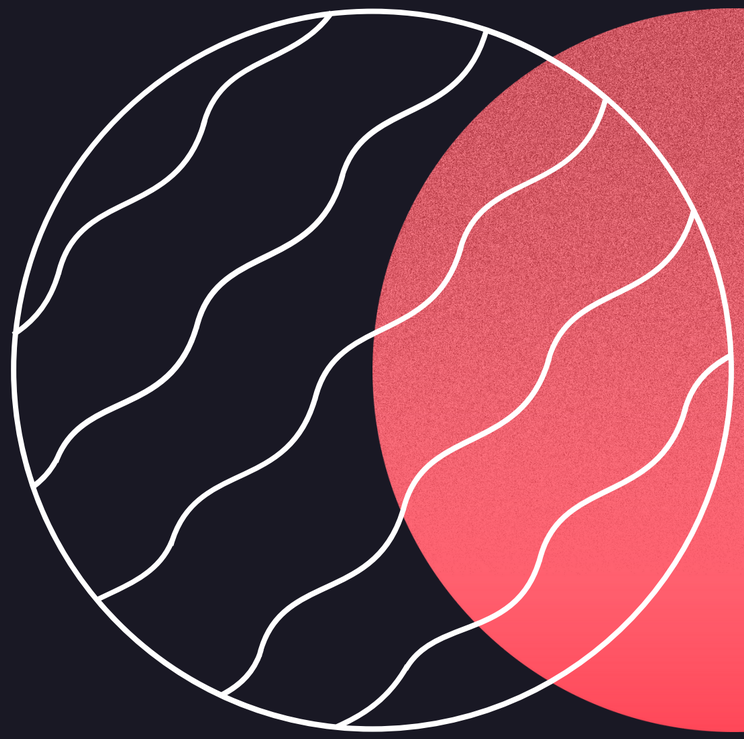


HEAP

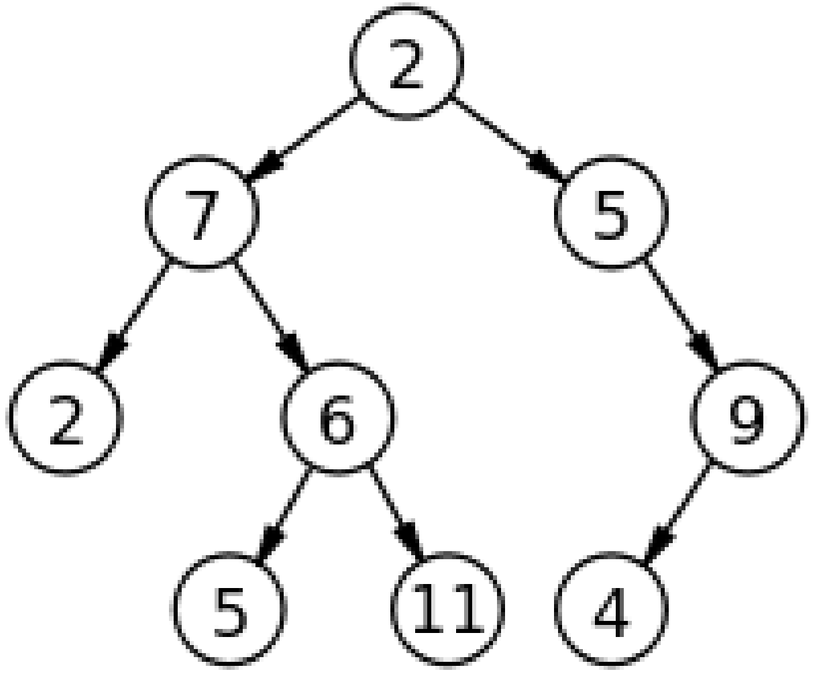
Augusto Cressembini
Camila Berbert
Mario Honda
Matheus Bonilha
Thalita Rodrigues

- **Estrutura de Dados**
- **Funcionamento da Árvore**
- **Aplicação**
- **Algoritmo**

O QUE É HEAP?

