MATHEMATICAL ALGORITHMS I

1. Homework 1:

For this first set of homework we will allow the use of python or any programming language provided you don't use external libraries to do the actual work for you. We won't allow this next time, and want to point out that memory allocation is actually part of the problem (probably the hardest part) in any fast implementation.

Efficient implementation is not the main point, but if your implementation is more than 100 times slower than the one I have, then you get a (small) penalty; don't worry about this, and just do your best. Correctness is of course important.

1.1. **Problem: Karatsuba multiplication.**

(1) Implement Karatsuba multiplication of polynomials via an array (linked list is also okay, but slower). Make sure you are using a recursive call.
(2) Implement classical multiplication of polynomials via an array (or linked list).
(3) Compare the speed of the two algorithms: for a given degree, fix polynomials f and g, and find the time you need to compute the product fg. Make a table with timings.
(4) What is the crossover point? In other words, for what degrees is Karatsuba multiplication faster than classical multiplication?

To make evaluation more convenient: we want to see the following rough layout (header files are of course okay, but the main should be short):

```
#include <very basic library for in/output>

Auxiliary functions

polynomial karatsuba_multiplication(polynomial f, polynomial g, other data)

int main()
{
   initialize f and g
   fg=karatsubamultiplication(f, g, other data)

   display i-th coefficient of fg
}
```

Remark 1.1. Comment your code, and program yourself (with a partner). You are supposed to learn something, so copying someone else's code will *not* help you. Grading will be extremely lenient, so don't worry about that.

Remark 1.2. For me the crossover point is at $\deg f = \deg g = 31$. It takes 0.4s to multiply two polynomials of degree 131071. Degree $2^{20} - 1$ polynomials takes 10.5s. Your mileage may vary (my desktop is old, so this should be achievable).

Due April 9th 2018 (send to me or to the TA's by email).