Übung zur Vorlesung Algorithmen und Datenstrukturen

Aufgabe T16

Gegeben seien die Funktionen $f(n)$ und $g(n)$. Beweisen oder widerlegen Sie:

a) $O(f) \cdot O(g) = O(f \cdot g)$

b) Falls $g = O(f)$ und $h = O(f)$ dann gilt auch $g = O(h)$

c) $\sqrt{2n(n+2)(n-1)} = \Theta(n^{3/2})$

Aufgabe T17

In der Vorlesung wurde Quicksort auch iterativ mit Hilfe eines expliziten Stacks implementiert. Da Speicherverbrauch immer ein wichtiger Faktor ist, sind wir an der maximalen Höhe dieses Stacks interessiert: Finden Sie ein Beispiel, in welchem die gegebene Quicksort-Implementation $\Omega(n)$ Paare gleichzeitig im Stack speichert. Überlegen Sie dann, wie der Algorithmus abgeändert werden kann, um diesen schmerzhaften Speicherverbrauch deutlich zu senken.

Aus Effizienzgründen bestimmt die vorliegende Implementation zunächst das minimale Element des Eingabearrays und vertauscht es mit dem ersten Element desselben, die verbleibenden Elemente werden dann wie gehabt sortiert.

```
public void quicksort() {
  Stack<Pair> stack = new Stack<Pair>();
  stack.push(new Pair(1, a.length − 1));
  int min = 0;
  for(int i = 1; i < a.length; i++)
    if(a[i] < a[min]) min = i;
  int t = a[0]; a[0] = a[min]; a[min] = t;
  while(!stack.isEmpty()) {
    Pair p = stack.pop();
    int l = p.first(), r = p.second();
    int i = l − 1, j = r, pivot = j;
    do {
      do { i++; } while(a[i] < a[pivot]);
      do { j−−; } while(a[j] > a[pivot]);
      t = a[i]; a[i] = a[j]; a[j] = t;
    } while(i < j);
    a[j] = a[i]; a[i] = a[r]; a[r] = t;
    if(r − i > 1) stack.push(new Pair(i + 1, r));
    if(i − l > 1) stack.push(new Pair(l, i − 1));
  }
}
```

Die Analyse von Quicksort setzt jedoch voraus, daß jede mögliche Permutation der zu sortierenden Schlüssel gleich wahrscheinlich ist. Sind nach der obigen Veränderung der Eingabe alle Permutationen der verbleibenden Elemente immer noch gleich wahrscheinlich?

Aufgabe H15 (8 Punkte)

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 H16 (8 Punkte)
Gegeben ist folgende Zahlenfolge: 8, 6, 3, 7, 4, 2, 20, -45

a) Wieviele Inversionen hat sie?

b) Wieviele Läufe hat sie?

c) Was macht Quicksort in der ersten Partitionierungsphase daraus?

d) Was macht Mergesort in der letzten Mischphase?