RWTH Aachen **SS 2015**
Lehrgebiet Theoretische Informatik Blatt 5
Rossmanith–Reidl–Sánchez 13. Mai 2015

Übung zur Vorlesung Algorithmen und Datenstrukturen

Aufgabe T12

Fügen Sie die Zahlen 23, 12, 5, 17, 28, 10 und 5 in einen anfangs leeren Min-Heap ein (die kleineren Zahlen sind oben). Wie sieht dieser aus?
Entfernen Sie jetzt nacheinander dreimal die kleinste Zahl aus dem Heap. Wie sieht er nach jeder der drei Operationen aus?

Aufgabe T13

Eine *Prioritätswarteschlange* ist eine Datenstruktur, welche folgende Operationen erlaubt:

1. *extract-min*: Gebe das kleinste gespeicherte Element zurück und lösche es aus der Warteschlange.
2. *insert*(x): Füge das Element x ein.

Wie können diese Operationen mithilfe von Heaps umgesetzt werden? Was ist die Laufzeit? Sie können davon ausgehen, daß anfangs bekannt ist, wieviele Elemente sich höchstens gleichzeitig in der Warteschlange befinden können.
Welche Vor- und Nachteile hat ein Heap gegenüber einer Implementierung basierend auf balancierten Suchbäumen?

Aufgabe H9

Hier ist noch einmal eine einfache Variante des Quicksort-Algorithmus:

procedure *quicksort*(L, R) :
  if $R \leq L$ **then return fi**;
  $p := a[L]$; $l := L$; $r := R + 1$;
  do
    do $l := l + 1$ **while** $a[l] < p$;
    do $r := r - 1$ **while** $p < a[r]$;
    vertausche $a[l]$ *und* $a[r]$;
  while $l < r$;
  $temp := a[r]$; $a[L] := a[l]$; $a[l] := temp$; $a[r] := p$;
  quicksort$(L, r - 1)$; *quicksort*$(r + 1, R)$

Das Array a enthalte die Zahlen $4, 1, 3, 2, 5, 9$. Welche rekursiven Aufrufe gibt es? Geben Sie den Inhalt des Arrays jeweils am Anfang jedes Aufrufs und bevor die letzte Zeile ausgeführt wurde an.

Aufgabe H10

Analysieren Sie die Laufzeit von Quicksort für den Spezialfall, daß *alle* Elemente identisch sind. Wie verhält sich Insertionsort und Heapsort in diesem Fall?