Technische Universität München
Institut für Informatik I-16
Dr. Tobias Lasser
Jakob Vogel

WiSe 2012/13
Übungsblatt 6
22. November 2012

Algorithmen und Datenstrukturen

Aufgabe 1 Verifikation der Multiplikation durch Addition

Auf dem letzten Aufgabenblatt haben wir eine Methode betrachtet, die mittels Additionen das Produkt $a \cdot b$ berechnet. Gegeben war eine bewusst fehlerhafte Implementierung, die nach entsprechenden Tests korrigiert wurde. Verifizieren Sie die nachfolgende korrigierte Implementierung!

Input: Ganze Zahlen a und b
Output: Produkt $c = a \cdot b$
Multiply(a,b)**:**

```
if (a = 0 ∨ b = 0) {
    c = 0;
}
else if (a < 0) {
    c = −Multiply(−a, b);
}
else {
    c = 0;
    i = 1;
    while (i ≤ a) {
        c = c + b;
        i = i + 1;
    }
}
return c;
```

a) Formulieren Sie Vor- und Nachbedingung der Methode!

b) Formulieren Sie eine Schleifeninvariante!

c) Verifizieren Sie unter Nutzung der zuvor erarbeiteten logischen Bedingungen die Korrektheit des Programmcodes.

Aufgabe 2 Rechnen mit Landau-Symbolen

Vergegenwärtigen Sie sich zunächst die Definitionen von $O(n)$ und $\Theta(n)$, wie sie in der Vorlesung vorgestellt wurden. Zeigen Sie dann folgende Behauptungen:

a) $7n^4 = O(n^5)$

b) $n^2/2 - 2n = \Theta(n^2)$

c) $2^{n+1} = O(2^n)$

d) $2^{2n} \neq O(2^n)$

Aufgabe 3 Komplexität von Selection Sort

Auf dem letzten Übungsblatt wurde der *Selection Sort* eingeführt, verifiziert, implementiert und validiert. Nun interessiert uns die Komplexität des Verfahrens, das, in Pseudo-Code, so lautete:

Input: Array $A[0..(n-1)]$ von $n \geq 0$ natürlichen Zahlen
Output: Array A aufsteigend sortiert
SelectionSort(A)**:**

```
for j = 0 to n − 2 {
    i = IndexOfMin(A, j);
    if (j ≠ i) {
        Swap(A, i, j);
    }
}
```

Input: Array $A[0..(n-1)]$ von $n > 0$ natürlichen Zahlen; Startindex j
Output: Index $i \geq j$ des kleinsten Elements im Rest $A[k...(n-1)]$
IndexOfMin(A, j)**:**

```
i = j;
for k = j + 1 to n − 1 {
    if (A[k] < A[i]) {
        i = k;
    }
}
return i;
```

Input: Array $A[0..(n-1)]$ von $n > 0$ natürlichen Zahlen; Indices i, j
Output: Array A mit Zellen i und j vertauscht
Swap(A, i, j)**:**

```
k = A[j];
A[j] = A[i];
A[i] = k;
```

a) Was ist die exakte Komplexität von **Swap** im RAM-Modell? Wie schreibt man diese in O-Notation?

b) Was ist die Laufzeit von **IndexOfMin**? Was ist der *Best Case*, was der *Worst Case*? Geben Sie auch für diese Funktion die Komplexität in O-Notation an!

c) Nutzen Sie nun die Erkenntnisse der oberen beiden Teilaufgaben und bestimmen Sie die Komplexität von **SelectionSort**!