Faculdade de Informática

PUCRS

Algoritmos e Estruturas de Dados II

Aula #5 – Filas de Prioridade (min-heap e max-heap)

Prof. Leonardo Heredia

PUCRS

Heap

- Estrutura de dados.
- Também chamado de heap binário.
- São árvores binárias completas e perfeitamente balanceadas (exceto no último nível).
- Árvores binárias, porém não são Árvores Binárias de pesquisa.

Heap mínimo
Nodo **pai** é **MENOR** ou igual aos dois nodos filhos

Heap máximo
Nodo **pai** é **MAIOR** ou igual aos dois nodos filhos

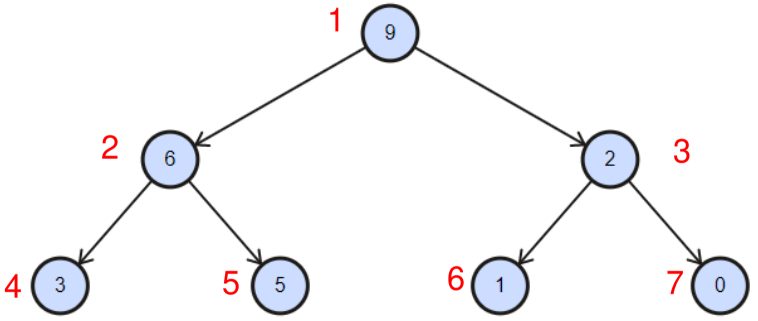
Vamos utilizar o heap-max nos exemplos

http://btv.melezinek.cz/binary-heap.html

Heap máximo (max-heap)

- A raiz SEMPRE contém a chave com maior valor.
- Heap binário representado/implementado através de um array.
- Maior chave está na raiz, que é a raiz da árvore binária.
- Os índices dos arrays podem ser utilizar para se movimentar através da árvore:
  - Pai do nodo k está em k / 2
  - Filhos do nodo k estão em:
    - 2k (filho a esquerda)
    - 2k + 1 (filho a direita)

Heap A – árvore binária



Heap h – array

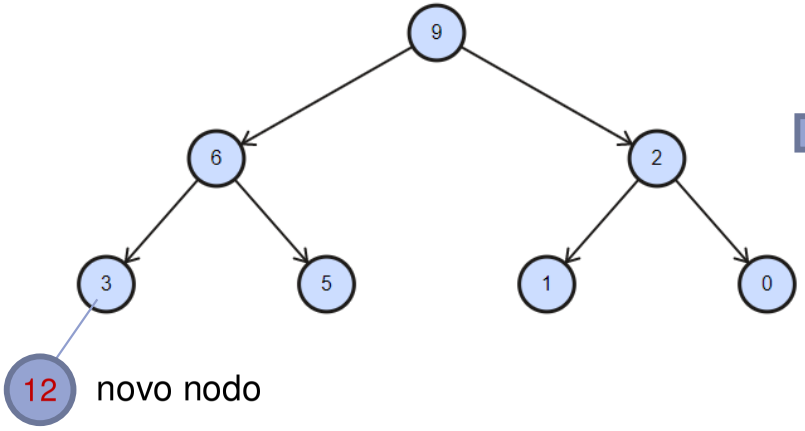
chave	A(i)	-	9	6	2	3	5	1	0
	i	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*

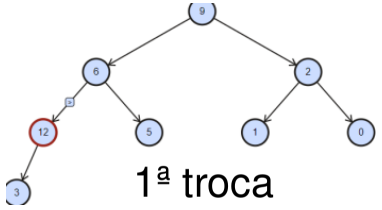
Exemplo:
- Pai do nodo **3** (k) → A(3 / 2) = A(1) = **9**
- Filhos do nodo **3** (k)
  - → A(2k) = A(2*3) = A(6) = **1**
  - → A(2k + 1) = A(2*3 + 1) = A(7) = 0

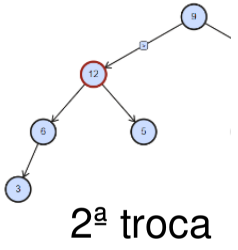
Inserção no heap-max

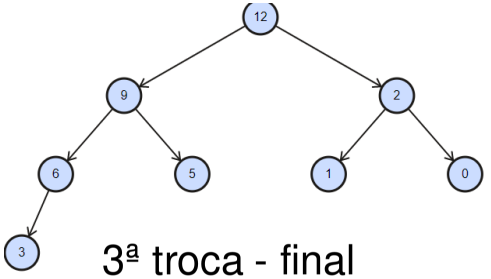
- Adiciona a nova chave sempre no final, uma posição após o último nodo.
- Após adicionar a chave o heap pode ficar inconsistente, o um nodo filho maior que o pai.
- Quando isso ocorre o nodo inserindo deve ir "nadando" (operação *swim*) para cima até chegar em uma posição em que ele não seja maior que o pai.

