PROGRAMAÇÃO ESTRUTURADA T2

Grupo: Amarildo Junior, Leonardo Coutinho, Elias Júnior.

Parte 1: Tempo de ordenação dos vetores.
Máquina utilizada nos testes:
Desktop
Processador: R7 2700 3.2GHz 8 cores 16 threads
Ram: 8gb | SSD: 240GB SATA |HD: 2Tb | Placa de Vídeo: RX570 4GB
SO: Windows 10 Pro 64 bit's

	Bubble	Selection	Insertion	Quick	Merge	Heap
Vetor A	0.087	0.032	0.017	0.000	0.000	0.001
Vetor B	0.359	0.122	0.068	0.002	0.001	0.002
Vetor C	1.512	0.490	0.273	0.002	0.003	0.004
Vetor D	4.277	1.096	1.162	0.003	0.006	0.004

Parte 2: Explicação da ordenação utilizando Mergesort e Heapsort de maneira **não-crescente**.

Vetor a ser ordenado:

5	1	9	0	8	3	1	4	6	2

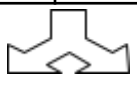
Mergesort:

O ***mergesort***, ou ordenação por mistura, é um exemplo de algoritmo de ordenação por comparação do tipo dividir para conquistar.

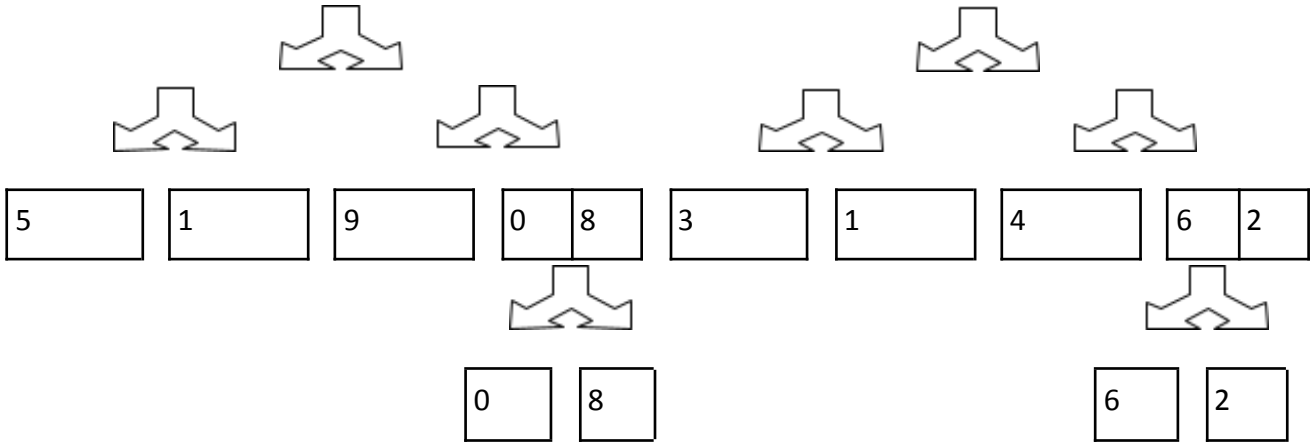
Sua ideia básica consiste em Dividir (o problema em vários subproblemas e resolver esses subproblemas através da recursividade) e conquistar (após todos os subproblemas terem sido resolvidos ocorre a conquista que é a união das resoluções dos subproblemas).

O vetor será dividido pela metade até sobrarem apenas itens individuais:

