Algorithmen und Datenstrukturen I

WS 2009/10, **3. Aufgabenblatt**, Abgabe 25.11.2009

Aufgabe 9

12 Punkte

Beim Umzug eines Informatik-Instituts ist eines der Programme beschädigt worden. Glücklicherweise ist nur an einer Stelle (.......) der Ausdruck für eine Bedingung verlorengangen. Das Programm sieht nun so aus:

```
INIT(S), i=0

Solange (i<Laenge(L)):
{
  Solange ((nicht EMPTY(S)) und (.......)): { Ausgabe(TOP(S)), POP(S) }
  PUSH(S,L[i])
  i=i+1
}

Solange (nicht EMPTY(S)): { Ausgabe(TOP(S), POP(S)) }
Ende
```

(a) Welche Bedingung steht sinnvollerweise statt ...., damit das Programm die Elemente aus der Eingabe "möglichst gut" in aufsteigender Reihenfolge sortiert ausgibt? Insbesondere soll das reparierte Programm bei Eingabe von (5,4,3,2,1) als Ausgabe (1,2,3,4,5) liefern. Auf (1,5,4,3,2,6) soll mit (1,2,3,4,5,6) geantwortet werden. Zeigen Sie, wie das reparierte Programm auf diesen zwei Beispielen arbeitet, indem Sie den Stapel direkt vor und direkt nach jedem `PUSH` explizit angeben. (6 Punkte)

(b) Begründen Sie kurz, daß die Laufzeit des Programms in $O(n)$ liegt, wobei die Problemgröße $n =$ `Länge(L)` ist. Wieso kann man bereits an dieser Abschätzung des Laufzeitverhaltens erkennen, dass es Eingabelisten gibt, die vom Programm nicht korrekt sortiert ausgegeben werden? (4 Punkte)

(c) Geben Sie eine möglichst kurze Liste an, die vom Programm nicht korrekt sortiert wird. (2 Punkte)

Aufgabe 10 9 Punkte

Verwenden Sie *Selection-Sort*, *Insertion-Sort* und *Bubble-Sort* mit Abbruchkontrolle, um die folgende Liste zu sortieren: $(1, 6, 3, 8, 4, 2, 9, 5)$. Geben Sie jeweils nach Beendigung der inneren Schleife die aktuelle Liste und die Anzahl der bis dahin stattgefundenen Schlüsselvergleiche an.

Aufgabe 11 5 Punkte

Welche der folgenden Aussagen treffen zu? Begründen Sie Ihre Aussage.

(a) Es gibt Fälle, in denen die Laufzeit von QuickSort in $O(n^2)$ ist.

(b) Es gibt Fälle, in denen die Laufzeit von QuickSort in $O(n)$ ist.

(c) Es gibt Fälle, in denen die Laufzeit von MergeSort in $O(n^2)$ ist.

(d) InsertionSort kann in $O(n)$ laufen.

(e) SelectionSort kann in $O(n)$ laufen.

Aufgabe 12 9 Punkte

Das Programm `WUSEL` simuliert folgenden Prozess. Studierende, die jeder einen mit ihrem Namen versehenen Zettel auf einen Stapel S gelegt haben, versuchen, ihren jeweiligen Zettel wiederzufinden. Dazu stellen sie sich in einer Schlange Q an. Die/der jeweils vorderste Studierende schaut nach, ob der oberste Zettel des Stapels ihrer/seiner ist. Falls ja, wird der Zettel vom Stapel genommen und der/die Studierende verläßt die Schlange. Ansonsten bleibt der Stapel unverändert, und die/der Studierende stellt sich hinten wieder an. Dies wird iteriert, bis entweder der Stapel oder die Schlange leer sind. Das Programm sieht so aus:

```
WUSEL(stack S, queue Q)
{
  Solange (nicht EMPTY(S) und nicht EMPTY(Q)):
  {
    Falls (TOP(S) == FRONT(Q)) POP(S)
    Sonst                      ENQUEUE(Q,FRONT(Q))
    DEQUEUE(Q)
  }
  ENDE
}
```

(a) Welche notwendige und hinreichende Bedingung müssen S and Q beim Aufruf von `WUSEL` erfüllen, damit das Programm terminiert, also der Befehl `ENDE` irgendwann erreicht wird? (2 Punkte)

(b) Betrachten Sie ab jetzt nur noch Instanzen, bei denen das Programm terminiert. Geben Sie eine exakte Schranke $\Theta(\ldots)$ für die Laufzeit von `WUSEL` als Funktion der Problemgröße n an. Hierbei sei $n = |S| + |Q|$ die Summe der Zahlen der Elemente in S und Q bei Aufruf. Begründen Sie Ihre Antwort. (3 Punkte)

(c) Nun sei bei Aufruf $Q = [1, 2, 3, 4, 5]$. S enthalte dieselben Elemente wie Q. Geben Sie eine Startreihenfolge der Elemente in S an, so dass `WUSEL` (i) möglichst wenige und (ii) möglichst viele Schleifendurchläufe braucht. Geben Sie in beiden Fällen die Anzahl der Durchläufe an. (4 Punkte)