Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Christian Scheideler
Jonas Pfoh

Wintersemester 2008/2009
Übungsblatt 01
17. Oktober 2008

Effiziente Algorithmen und Datenstrukturen I

Aufgabe 1

Please prove the following statements:

1. $\forall_{c \in \mathbb{R}_+} c \cdot f(n) = \Theta(f(n))$
2. $f(n) + g(n) = \Omega(f(n))$
3. $g(n) = \mathcal{O}(f(n)) \Rightarrow f(n) + g(n) = \mathcal{O}(f(n))$
4. $\mathcal{O}(f(n)) \cdot \mathcal{O}(g(n)) = \mathcal{O}(f(n) \cdot g(n))$

Aufgabe 2

Prove or disprove the following:

1. $n^3 + 5n^2 - 10 = \mathcal{O}(n^3)$
2. $n \ln n = \mathcal{O}(n)$
3. $n \log n = \Omega(n)$
4. $\log n = o(n)$

Aufgabe 3

Put the following terms in asymptotic order (i.e. $\mathcal{O}(A) \subset \mathcal{O}(B) \subset ...$):

$$n^2 + n \qquad 2^n \qquad \log n \qquad n \log n \qquad 1 \qquad n! \qquad \sqrt{n} \qquad n \qquad n^n$$

Aufgabe 4

Give an asymptotic upper bound (in big-O notation) of time for the worst case for the following code segments (Assume undefined functions like swap, length, etc take $\mathcal{O}(1)$ time):

1.
```
Array A
Array B
if length(A) < length(B) then
    l := length(A)
else
    l := length(B)
end if
for i = 1 to l do
    for j = l downto i + 1 do
        if A[j − 1] > A[j] then
            swap(A[j − 1], A[j])
        end if
    end for
    for k = l downto i + 1 do
        if B[k − 1] > B[k] then
            swap(B[k − 1], B[k])
        end if
    end for
end for
for i = 1 to length(A) do
    if A[i] ≠ B[i] then
        output(Sorted list A & B have the first i − 1 elements in common.)
    end if
end for
output(Sorted list A & B have the first i − 1 elements in common.)
```

2.
```
Function MergeSort(Array A):
if length(A) = 1 then
    return A
end if
int mid := round(length(A)/2)
Array A1 := A[1]...A[mid]
Array A2 := A[mid + 1]...A[length(A)]
MergeSort(A1)
MergeSort(A2)
int a := 1, a1 := 1, a2 := 1
while a1 ≤ length(A1) ∧ a2 ≤ length(A2) do
    if a1 > length(A1) ∨ A2[a2] ≤ A1[a1] then
        A[a] := A2[a2]
        a := a + 1
        a2 := a2 + 1
    else
        A[a] := A1[a1]
        a := a + 1
        a1 := a1 + 1
    end if
end while
```