Algorithms Lab 2

The in-lab problems are to be solved during the lab time. Work with your team, but write your solutions individually. You do not need to turn in your answers, only discuss them with me. Ask me for feedback on how you write your answers as well.

The homework problem set is due in one week. Work with your team, but write your solutions individually. List the people with whom you discussed the problems.

1 In lab exercises

1. Finish/review the exercises from class, which you can also find on the class website.
2. Find a simple formula for $\sum_{k=1}^{n}(2k-1)$.
3. (CLRS 3-4.a) Prove or disprove: $f = O(g)$ implies that $g = O(f)$.
4. Prove or disprove: $f = O(g)$ implies that $g = \Omega(f)$.

2 Homework problems

Your assignment will be evaluated based not only on the final answer, but also on clarity, neatness and attention to details. For example, you'll want to leave plenty of space in between problems so that we can give you feedback.

1. Consider the folowing code for BubbleSort that we discussed in class:

```
Bubble-Sort($A$)
1 For $k = 1$ to $n-1$
2     // do a bubble pass
3     For $i = 0$ to $n-2$
4         if $A[i] > A[i+1]$: swap
```

(a) Give the best-case and worst-case running time of the algorithm. Express them as $\Theta()$ bounds and give a brief justification.

(b) Show how to change the code so that the algorithm dos not do any redundant bubble-passes (i.e. if the input needs only 3 bubble passes to be sorted, the algorithm does only 3 passes).

Find a tight bound for the solution of the following recurrences using iteration.

2. $T(n) = T(n/3) + 1$
3. $T(n) = T(n/3) + n$
4. $T(n) = T(\sqrt{n}) + 1$
5. $T(n) = T(n-1) + n$
6. $T(n) = 7T(n/2) + n^3$
7. $T(n) = 7T(n/2) + n^2$