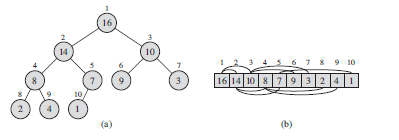
**Heap Sort**

En binær heap er et array-object som ligner et fuldt-binært træ som her:



En heap ”A” har 2 værdier. A.length som beskriver mængden af elementer der kan være i A.

Og A.heap-size som beskriver hvor mange elementer der er gemt i A: 0 <= A.heap-size <= A.length.

**Parent-Child relations**

En heap har både parent, left-child og right-click. Alle typer heaps har denne egenskab. Indexet af parent, left og right er defineret således:



Generelt er der 2 slags heaps: min-heap og max-heap.

**Typer af heaps**

En **max-heap** skal overholde egenskabet: 

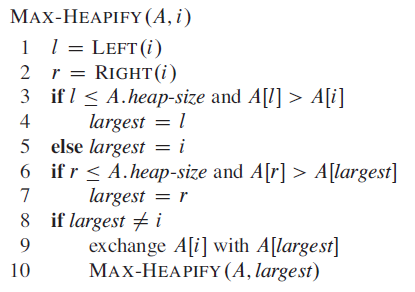
Hvilket vil sige left eller right child skal være mindre eller lig med dets parent. Dette vil sige det største element er roden af træet.

En **min-heap** skal overholde egenskabet: 

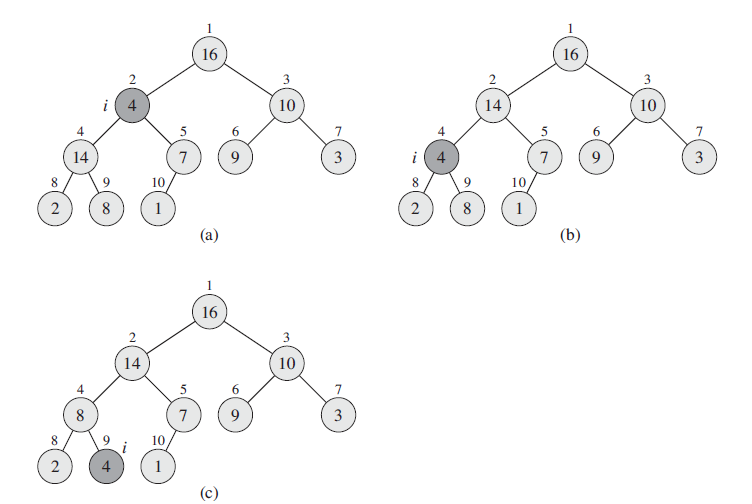
Altså det modsatte af en max-heap.

**Opretholdelse af heap-egenskabet: Max-heapify**

Max-heapify tager et array A og et index I som input. Max-heapify algoritmen er baseret på en max-heap. For en min-heap skal algoritmen ændres lidt.



Max-heapify er illustreret sådan:

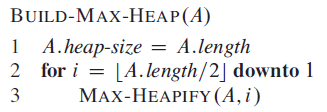


Her ser man I = 2 ikke overholdet egenskaben for en heap. Dette bliver fixet ved at bytte om på 2 og 4. Herefter bliver Max-heapify igen kaldt rekursivt på sub-træet hvor den igen bytter i = 4 og i = 9, på den måde er heapen opretholdt.

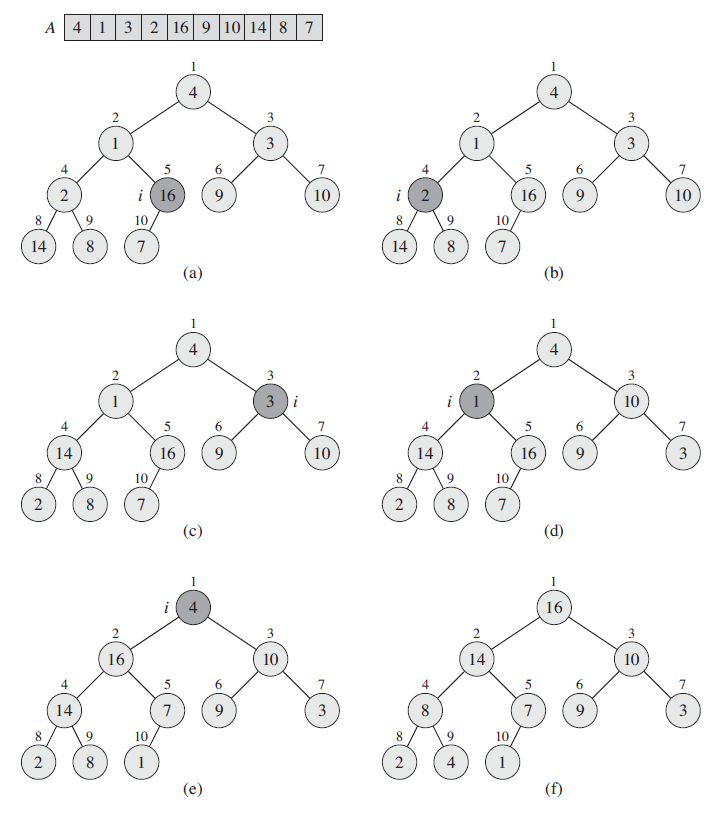
**Køretiden på max-heapify er T(n) = O(lg n) – eller O(h)**, fordi højden af en heap er lg n