Heap A – árvore binária









Heap h – array Heap inconsistente

chave	A(i)	-	9	6	2	3	5	1	0	12
	i	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*

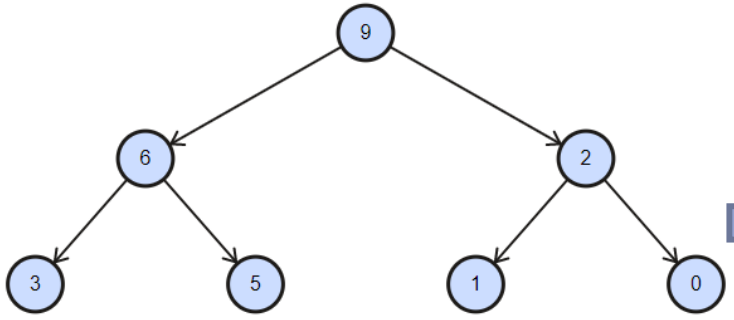
A(i)	-	12	9	2	6	5	1	0	3
i	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*

Heap consistente

Remoção do valor máximo

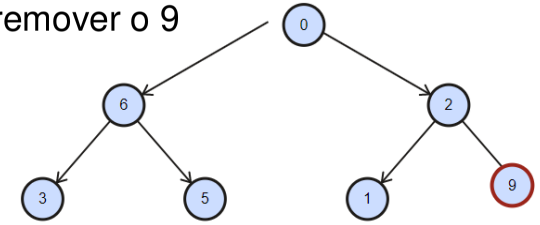
- No heap máximo a remoção do elemento da maior chave consiste na remoção do nodo raiz.
- Deve-se trocar a raiz com o nodo final e então a raiz “afunda”, operação *sink*.

Heap A – árvore binária



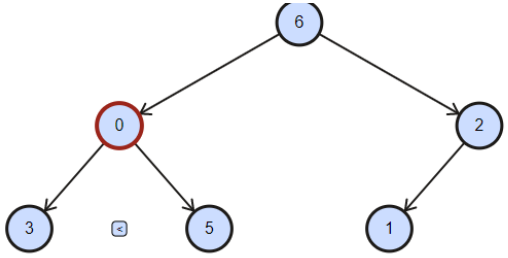
Remover a maior chave

a) Trocar o 9 pelo 0, remover o 9

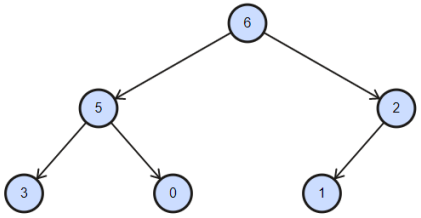


O 0 vai “afundando” até o heap ficar consistente

b) Trocar 0 com o 6



c) Trocar 0 com o 3



Heap h – array

chave

A(i)	-	9	6	2	3	5	1	0
i	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*

A(i)	-	6	5	2	3	0	1	
i	*0*	*1*	*2*	*3*	*4*	*5*	*6*	

Operação swim (nadar)

- Realizada quando a **chave do filho** se torna **maior** que a do **pai**.
- Algoritmo:
  - Trocar a chave do filho com a chave do pai
  - Repetir até que a ordem do heap seja restaurada.

Operação sink (afundar)

- Realizada quando a **chave do pai** se torna **menor** que a do **filho maior**.
- Algoritmo:
  - Trocar a chave do com a do filho maior
  - Repetir até que a ordem do heap seja restaurada

Exercícios

1. Construa um heap máximo inserindo os seguintes inteiros: 23, 40, 21, 33

   Como ficou o heap?

2. Adicione 56, 15 e 22. Como ficou?
3. Construa o heap inserindo 50, 12, 30, 59.
4. Remova a raiz. Como ficou?
5. Desenhe a árvore binária do array abaixo:

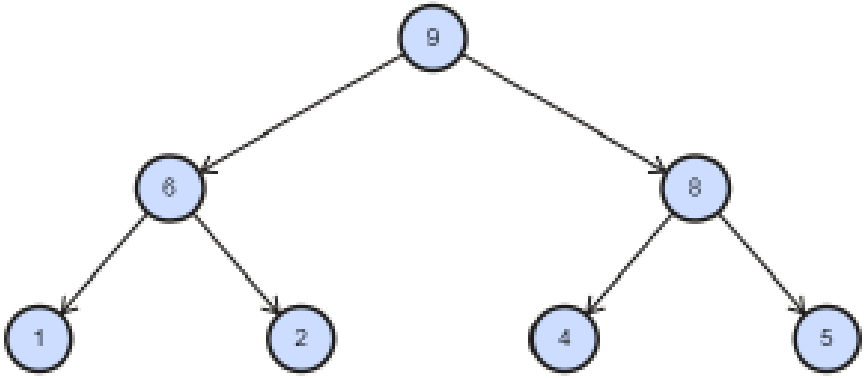
   [15, 23, 30, 11, 50, 33]

6. O array anterior é um heap máximo?
7. Caso negativo converta o mesmo em um heap máximo.

PUCRS

Exercícios (prova anterior)

1) (1,0 ponto) Um heap máximo é uma árvore binária (completa ou quase completa) da esquerda para a direita, onde a chave de cada nodo pai não pode ser menor que a chave dos filhos.
Por exemplo:



Considerando a árvore binária acima que representa um heap máximo, responda aos itens abaixo:

a) Qual a complexidade de acesso ao elemento mais prioritário do heap? O(n) ou O(1) ou O(log n)?

b) Apresente o array correspondente a esse heap.

2) (0,5 ponto) Dado o array de inteiros abaixo, converta ele em um heap máximo. Apresente a árvore e o array final:
Array = [1, -5, 87, 58, 13, 12, 5]

3) (1,0 ponto) Adicione ao heap gerado na questão 2 os valores 4 e 18 e depois remova a raiz, qual heap final? Apresente a árvore e o array.