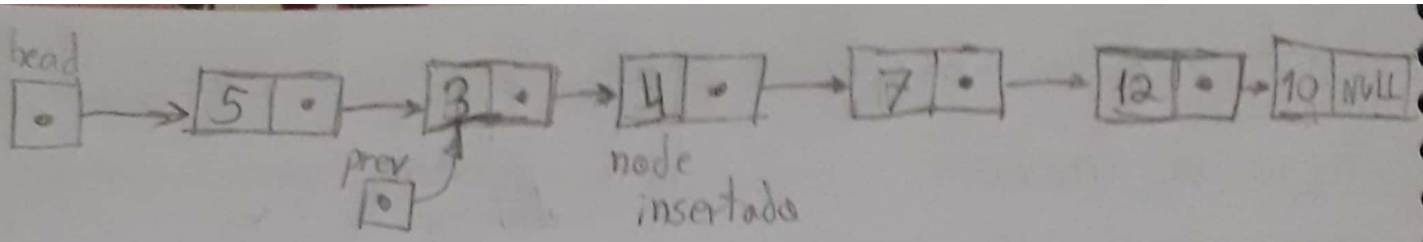
Se $A[\lceil \frac{n}{2} \rceil] > \lceil \frac{n}{2} \rceil$:

chama o mesmo algoritmo denovo só que em $A[1 : \lceil \frac{n}{2} \rceil]$ e retorna esse valor.

Fim.

Esse algoritmo dá como resultado o índice e, caso não haja, dá 0.

⑧ a)



b) prev → item é 3.

c)