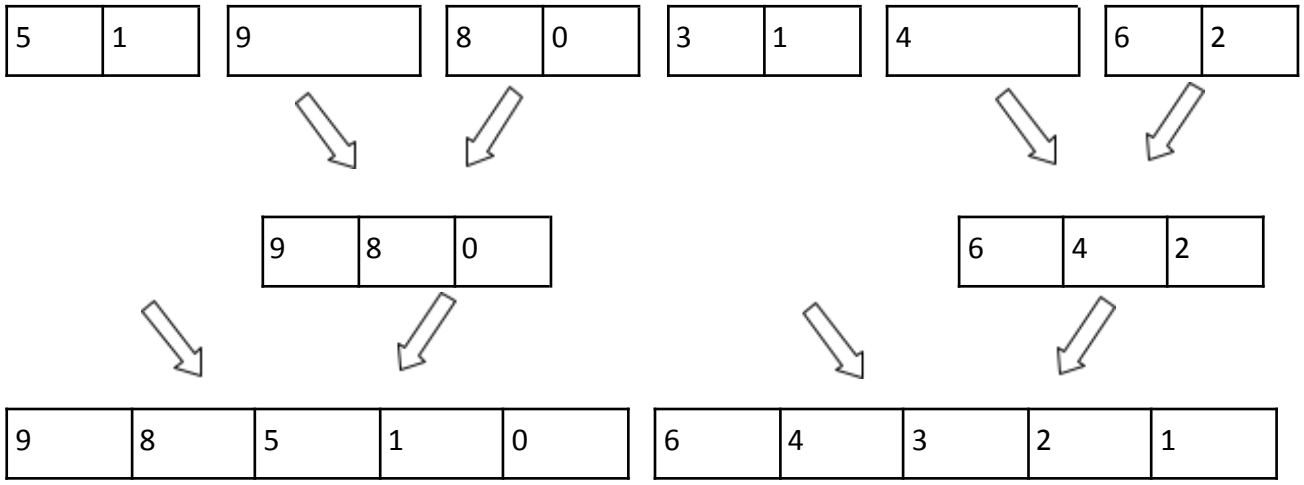
5	1	9	0	8	3	1	4	6	2



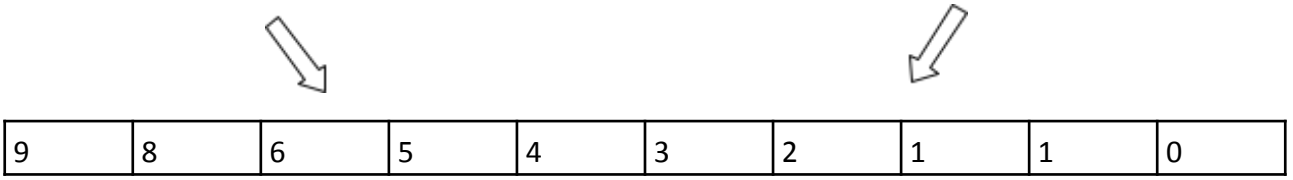
5	1	9	0	8	3	1	4	6	2



Com todos os itens do vetor já divididos em partes individuais eles serão colocados em vetores temporários de maneira já ordenada (ordenação feita no procedimento recursivo):



Quando restam apenas dois vetores temporários já ordenados eles são colocados no vetor principal com a ordenação completa.

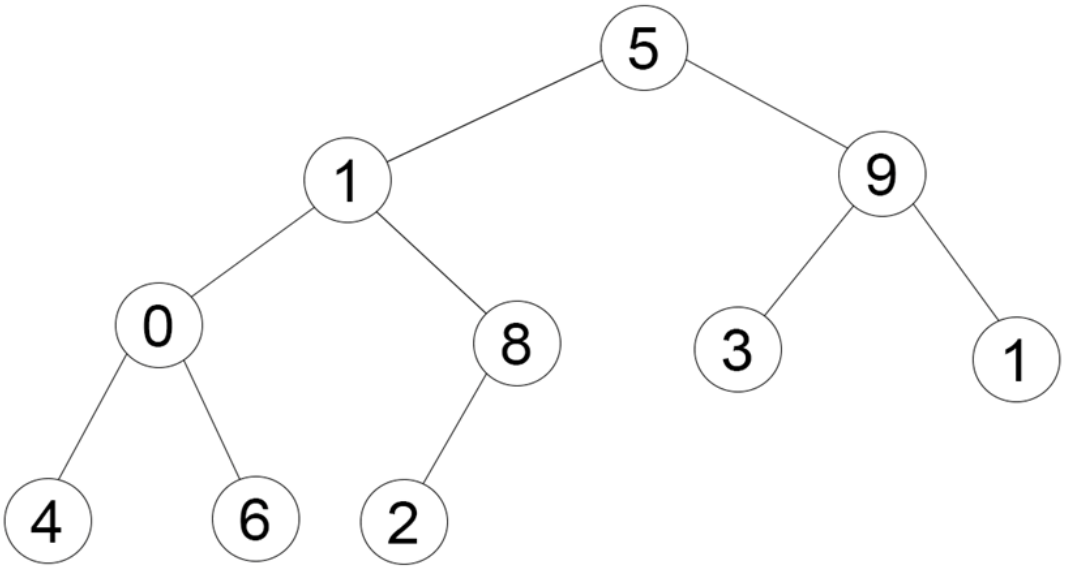


Heapsort:

