Algoritmer og Datastrukturer #4

Heap og prioritetskø

Anvendelser for heap

* Sortering med *heapsort*
  + O (n log n)
  + Trenger ikke ekstra plass, slik som flettesortering
* Prioritetskøer
  + Plukke ut elementer i prioritert rekkefølge
  + Egnet for korteste-vei problemet
  + Fjerne elementer, legge til, samt endre prioritet kjapt
* En heap er en *delvis ordnet* struktur

Rask beregning av indekser

* Heltallsdivisjon med 2 kan gjøres raskere med høyreskift.
* Multiplikasjon med 2 kan gjøres med et venstreskift.
* Høyreskiftoperator: >>, venstreskiftoperator: <<

Oversikt

* Fiks heap
  + Har et komplett tre der rotas subtrær er heaper. Ordner rota slik at hele treet blir en korrekt heap.
* Lag heap
  + Har et komplett tre, lager en heap av det.
* Hent maks
  + Fjerner max-elementet (rota), omorganiserer resten så det blir en heap igjen.
* Ny prioritet
  + En node endrer verdi, må kanskje flyttes opp eller ned i heapen.
* Sett inn
  + Setter inn en ny node i heapen.
* Heapsort
  + Sorterer ved først å lage en heap og deretter plukke vekk det største elementet til det ikke er flere igjen.

Høyden på heap er alltid log2 n, der n er antall noder.

Quicksort raskere enn heapsort.