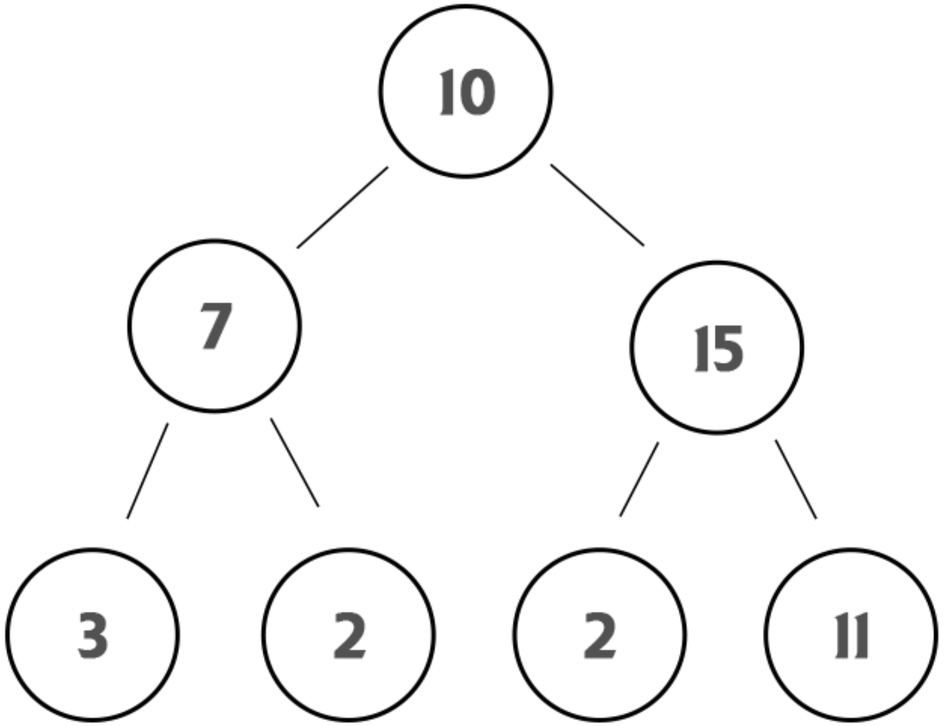
Heap é um tipo específico de Árvore Binária, mas para isso, é essencial que o conceito de árvore Binária esteja bem esclarecido para todos.

ÁRVORE BINÁRIA



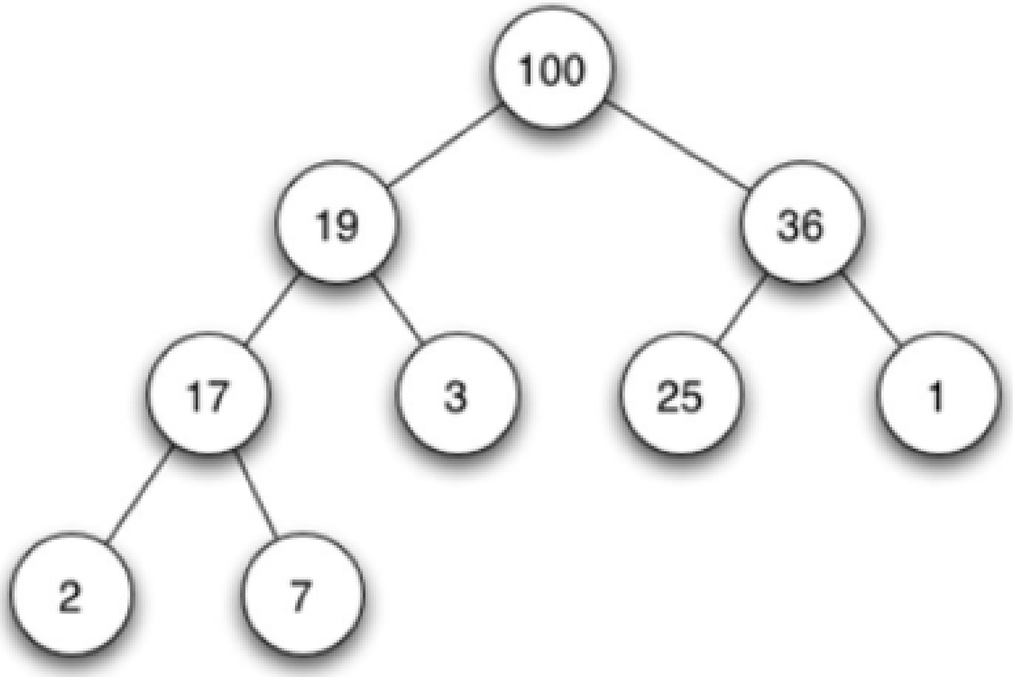
- Cada nó pode conter nenhum, 1 ou 2 filhos;
- A árvore binária é formada por nós, onde cada nó contém uma referência para outros 2 nós (filho esquerdo e filho direito);
- Uma árvore só é estritamente binária quando todo nó que não é folha possuir sub árvores esquerda e direita não vazias; ou seja, todo nó que não é folha tem que ter 2 filhos.

