modulo n prime to n, where negatives are grouped with one another. Let $f: \mathcal{P} \longrightarrow \mathcal{C}$ be the map $x \mapsto x^2 \ mod\ n$. Show that this set-up is an example of Exercise 4 in the last section. This gives us a way to implement long-distance coin flips.

6. Let n be any squarefree integer (i.e., product of distinct primes). Let d and e be positive integers such that $de-1$ is divisible by $p-1$ for every prime divisor p of n. (For example, this is the case if $de \equiv 1 \ mod\ \varphi(n)$.) Prove that $a^{de} \equiv a \ mod\ n$ for *any* integer a (whether or not it has a common factor with n).
7. Prove the statements in Remark 2 about the percent of the time the different congruences for $a^{m/2}$ occur in cases (i) and (ii).

References for § IV.2

1. L. M. Adleman, R. L. Rivest and A. Shamir, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, **21** (1978), 120–126.
2. R. L. Rivest, "RSA chips (past/present/future)," *Advances in Cryptology, Proceedings of Eurocrypt 84*, Springer, 1985, 159–165.
3. J. A. Gordon, "Strong primes are easy to find," *Advances in Cryptology, Proceedings of Eurocrypt 84*, Springer, 1985, 216–223.

3 Discrete log

The RSA system discussed in the last section is based on the fact that finding two large primes and multiplying them together to get n is far easier than going in the other direction (given n, finding the two primes). There are other fundamental processes in number theory which apparently also have this "trapdoor" or "one-way" property. One of the most important is raising to a power in a large finite field.

When working with the real numbers, exponentiation (finding b^x to a prescribed accuracy) is not significantly easier than the inverse operation (finding $log_b x$ to a prescribed accuracy). But now suppose we have a finite group, such as $(\mathbf{Z}/n\mathbf{Z})^*$ or $\mathbf{F}_q^*$ (with the group operation of multiplication). Because of the repeated–squaring method (see § I.3), one can compute b^x for large x rather rapidly (in time which is polynomial in $log\, x$). But, if we're given an element y which we know to be of the form b^x (we suppose that the "base" b is fixed), how can we find the power of b that gives y, i.e., how can we compute $x = log_b y$ (where here "log" has a different but analogous meaning than before)? This question is called the "discrete logarithm problem." The word "discrete" distinguishes the finite group situation from the classical (continuous) situation.