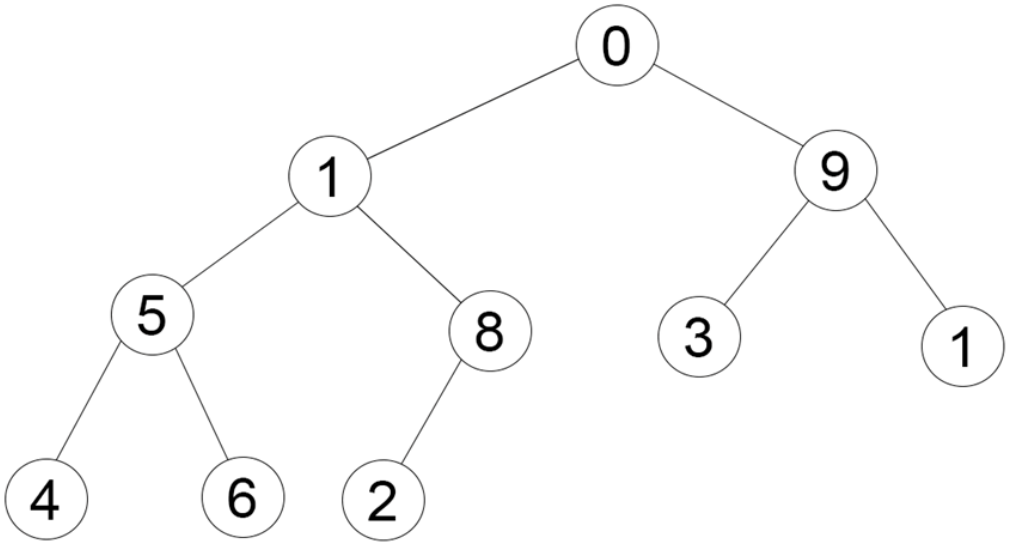
O heapsort utiliza uma estrutura de dados chamada heap, para ordenar os elementos à medida que os insere na estrutura. Assim, ao final das inserções, os elementos podem ser sucessivamente removidos da raiz da heap, na ordem desejada, lembrando-se sempre de manter a propriedade de max-heap(pai>filho).

5	1	9	0	8	3	1	4	6	2

Ao executar o algoritmo ele percorrerá o vetor e criará uma árvore com os elementos. Da esquerda para a direita é criada uma árvore com os valores encontrados no vetor. A árvore ficará com a seguinte organização:

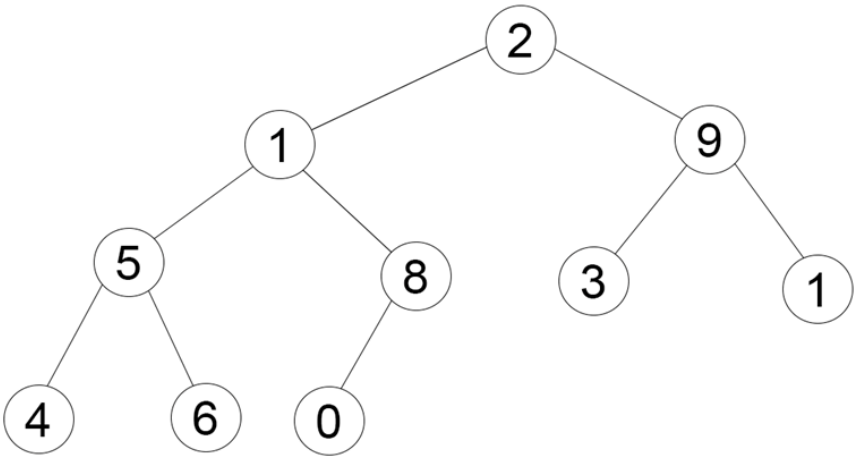


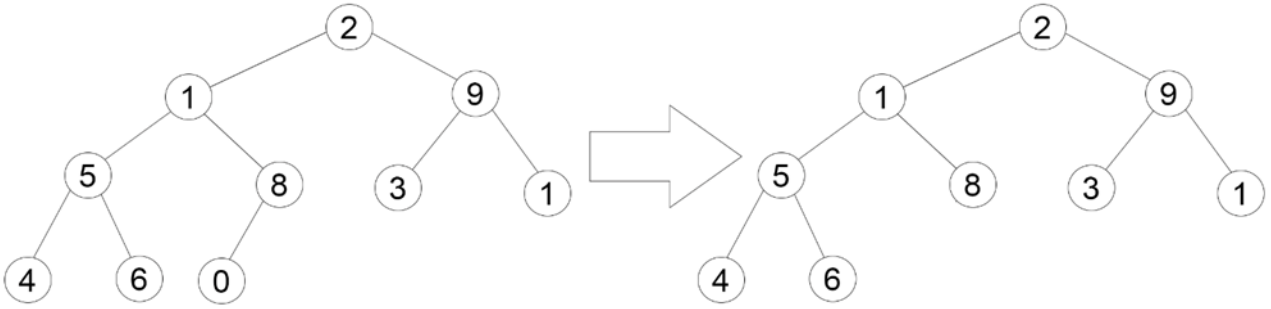
Após a estruturação da árvore será encontrado o maior valor do vetor, este valor será o max_heap, nesse caso(ordenação decrescente, o valor a ser posto no topo é o menor) o valor 0 será trocado com o valor 5.



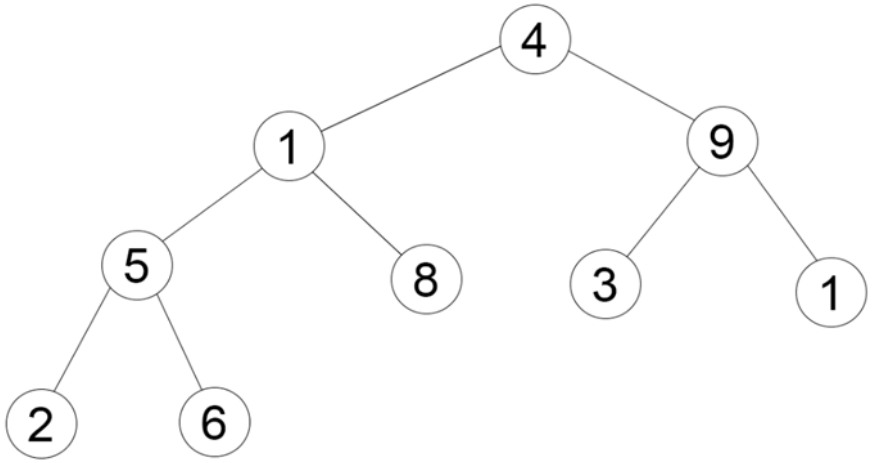
0	1	9	5	8	3	1	4	6	2

Encontrado o max_heap é trocado o valor da primeira posição com o valor da última:
O número 0 já é considerado ordenado e está em sua posição final.



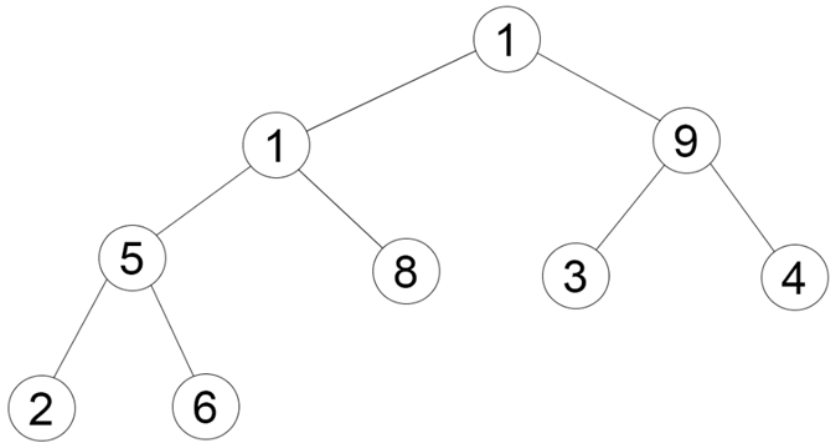


O número 2 vai para a última posição do nó da esquerda e é substituído pelo menor valor encontrado (4 nesse caso).