ÁRVORE BINÁRIA CHEIA



- Os nós folhas se encontram no último nível, e possuem 0 ou 2 filhos; e os nós filhos sempre se encontram no último nível.

ÁRVORE BINÁRIA COMPLETA



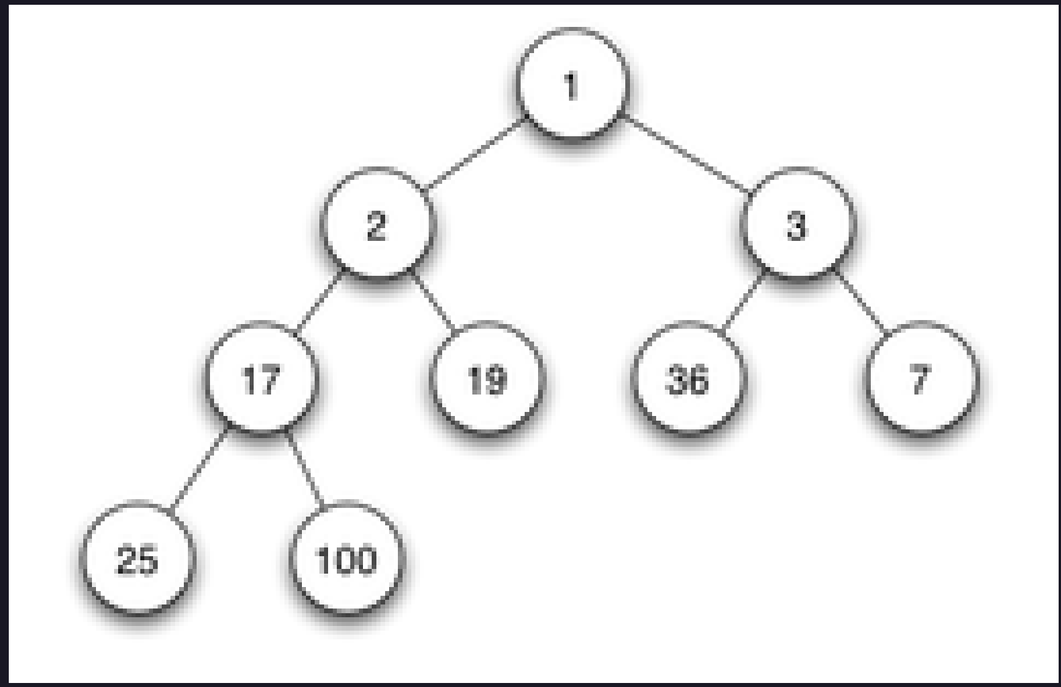
- E uma árvore cheia até o penúltimo nível, no último nível todos os nós devem estar mais à esquerda possível.

HEAP?

É uma estrutura de prioridades, na forma de Árvore Binária Completa, que representa uma ordem parcial entre os elementos do conjunto.

