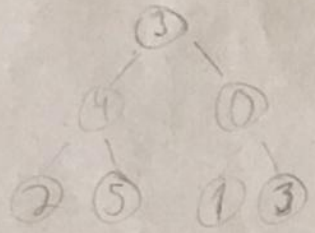
Gabriel Catizani Faria Oliveira - 20.1.4004

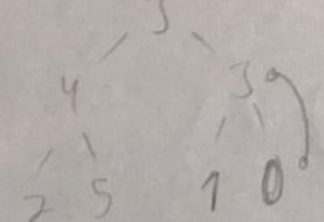
HeapSort 3 4 0 2 5 1 3

① Vamos realizar a construção do heap



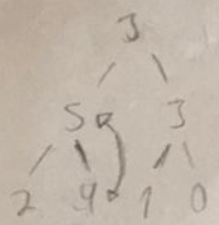
② Como queremos em ordem crescente, precisamos de colocar o maior valor na raiz. Portanto começamos comparando lo de baixo

③ 1, 3? Qual é o maior? 3! Logo, agora comparamos 0 com 3. 3 é maior. Logo, os trocamos de posição. Agora verificamos o 0 e vemos que é um nó folha. Logo, está correto e temos uma subárvore válida e, assim, temos no fim o seguinte:



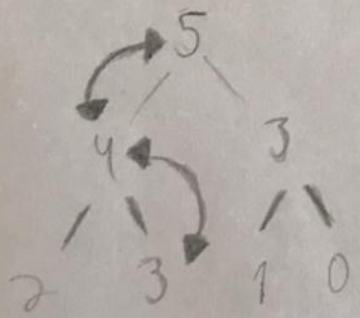
④ 2, 5? O 5 é maior! Logo, agora comparamos 5 com 4. 5 é maior. Logo, os trocamos de posição. Agora verificamos o 4 e vemos que é um nó folha. Logo, está correto e temos uma subárvore válida

(5ª) Agora com a árvore construída inici... troca...



(5ª) 5 e 3 ? 0 5 é maior. Logo, agora comparamos 5 com 3 do raiz (nó pai). 5 é maior. Logo, os trocamos de posições. Agora, verificamos o 3 que estava na raiz e ele tem nós filhos. Como ele tem verificamos se o subarvore é válido. Percebe-se que não já que 4 é maior que 3 (2, 4 e 4 é maior). Logo, trocamos o 4 e o 3 de posição.

(6ª) Agora, verificamos o 3 novamente e vemos que ele é um nó folha. Logo, o subárvore é válida.

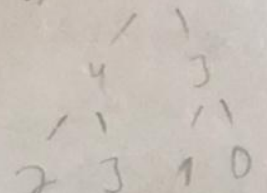


(7º) Agora com o heap construído iniciaremos o processo de ordenação

- Ordenação

	4	3	2	3	1	0

1) Removemos o raiz



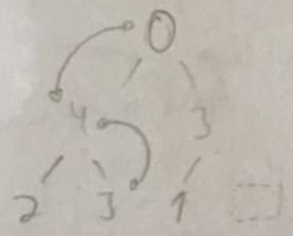
Item removido: 5

↳ Q trocamos com o item do ultimo posição

Assim fica

2) Colocamos no raiz o ultimo elemento

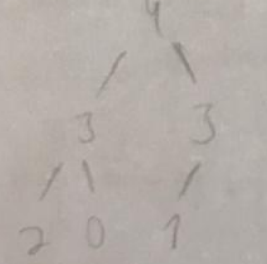
0	4	3	2	3	1	5



Agora refazemos o heap

3) Refazendo o heap

0	1	2	3	4	5	6
4	3	3	2	0	1	5



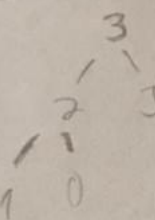
4) Removemos novamente o raiz e coloco na posição dele o ultimo elemento

Item removido: 4 !

1	3	3	2	0	4	5

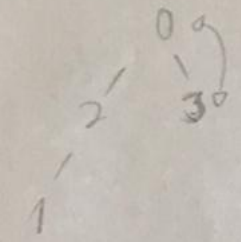
Refazemos o heap, assim fica

5) Refazemos o heap



0	1	2	3	4	5	6
3	2	3	1	0	4	5

6) Removemos a raiz e colocamos no posição dele o último elemento da árvore

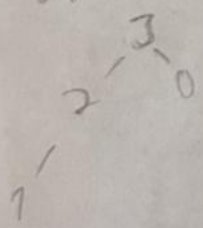


Item removido: 3

0	1	2	3	4	5
0	2	3	1	3	4	5

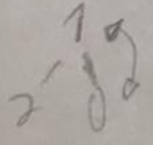
Refazemos o heap e ainda fica:

7) Refazemos o heap



0	1	2	3	4	5	6
3	2	0	1	3	4	5

8) Removemos a raiz e colocamos no posição dele o último elemento da árvore

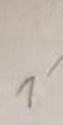


Item removido: 3

0	1	2	3	4	5
1	2	0	3	3	4	5

Refazemos o heap e ainda fica:

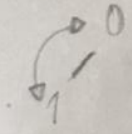
9) Refazemos o heap



0	1	2	3	4	5	6
2	1	0	3	3	4	5

10) Removemos normalmente o raiz e colocamos na posição dele o último elemento do árvore

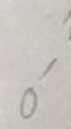
Item removido: 2



0	1	2	3	3	4	5

Refazemos o heap

11) Refazemos o heap



0	1	2	3	4	5	6
1	0	2	3	3	4	5

12) Removemos normalmente o raiz e colocamos na posição dele o último elemento do árvore.

0

6

0	1	2	3	4	5	6
0	1	2	3	3	4	5

Como só sobrou o 0, não precisa refazer o heap.

Portanto, já temos o vetor ordenado de forma crescente

✓