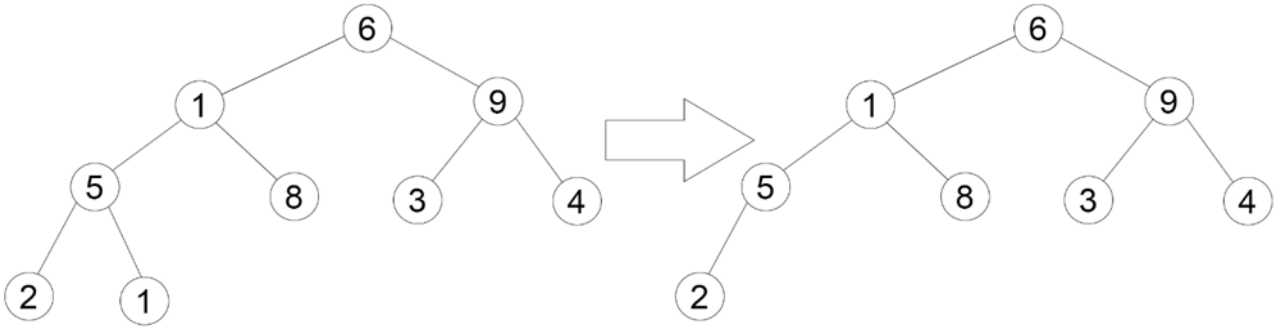
4	1	9	5	8	3	1	2	6	0

O número 1 mais abaixo vai para a posição max_heap e é substituído pelo valor max_heap encontrado (4 nesse caso).



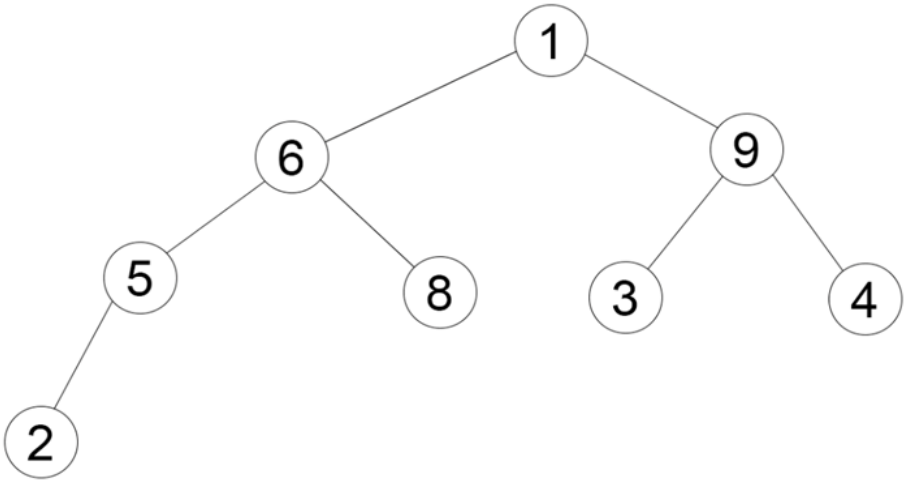
1	1	9	5	8	3	4	2	6	0

O max_heap (número 1) é substituído pela última posição (número 6) e é excluído da árvore.



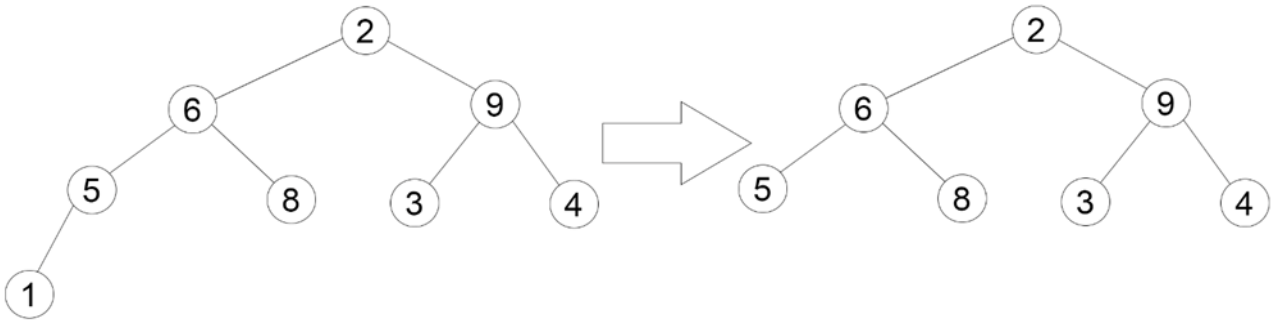
6	1	9	5	8	3	4	2	1	0

É verificado que o menor número não está no max_heap, então troca o menor número (número 1) com o max_heap (número 6).