CARACTERÍSTICAS DO HEAP

- A árvore está completamente preenchida em todos os níveis exceto, talvez, o mais baixo;
- Nível mais baixo preenchido a partir da esquerda.

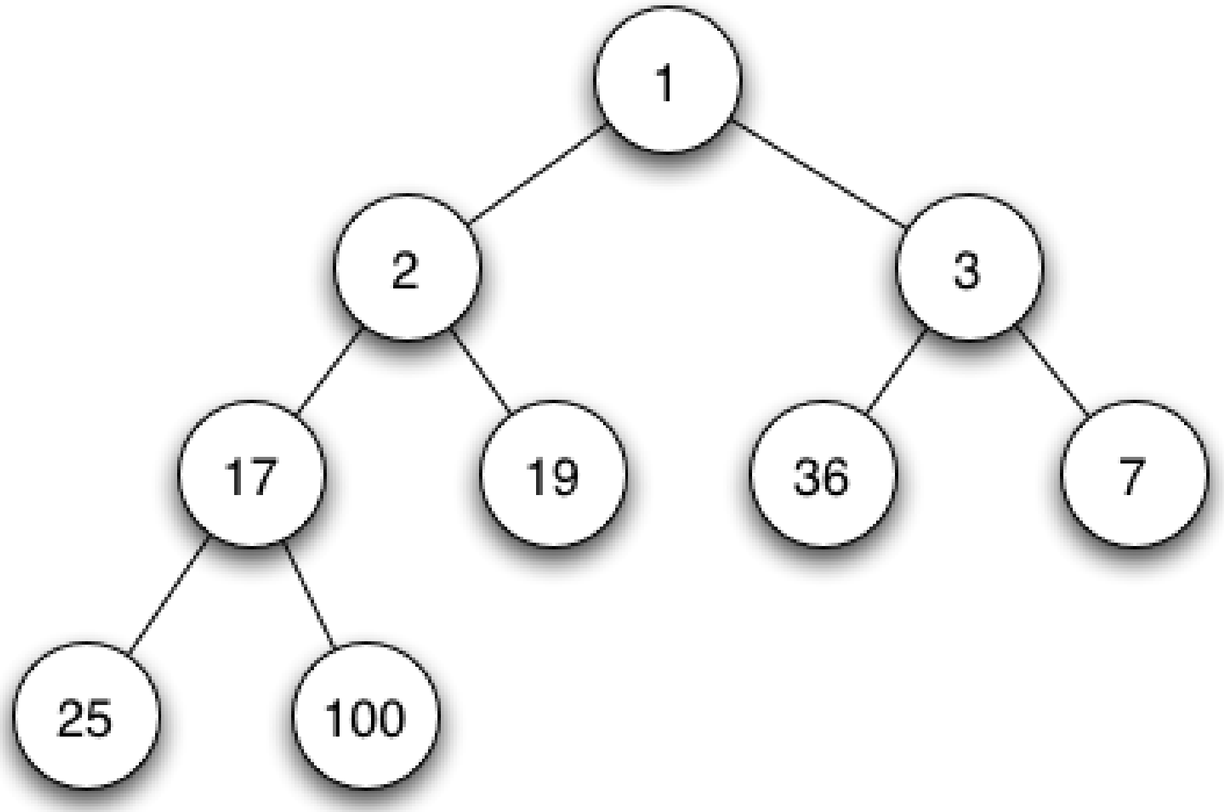


TIPOS HEAP

- **HEAP MÍNIMO:**

No heap mínimo o nó raiz vai conter a menor chave, de forma que todo nó filho é maior ou igual ao nó pai (esquerda e direita); ou seja, **o menor elemento sempre estará guardado na Raiz**.

