## Exercise 1

Give asymptotic upper and lower bounds for $T(n)$ in each of the following algorithmic recurrences. Justify your answer by identifying $a$, $b$ and $f(n)$ and compare the watershed function $n^{\log_b a}$ with the driving function $f(n)$.

1. $T(n) = 2T(n/4) + 1$
2. $T(n) = 2T(n/4) + \sqrt{n}$
3. $T(n) = 2T(n/4) + n$
4. $T(n) = 2T(n/2) + n^3$
5. $T(n) = T(8n/11) + n$
6. $T(n) = 16T(n/4) + n^2$
7. $T(n) = 4T(n/2) + n^2 \log n$
8. $T(n) = 8T(n/3) + n^2$
9. $T(n) = 7T(n/2) + n^2 \log n$

## Exercise 2

Finish the exercises from the first session. If you are done or want to train your creative problem solving consider the following problem:

You have $n$ nuts and $n$ bolts. Each nut goes together with one bolt, but you have forgot which ones belong together! And you are in a hurry, so the brute force solution of trying every nut on every bolt until you find a match does not work for you (what's the runtime of this approach?). Luckily, you know about QUICKSORT and you have an idea that this might help you to solve the problem in $\Theta n \log n$.

In this situation, you cannot compare two nuts or two bolts - you can only check if a bolt is to big or small to fit in a nut (or vice versa). You have solved the problem, if you have arranged both nuts and bolts in two sequences so the $i$th nut fits the $i$th bolt for all $i \in 1 \dots n$.

*Hint:* Maybe you can use the bolts to partition the nuts and the nuts to partition the bolts? Remember, that doing more than one partioning operation per recursive call does not necessarily change the asymptotic runtime.