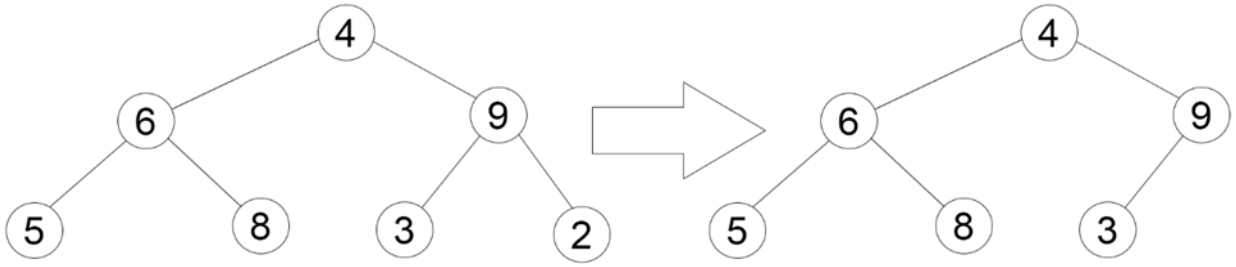
1	6	9	5	8	3	4	2	1	0

Verificado que o max_heap é o menor número (número 1), então é trocado com o último nó da árvore (número 2), após a troca está ordenado então é excluído da árvore.



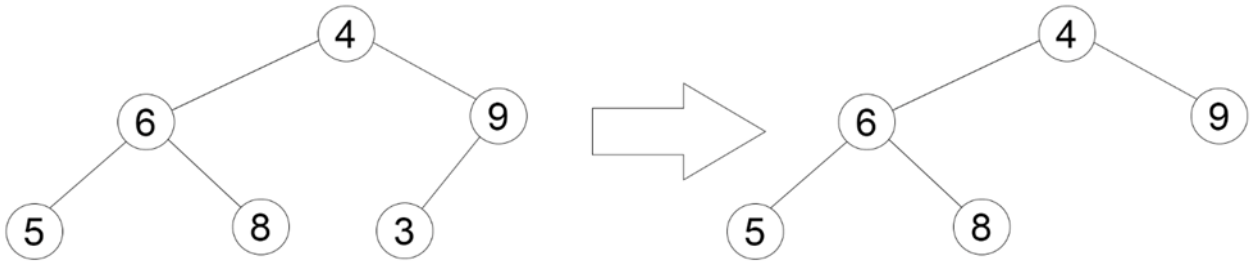
2	6	9	5	8	3	4	1	1	0

Verificado que o max_heap novamente é o menor número da arvore, trocasse então o 2 com 4, nó mais a direita da árvore, verificado então que o 3 está na posição ordenada é excluído.



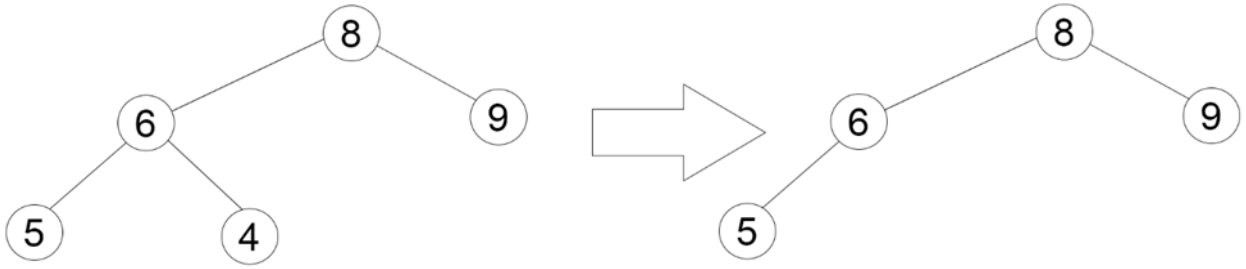
4	6	9	5	8	3	2	1	1	0

Verificado que a última posição (número 3) é a menor posição da arvore, então apenas exclui a menor posição da árvore, pois já está ordenada.



