Handout
Binär Heaps & Binomial Heaps

Allgemeines

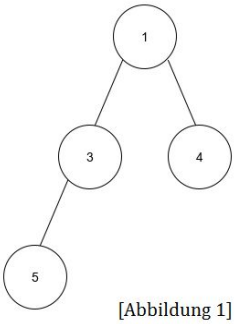
‘Priority Queues’ sind ein abstrakter Datentyp, welcher wie eine Warteschlange oder Stack (Stapel) verstanden werden kann in dem zusätzlich jeder Eintrag einen Prioritätswert enthält. Eine Priority Queue stellt mindestens die Operationen *isEmpty*, *Insert* und *extractMin/Max* zur Verfügung. Heaps sind eine Möglichkeit die abstrakte Klasse Priority Queue zu implementieren. Sie können zusätzlich die folgenden Operationen implementieren: *merge, heapifyUp / Down.*

min/maxHeap:	kleinster/größter Wert (Priorität) liegt in der Wurzel.
insert:	Fügt dem Baum einen neuen Knoten links unten hinzu.
peek:	Liefert Inhalt der Wurzel.
extractMin / Max:	Entnimmt Wurzelelement und gibt dieses zurück.
heapifyUp / Down:	sortiert Elemente im Tree nach oben (heapifyUp) oder unten (heapifyDown)
merge:	wird nur von binomialen Heaps verwendet. ‘Verbindet’ die einzelnen Teilbäume.

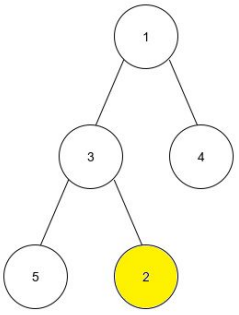
Binär Heap:

Insert (MinHeap)

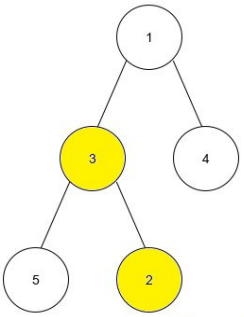
[Abbildung 1] Der Baum muss sortiert vorliegen.
[Abbildung 2b] Knoten wird unten links an den Baum gehangen.
[Abbildung 2c] Knoten wird mit dem Parentnode verglichen.
[Abbildung 3a] Wenn Childnode < Parentnode, Dann *swap*



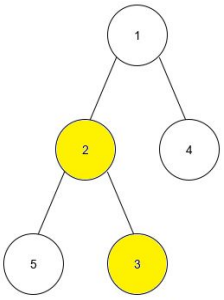
[Abbildung 1]



[Abbildung 2b]



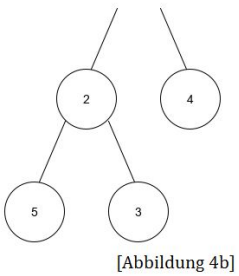
[Abbildung 2c]



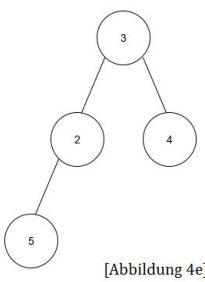
[Abbildung 3a]

***extractMin* (MinHeap)**

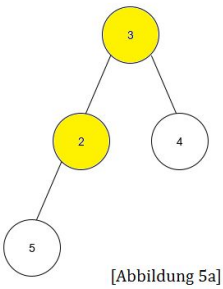
[Abbildung 4b] Wurzelelement wird entnommen und zurückgegeben.
[Abbildung 4e] Koten mit dem größten Index wird in die Wurzel gesetzt.
[Abbildung 5a] Wurzelelement wird mit den Childelementen verglichen.
[Abbildung 5c] Wurzelelement wird anschließend mit dem kleineren Childelement getauscht.
[Abbildung 6] Allen Knoten ist ein Index zugewiesen.



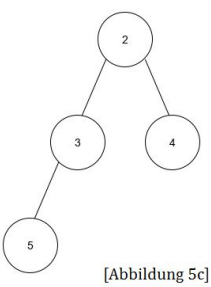
[Abbildung 4b]



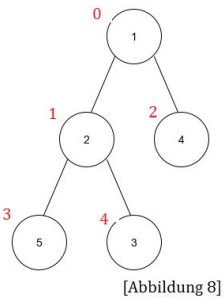
[Abbildung 4e]



[Abbildung 5a]



[Abbildung 5c]



[Abbildung 8]

Binomial Heap

Mehrere Teilbäume stellen einen Heap dar.

***Insert* (gegeben):**

Grad 0: **Grad 1:** **Grad 2:**

Zusätzliches Element (5) wird zum Heap hinzugefügt. Es gibt immer nur einen Teilbaum von jedem Grad. Aus dem Grund werden diese verbunden (merge) und zu einem Grad 1 Teilbaum.

Grad 0: **Grad 1:** **Grad 2:**

Grad 0: **Grad 1:** **Grad 2:**

Remove:

Beim Löschen / Entfernen eines Elements entstehen wieder mehrere Teilbäume, welche nach demselben Verfahren wie bei Insert wieder in verschiedene Grade aufgeteilt werden müssen. Dabei muss darauf geachtet werden, dass zu jedem Grad nur ein Teilbaum existieren darf. Dies wird mit der Operation merge wie im obigen Beispiel getan.

Man kann die Anzahl der Knoten in einem Binomial Heap mit einer Dualzahl darstellen, dies ist eine informative Darstellung über die Art und Anzahl der Binomialbäume im Binomial Heap.