HEAP MÍNIMO EXEMPLO:

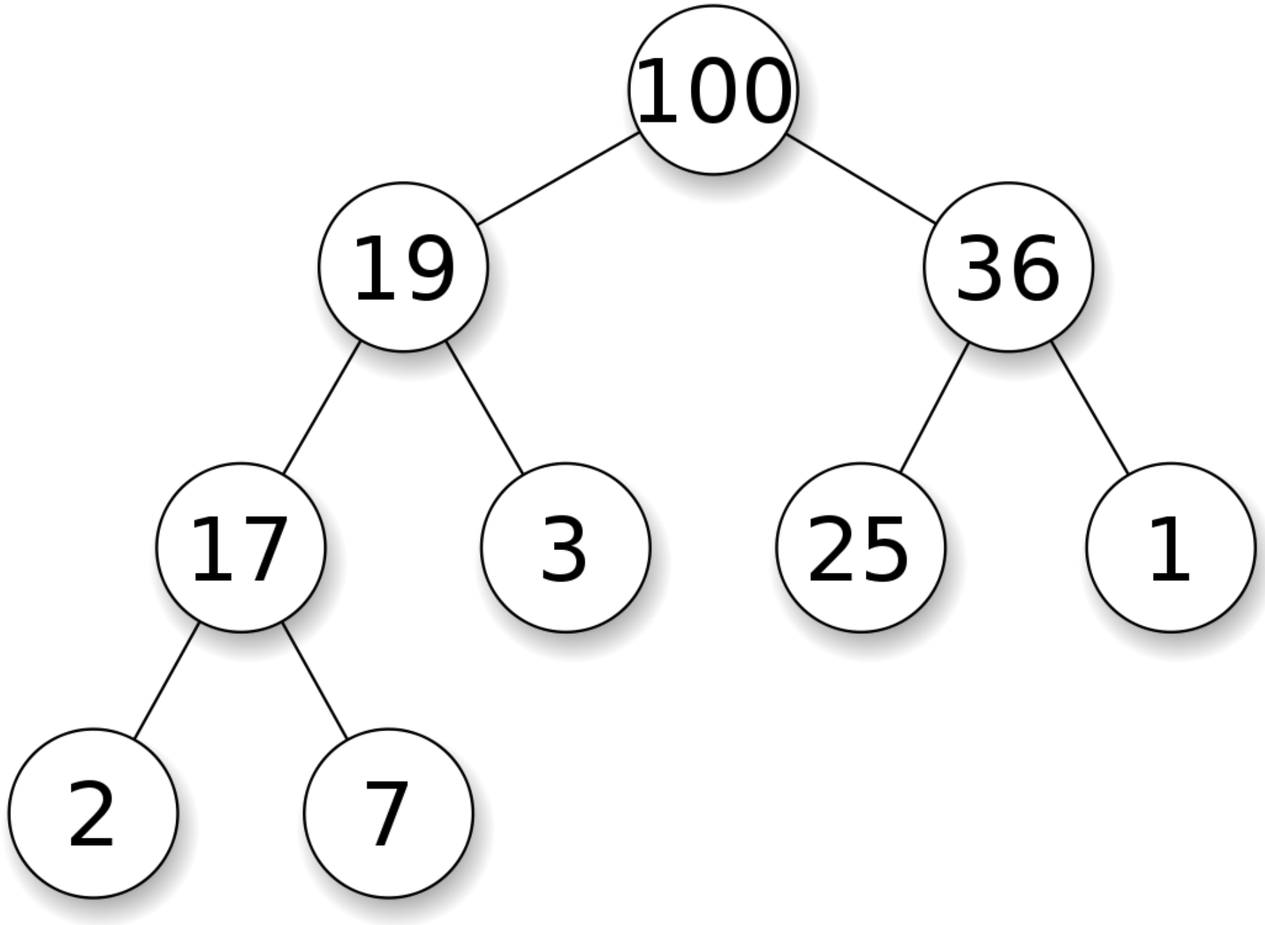


TIPOS HEAP

- **HEAP MÁXIMO:**

No heap máximo nó raiz vai conter a maior chave, de forma que todo nó filho é menor ou igual ao nó pai (esquerda e direita); ou seja, **o maior elemento sempre estará guardado na Raiz**.