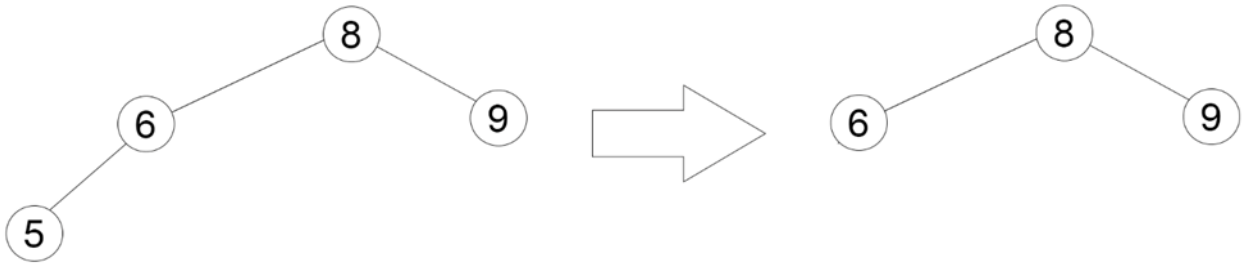
4	6	9	5	8	3	2	1	1	0

Verificado que o max_heap é a menor valor da árvore, então troco o 4 com o 8 e excluo o 4, pois está ordenado.



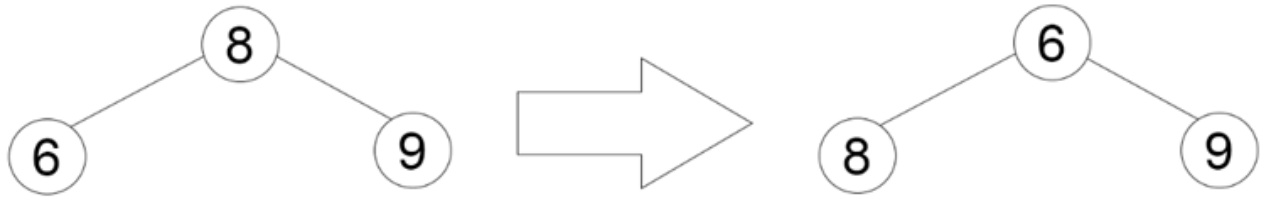
8	6	9	5	4	3	2	1	1	0

Verificado o menor número da árvore já está na extremidade da árvore, então apenas exclui o número 5 da árvore.



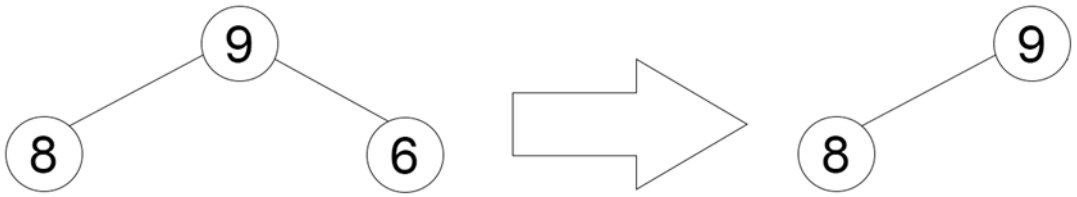
8	6	9	5	4	3	2	1	1	0

Verificado que o nó da árvore mais à esquerda é o menor valor da árvore, então trocasse o menor valor com o max_heap (número 8).



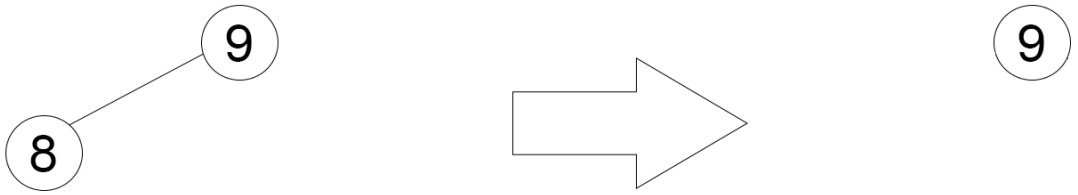
6	8	9	5	4	3	2	1	1	0

Verificado que o menor valor da árvore é o max_heap e o nó mais à direita é o maior valor da árvore, então trocasse o número 6 com o 9 e exclui o nó mais à direita da árvore (número 6).



9	8	6	5	4	3	2	1	1	0

Verificado que o nó mais à esquerda é o menor valor restante na árvore, então excluísse o nó mais à esquerda da árvore e o vetor está ordenado de forma não-crescente.



9	8	6	5	4	3	2	1	1	0