**Build-max-heap**

Man bruger build-max-heap til at lave en heap. Build-max-heap tager et array A som input.



Her bliver Max-heapify kaldt iterativt på arrayet. I = A.length/2 beskriver roden af træet. Eksempel:

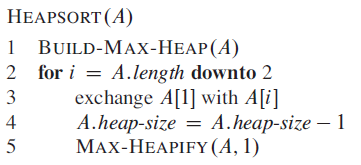


Her starter man på i = 5, fordi A.length = 10. Her efter bliver max-heapify kaldt til og med i = 1.

**Kørertiden for Build-max-heap er O(n)** – Se side 159 i bogen for bevis.

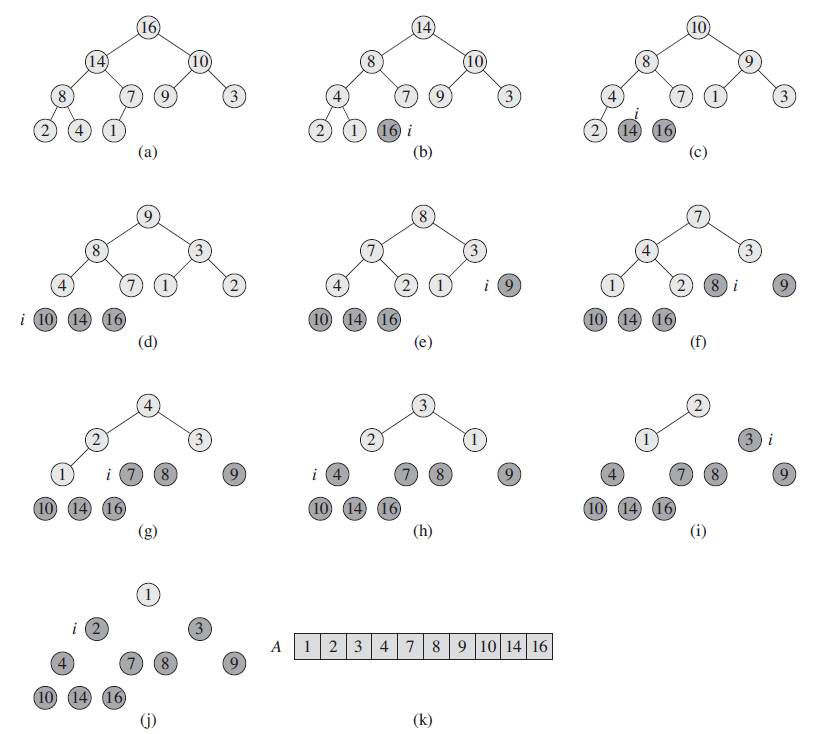
**Heap Sort algoritmen**

Heap sort tager et array A som input og giver et sorteret array i ”decending order”.



Fra heap-egentskabet ved vi, at det største element er i A[1] og placere det korrekt ved A[n] (hvor n er det sidste index). Herefter fjernes elementet fra heapen ved linje 4. Tilsidst bruger man Max-heapify til at opretholde heap-egentskaben. (herefter er A = [1...n-1]).

Kørertiden af heap sort er O(n lg n). Eksempel (efter heapen er lavet med build-max-heap):



Video eksempel: <https://www.youtube.com/watch?v=MtQL_ll5KhQ>

Asending: <https://stackoverflow.com/questions/41212072/ascending-and-descending-heapsort>