8. Assignment: algorithms for the discrete logarithm
(Due: Wednesday, 07 January 2009, 13:40, bitmax)

NOTE. Since it is Christmas, you may earn up to 15 bonus points with this assignment, i.e. the points contribute to your personal score, but not to the score which is considered 100%.

You have encountered several algorithms for the discrete logarithm problem in a multiplicative group $\mathbb{Z}_p^\times$. You have also seen the results on their time complexity and storage requirements.

Now it is time to put these results into perspective and gain some hands-on experience using a computer algebra system, e.g. SAGE, MuPAD, Maple, Mathematica, …. We start with the multiplicative group $G = \mathbb{Z}_p^\times$ where $p$ is the following 8-digit prime:

$$41723027$$

As usual the first task is to determine a generator $g$ of $G$ which will serve as the base for our discrete logarithms. You do not have to find one for yourself, since we have hidden one in the following set of group elements:

$$S = \{4, y = 1063, 1069, y^{-1} = 12049830, 41723026\} \subset G$$

You can rule out some of the elements right away. In fact, given the additional information that there is exactly one generator in this set, you can determine it without any computation.

EXERCISE 8.1 (a needle in a hay-stack). Find the generator $g$ of $G$ in the set $S$ and give a complete argument for your decision.

We want to consider the following three algorithms to solve the problem of the discrete logarithm

- the birthday algorithm for discrete logarithms (cf. algorithm 3.11, p. 68\footnote{The algorithms and pages refer to the lecture notes, version from 18 December 2008})
- Pollard’s rho method with Floyd’s trick for discrete logarithms (cf. algorithm 3.18, p. 72)

- Chinese remaindering for discrete logarithms, where you call one of the previous two algorithms for the computations in the subgroups (cf. algorithm 3.24, p. 76)

**Exercise 8.2.**  
(i) Use a programming language or a computer algebra system of your choice to implement the three algorithms.

(ii) Use your programs to compute the discrete logarithm \( \text{dlog}_g x \) where \( x \) is 24122008

Those algorithms rely on random choices, so several calls – even for the same \( x \) – will result in different CPU times.

**Exercise 8.3.**  
(i) Add a variable to your programs that outputs the total time that was needed for the computation and compare the average over 10 runs.

(ii) Formulate expectations for the runtime of your algorithm for different group orders, based on the estimates that were established in the lecture.

(iii) Put your predictions from (ii) to a test by using different groups (maybe of order near \( 2p, 4p, 8p, \ldots \)) and document your results in a table.