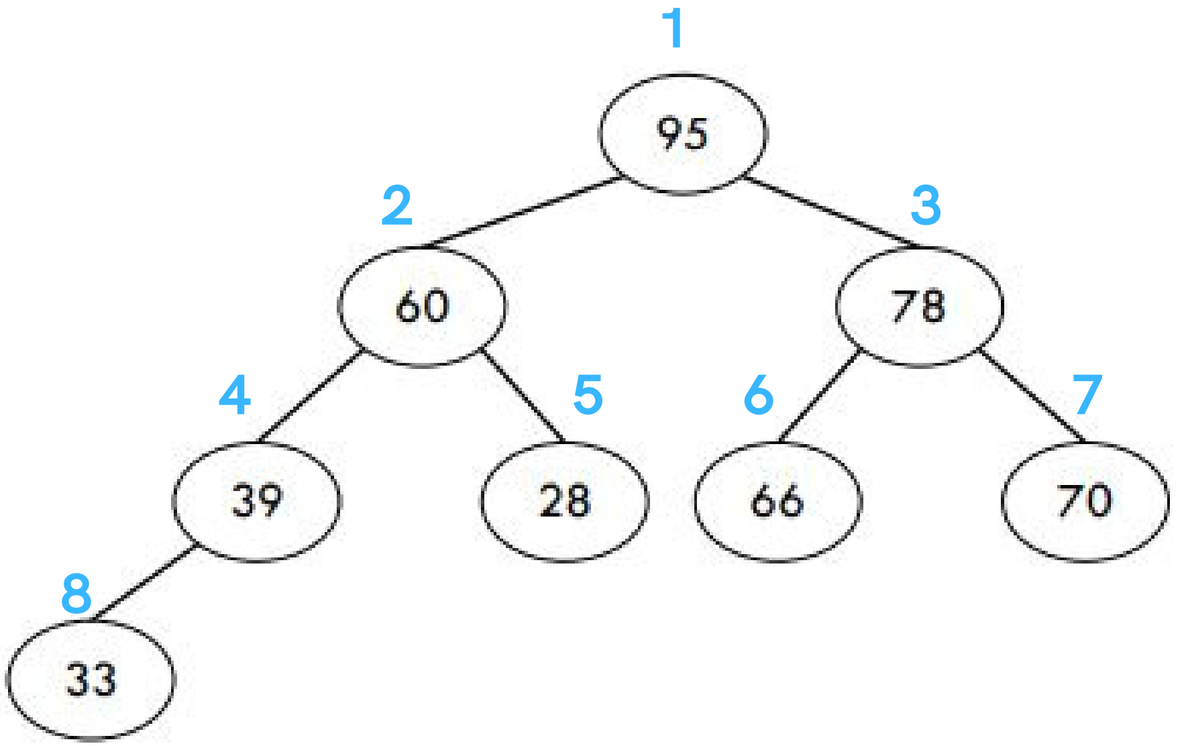
HEAP MÁXIMO EXEMPLO:



PROPRIEDADES

- Cada nó possui prioridade maior do que seus dois filhos
- O elemento de maior prioridade é sempre a raiz da árvore
- A representação em memória pode ser feita usando um vetor

EXEMPLO:

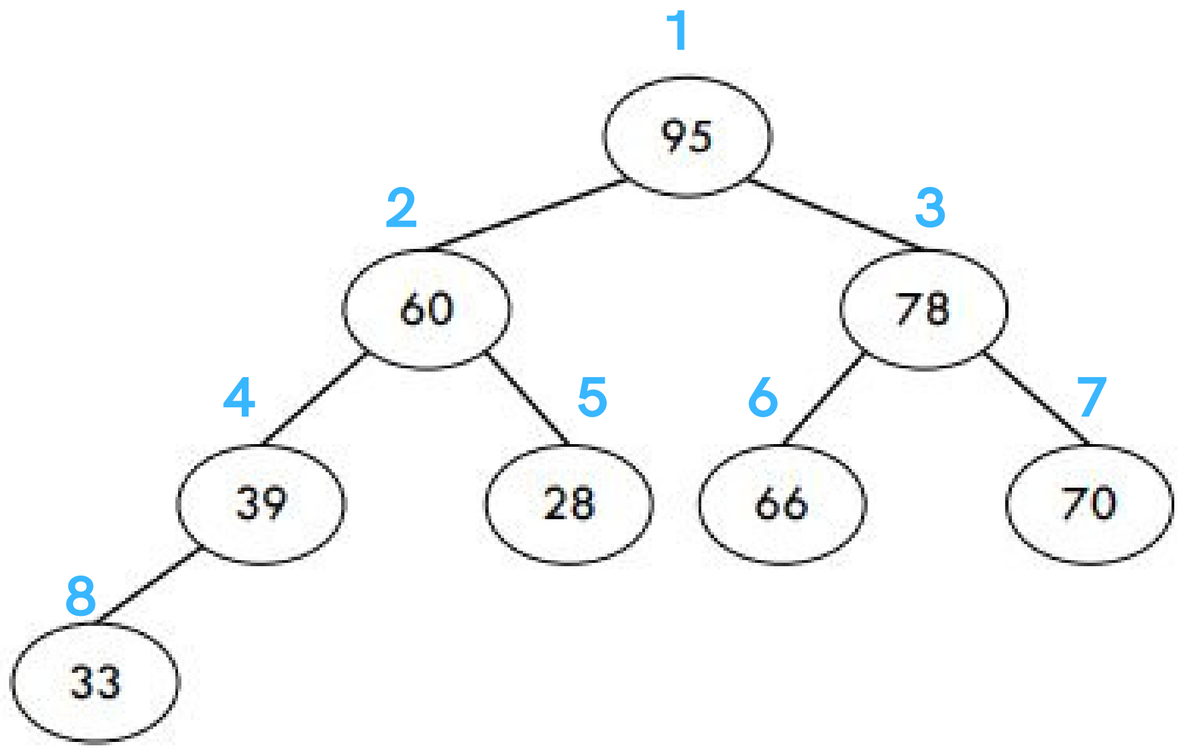


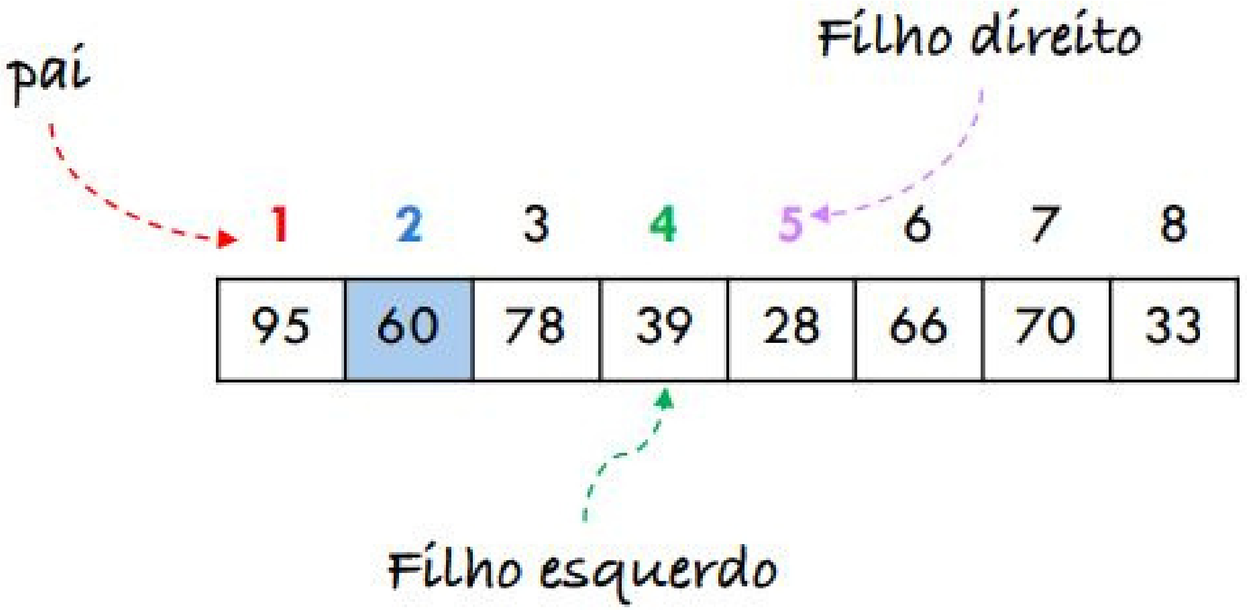
1	2	3	4	5	6	7	8
95	60	78	39	28	66	70	33

PROPRIEDADES

Para um determinado elemento i:

- Pai de i é i/2
- Filho esquerdo é i * 2
- Filho direito de i * 2 + 1





INSERÇÃO NA ÁRVORE

Sempre que houver uma inserção de um elemento na árvore, este sempre será inserido no último nível da árvore, na posição mais à esquerda possível.

INSERÇÃO NA ÁRVORE

Para organização da árvore é utilizada uma função chamada **HEAPFY-UP** - ela é responsável por comparar o elemento inserido com os demais elementos da árvore. Caso a árvore seja um Heap Mínimo, por exemplo, esta função compara os elementos de forma que o Nó Pai seja sempre menor ou igual aos Nós Filhos; ou seja, essa função tem por objetivo manter as características de ordem de uma árvore do tipo Heap, sendo ela Mínima ou Máxima, sempre trocando os elementos de lugar.

REMOÇÃO DA ÁRVORE

As remoções feitas na árvore Heap são diferente das demais árvores, pois só é removido o elemento que está na Raiz.

REMOÇÃO DA ÁRVORE

O funcionamento das remoções se dá pela seguinte maneira: o nó raiz é substituído pelo último nó do último nível. No entanto, quando isto ocorre, a árvore perde sua característica de árvore Heap, por isso é utilizada a função HEAPFY-DOWN.

BUSCA NA ÁRVORE

Geralmente procura-se somente o valor presente no Nó Raiz.

HEAPSORT - DEFINIÇÃO

O algoritmo HeapSort é um algortimo de ordenação por seleção que foi desenvolvido por Robert W Floyd em 1964. Tem um desempenho em tempo de execução muito bom em conjuntos ordenados aleatoriamente, tem um uso de memória bem comportado e o seu desempenho em pior cenário é praticamente igual ao desempenho em cenário médio.

HEAPSORT - ESTABILIDADE

O HeapSort não é um algorítimo de ordenação estável, mas pode trabalhar conjuntamente com outros algorítmos de ordenação, como são os casos do Bubble Sort, Insertion Sort, e Merge Sort, que são algorítmos estáveis.

HEAPSORT - LÓGICA

- **Construir de um Heap Máximo;**
- **Trocar o Elemento da Raiz com o elemento que está na última posição do Vetor, e guardar esse valor, o retirando da árvore, isolando este em uma sequência ordenada;**
- **Rearranjar os elementos conforme as características do Heap Máximo; ou seja, deixará o maior elemento no lugar do Nó Raiz.**

Obs.: Este processo será realizado até que tenha sido verificado todos os elementos da Heap.

HEAPSORT - FUNCIONAMENTO

O HeapSort utiliza uma estrutura de dados chamada Heap para ordenar os elementos a medida que os insere na estrutura.
No final das inserções os elementos podem ser sucessivamente removidos da raiz da heap, na ordem desejada.
É essencial que a propriedade MAX HEAP seja mantida. Essa propriedade garante que o valor de todos os nós sejam menores que os de seus respectivos pais.

HEAPSORT - FUNCIONAMENTO

6 5 3 1 8 7 2 4

UTILIZAÇÃO

Muito utilizado para o gerenciamento de filas com prioridade, pois o tempo de execução, estabilidade e o custo de memória em situações onde os dados estão totalmente desordenados, costumam ser maiores. Além disso, são utilizados para agendar processos em diversos sistemas operacionais, como é o caso dos roteadores, onde a Heap é utilizada para aumentar a qualidade destes. Sem contar também que sua aplicação serve para os algorítmos de localização de caminhos em apps de IA, assim como em robótica e até em videogames.